(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
24 July 2008 (24.07.2008)

PCT

(10) International Publication Number
WO 2008/087543 A1

(51) International Patent Classification:
H04L 12/46 (2006.01) *H04L 12/56* (2006.01)

(21) International Application Number:
PCT/IB2008/000115

(22) International Filing Date: 18 January 2008 (18.01.2008)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/885,669 19 January 2007 (19.01.2007) US

(71) Applicant *(for all designated States except US)*: **TELEFONAKTIEBOLAGET LM ERICSSON (publ)** [SE/SE]; Telefonplan, S-164 83 Stockholm (SE).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **FARKAS, Jànos** [HU/HU]; Daroczi köz 9, H-6000 Kecskemét (HU). **ANTAL, Csaba** [HU/HU]; Ràkòczi út. 19, H-2340 Kiskunlachàza (HU). **TAKACS, Attila** [HU/HU]; Zrinyi u. 11, H-1195 Budapest (HU). **SALTSIDIS, Panagiotis** [GR/SE]; Asögatan 62, 1Tr, S-11829 Stockholm (SE).

(74) Agents: **WEATHERFORD, Sidney L.** et al.; Ericsson Inc., 6300 Legacy, MS EVR 1-C-11, Plano, TX 75024 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL,

[Continued on next page]

(54) Title: METHOD, BRIDGE AND COMPUTER NETWORK FOR CALCULATING A SPANNING TREE BASED ON LINK STATE ADVERTISEMENTS (LSA)

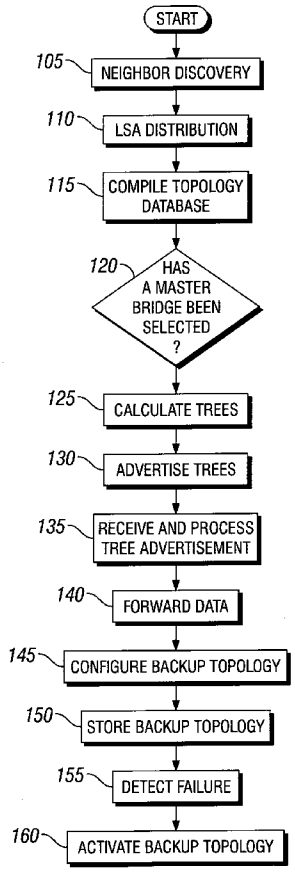

FIG. 4

(57) Abstract: There is disclosed a method and apparatus for facilitating a network, such as an Ethernet LAN, for efficient forwarding of data traffic by collecting neighbor information, generating and distributing link state advertisements, populating a topology database for the network, and calculating trees for each bridge serving as a root bridge. In a preferred embodiment the method and system also prepare one or more backup topologies, and store them for use if they are needed due to a failure condition detected in the network. In a particularly preferred embodiment, probabilities are assigned to various potential failure conditions, and the probability values are used to decide which backup topologies to calculate, store, or use.

WO 2008/087543 A1

NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

METHOD, BRIDGE AND COMPUTER NETWORK FOR CALCULATING A SPANNING TREE BASED ON LINK STATE ADVERTISEMENTS (LSA)

RELATED APPLICATION AND CLAIM OF PRIORTY

This application is related to and claims benefit of the filing date of U.S. Provisional Patent Application No. 60/885,669 filed January 19, 2007, which is entirely incorporated by reference herein.

TECHNICAL FIELD OF THE INVENTION

The present invention is directed, in general, to controlling data traffic in computer networks and, more specifically, to a method and system for controlled tree management in Ethernets and similar networks using link state principles.

BACKGROUND OF THE INVENTION

Computers, broadly speaking, are electronic machines capable of storing and manipulating information, often called data, to useful ends. Frequently, a number of computers are connected together in such a way that they are able to send data to each other. A collection of computers so connected is often called a network, and the connector between two network nodes is referred to as a link. One type of computer network is called a LAN (local area network), and may be found, for example, in the offices of a small business or educational institution. A number of LANs or other networks may also be brought into communication with each other. As might be expected, a system of rules or set procedures must be put in place so that the computers and networks can communicate with each other effectively. Such a system of rules and procedures is often called a protocol, and several have been developed for communications involving computer networks.

A widely accepted set of protocols for LAN communications has been developed under the auspices of the IEEE (Institute of Electrical and Electronics Engineers). A standard generally referred to as IEEE 802, for example, covers general network architecture, IEEE 802.1 deals with bridging and management, and IEEE 802.3 is the Ethernet protocol. An Ethernet LAN

is one that handles traffic, that is, the flow of data from one computer to one or more other, using a system of collision detection and avoidance. (A 'collision' occurs when two or more computers attempt to send data over the same link at the same time.) These standards are regularly reviewed and updated as necessary to improve networks operations and account for developments in technology.

In a typical Ethernet network, data sent from one computer to another, or from one network to another, is not transmitted all at once or continuously, but is instead broken up into discrete 'frames'. The frames may vary in length, but each frame includes sufficient address information (in addition to the actual data) so it they may be routed to its desired destination or destinations. Routing is necessary because every computer is not connected directly to every other. Instead, computers and networks are connected to intermediary devices that receive data, determine its destination address, and then route it accordingly. One such intermediary device is referred to as a bridge. A bridge is a type of software switch resident on a network component. A frame of data may be routed through many bridges on its way from source to destination.

Figure 3 is a simplified schematic diagram illustrating the bridges of an exemplary network 10, in which embodiments of the present invention may be advantageously implemented. The network may have many components, but for clarity only the bridges themselves are shown. Individual components such as computers and sub-networks may be connected to one of the bridges. In the embodiment of Figure 3, there are nine bridges, numbered 1 through 9, although the present invention is also suitable for implementation in larger or smaller networks. Each of these bridges 1 through 9 is connected by a link to one or more other bridges. In Figure 15 the links are numbered according to the bridges it connects, for example link 24 connects bridge 2 with bridge 4. When a bridge receives a data frame from, for example, a network computer or another bridge, it examines the address information and forwards the frame accordingly. Data may have to pass through several bridges from its source to destination. Without some governing protocol, however, it is possible that frames of data might inadvertently be sent from bridge to bridge, eventually

looping back to a previously-visited bridge from which they are re-sent back into the same loop. As should be apparent, this is not a desirable phenomenon and techniques have evolved for avoiding this looping problem.

One technique to avoid looping would be to use a fixed-configuration network and always route frames intended for a particular destination by the same route. Most networks are subject to change, however, and occasionally encounter failures in components and links. A static routing system is therefore not the best solution. In a more successful solution, a 'spanning tree' is calculated periodically according to a spanning tree protocol (STP). The spanning tree provides path definitions for the network as it exists at the time of the tree calculation. In the event of a failure or other event, the tree can be recalculated to adjust to the new conditions.

In Shortest Path Bridging (SPB), specified for example in IEEE 802.1aq Virtual Bridged Local Area Networks – Amendment 9: Shortest Path Bridging, Draft D0.3, May 9, 2006, an attempt is made to provide the shortest path between any two bridges of and Ethernet network. In this proposed solution, each bridge maintains a separate tree. (Or at least each 'edge bridge'; bridges that connect only to other bridges and not to any other device may not form the root of their own tree.) Frames of data arriving at a bridge directly from an end station are forwarded from the bridge to the bridge's tree, in which it is the root bridge.

The current SPB proposal is based on the path vector approach. The path vector scheme allows any two bridges to choose symmetrical paths between them, which are required for the MAC (media access control) learning process to work correctly. The path vector approach provides the shortest path (that is, the one with the least administrative cost) between any two bridges. It provides only the shortest path, however, meaning that other objectives cannot be considered even if that would be desirable. And since MAC learning is applied for the multiple spanning trees associated with the various bridges, different convergence times for the different trees may result. This produces a temporal inconsistency that may result in excessive broadcasting. It is

believed, however, that recovery time could be improved if the path vector approach could be replaced with a link state approach.

In general, the link state approach, using routing protocols such as OSPF (open shortest path first) and IS-IS (intermediate system to intermediate system), facilitates application of traffic engineering and allows the active topology to be optimized. Directly implementing these IP routing protocols, however, would mean each bridge would have to set up its own forwarding tables. This procedure is not a detriment in and of itself, and could be applied to SPB, but on the other hand it may produce long unavailable periods resulting in at least transient loops. Another complication is that Ethernet frames do not include a TTL (time to live) field (as does an IP packet) so transient loops may be problematic when changes in active topology are occurring. Of course, bridges that notice the topology is changing could simply stop forwarding data frames until a new topology is calculated, but this procedure slows down the recovery significantly.

The convergence time in the currently proposed SPB approach could theoretically be improved by a protection switching scheme where both primary and backup trees are maintained. In practice, however, a single backup tree cannot provide protection against all possible failures, and using multiple backup trees in a system, such as SPB, that requires a tree for each bridge, might significantly or even severely tax system resources.

There is therefore a need in the art for a way to implement a link state approach to support the multiple spanning trees used in SPB. The present invention provides just such a solution.

.

SUMMARY OF THE INVENTION

To address the above-discussed deficiencies of the prior art, it is a primary object of the present invention to provide a method and arrangement for implementing a link state spanning tree approach that can support an SPB (Short Path Bridging) application using of multiple spanning trees. It is a further object of the present invention to provide greater assurance of symmetric forwarding paths between any pair of root bridges on SPB trees. It is a further

object of the invention to attempt to decrease the recovery time in the event of a failure in the network.

In one aspect, the invention is a method for configuring a network, such as an Ethernet LAN, for efficient forwarding of data traffic, including collecting neighbor information, generating and distributing link state advertisements, populating a topology database for the network, and calculating trees for each bridge serving as a root bridge. The trees may be calculated in the respective root bridges themselves, or may be calculated in a master bridge if one has been selected. In either case, however, the calculation is based on the populated topology database or databases. The method may further include advertising the tree by selectively distributing tree advertisement messages, receiving tree advertisement messages in bridges of the network, and configuring the ports of each bridge according to the advertisement message. The method may also further include the feature of recognizing, in a network bridge, that a link state advertisement has already been processed, and to discard rather than forward them. In this way, a protocol for links states STP is defined that allows the application of flexible sets of routing objectives. In a preferred embodiment the method may further include preparing one or more backup topologies, and storing them for use if they are needed do to a failure condition detected in the network.

In another aspect, the present invention is a master bridge arranged for performing the method described above. In yet another aspect, the present invention is a network, such as an Ethernet LAN, arranged to perform the method of the present invention.

The foregoing has outlined rather broadly the features and technical advantages of the present invention so that those skilled in the art may better understand the detailed description of the invention that follows. Additional features and advantages of the invention will be described hereinafter that form the subject of the claims of the invention. Those skilled in the art should appreciate that they may readily use the conception and the specific embodiment disclosed as a basis for modifying or designing other structures for carrying out the same purposes of the present invention. Those skilled in the

art should also realize that such equivalent constructions do not depart from the spirit and scope of the invention in its broadest form.

Before undertaking the DETAILED DESCRIPTION, it may be advantageous to set forth definitions of certain words and phrases used throughout this patent document: the terms "include" and "comprise," as well as derivatives thereof, mean inclusion without limitation; the term "or," is inclusive, meaning and/or; the phrases "associated with" and "associated therewith," as well as derivatives thereof, may mean to include, be included within, interconnect with, contain, be contained within, connect to or with, couple to or with, be communicable with, cooperate with, interleave, juxtapose, be proximate to, be bound to or with, have, have a property of, or the like; and the term "controller" means any device, system or part thereof that controls at least one operation, such a device may be implemented in hardware, firmware or software, or some combination of at least two of the same. It should be noted that the functionality associated with any particular controller may be centralized or distributed, whether locally or remotely. In particular, a controller may comprise one or more data processors, and associated input/output devices and memory, that execute one or more application programs and/or an operating system program. Definitions for certain words and phrases are provided throughout this patent document, those of ordinary skill in the art should understand that in many, if not most instances, such definitions apply to prior, as well as future uses of such defined words and phrases.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention, and the advantages thereof, reference is now made to the following descriptions taken in conjunction with the accompanying drawings, wherein like numbers designate like objects, and in which:

Figure 1 is a flow diagram illustrating a method of tree calculation in the master bridge according to an embodiment of the present invention.

Figure 2 is a flow diagram illustrating a method of tree calculation in the master bridge according to an embodiment of the present invention.

Figure 3 is a simplified schematic diagram illustrating the bridges of an exemplary network, in which embodiments of the present invention may be advantageously implemented.

Figure 4 is a flow diagram illustrating a method of facilitating data traffic flow according to an embodiment of the present invention.

Figure 5 is a flow diagram illustrating a method of performing neighbor discovery according to an embodiment of the present invention.

Figure 6 is a flow diagram illustrating a method of processing a link state advertisement (LSA) according to an embodiment of the present invention.

Figure 7 is a flow diagram illustrating a method of processing a tree configuration message according to an embodiment of the present invention.

Figure 8 is a flow diagram illustrating a method of tree configuration in a non-root bridge according to an embodiment of the present invention.

Figure 9 depicts a LSSTP_BPDU (link state spanning tree protocol bridge protocol data unit) according to an embodiment of the present invention.

Figures 10a through 10c illustrate LSSTP parameters for an LSA_BDPU according to embodiments of the present invention.

Figure 11 illustrates LSSTP parameters for a TA_BDPU (tree advertisement BDPU) according to an embodiment of the present invention.

Figure 12 illustrates simplified LSSTP parameters for a TA_BDPU according to another embodiment of the present invention.

Figure 13 illustrates an exemplary spanning tree topology for which embodiments of the present invention may be advantageously implemented

Figure 14 illustrates a tree description for the Bridge 1 shown in Figure 9 according to an embodiment of the present invention.

Figure 15 illustrates a tree description for the Bridge 2 shown in Figure 9 according to an embodiment of the present invention.

Figure 16 illustrates a tree description for the Bridge 3 shown in Figure 9 according to an embodiment of the present invention.

Figure 17 illustrates a bridge architecture according to an embodiment of the present invention.

Figure 18 illustrates an operation of a spanning tree protocol in routing protocol according to an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

FIGURES 1 through 18, discussed herein, and the various embodiments used to describe the principles of the present invention in this patent document are by way of illustration only and should not be construed in any way to limit the scope of the invention. Those skilled in the art will understand that the principles of the present invention may be implemented in any suitably arranged computer network.

The present invention is directed to a manner of using link state protocol principles in the implementation of a SPB (shortest path bridging) scheme for facilitating the flow of data traffic in a computer network, for example an Ethernet LAN (local area network). Figure 4 is a flow diagram illustrating a method 20 of facilitating data traffic flow according to an embodiment of the present invention. At START it is assumed that a network operable to communicate data via a plurality of bridges has been formed, such as the one shown in Figure 3. The network is, for example, an Ethernet LAN network operable according to the IEEE 802 family of protocols, but other similarly arranged networks may be used as well. The process begins when the network is assembled and energized for operation, or at some later time as determined by a network operator.

Returning to the method 20 of Figure 4, in step 105 neighbor discovery is undertaken, so that each of the network bridges may learn the other bridges to which they are connected. The information discovered in the neighbor discovery step 105 is then transmitted (step 110) to other bridges via LSAs (link state advertisements). Although the LSAs flood the network, in a preferred embodiment they will be discarded (step not shown) by bridges that, for example, realize they have already learned the information contained in a particular LSA.

In this embodiment, each of the bridges then compiles a topology database (step 115). It is then determined whether a master bridge has been

selected for the network (step 120). Trees for each bridge are then calculated (step 125); this is performed by the master bridge if one has been selected, or by each bridge configuring their own tree as a root bridge. Calculating trees for each bridge as a root bridge also includes determining the port settings for the non-root bridges in each of the trees. The tree configurations are then advertised (step 130) using TAs to selected network components. Each bridge then processes the TA messages and sets its ports (step 135) according to the instructions contained in the TA. The spanning tree has at this point been configured according to an embodiment of the present invention and the routing of data traffic (step 140) may either begin, or continue, as the case may be.

In accordance with a preferred embodiment of the present invention, one or more backup topologies are then configured (step 145). This is best done when the primary active topology has been established and the network is stable, but can also be performed earlier (not shown) and advertised at the same time as the active topology. In the preferred embodiment, however, the backup topology or topologies are stored (step 150). There are a number of storage options available. The backup topology could be stored at the master bridge, if one has been selected. (Presumably, if one has been selected the backup topology has also been calculated there.) It could also be stored by including corresponding tree configurations in tree advertisement messages and distributing them. The recipient bridges, upon receiving the backup tree advertisement messages, preferably recognize them and such and simply store them. These may in some cases be each root bridge for a backup tree, or could include all of the bridges in the tree.

In either case, when a failure condition is detected (step 155), that is, when data forwarding cannot for some reason proceed according to the active topology, a backup topology is activated (step 160). The method of activation will vary according to the storage method selected. If stored at the master bridge, for example, the backup topology is advertised in the normal fashion. If backup tree advertising messages have already been distributed, however, an activation message of some kind is sent, alerting the affected bridges to retrieve the stored advertising message and act on it. Where multiple backup

topologies have been calculated, of course, only one is typically selected for activation.

In a particularly preferred embodiment, an analysis (not shown) of potential failure conditions is made, and a probability value assigned to each one. When this is done, the calculation of backup trees may be limited to only those scenarios having a higher probability of occurrence. The probability values may also be used when selecting one of a number of previously stored backup topologies for implementation.

The variation operations associated with embodiments of the present invention will now be described in greater detail. For this purpose, it will be presumed that various of the message described above, such as LSAs and TAs, are in the format of standard BPDUs (bridge protocol data units) that have been modified according to the present invention for their respective purpose. The use of modified BPDUs in this way is preferred but not required.

Figure 5 is a flow diagram illustrating a method 25 of performing neighbor discovery according to an embodiment of the present invention. Again, at START, the network is presumed to be physically assembled and operable according to the method of the present invention. In this embodiment, neighbor information is collected via neighbor discovery messages, and preferably ND_BPDUs (step 205). Other types of messages may be used for collection of neighbor information may be used in other embodiments. In this embodiment, ND_BPDUs are sent and received according to a predetermined time period (the "Hello Time Period"). The received ND_BPDUs are examined and the information stored in a topology database (step not shown) for later reference.

As ND_BPDUs are received each Hello Time Period, many times they will contain information already stored in the topology database. At other times, a new neighbor may manifest itself, or an expected ND_BPDU is not received, signaling a change in the configuration of the network. Therefore, each Hello Time Period each bridge examines the ND_BPDUs received and determines (step 210) if a new link has appeared or an existing one has timed out. That is, entries in the topology database may be timed out and erased after a

predetermined number of Hello Time Periods if no ND_BPDU confirming their validity has been received. If no change occurs, the process simply returns to step 205 and collecting neighbor information as it is disseminated each Hello Time Period. If a change has been perceived, the bridge distributes LSA_BPDUs (step 215) to advertise the change. The topology database may also be updated at this time (step not shown). Each bridge then determines if a new tree calculation is necessary (step 220) and, if so, performs a tree calculation (step 225) with the newly acquired network information. Note that in accordance with the present invention, each bridge that receives data from a network element performs the tree calculation process. In accordance with the present invention, the bridges may run any active topology algorithm to build up the tree.

The LSA_BPDUs distributed in step 215 are subsequently received in the various network bridges. Figure 6 is a flow diagram illustrating a method of processing a link state advertisement (LSA) according to an embodiment of the present invention. When an LSD_BDPU is received (step 230) at a bridge it is first examined (step 235) to determine whether it is outdated or has been received previously. This may be done using one or more techniques. For example, the receiving bridge may examine a sequence number and sending-bridge identifier of the LSA_BPDU. If a newer LSA_BPDU has already been received from that bridge, the one being examined may be discarded. In another embodiment, a path vector mechanism is employed and the receiving bridge checks to see if its own bit is set to determine whether the LSA_BDPU has been seen before. In yet another embodiment, a time-to-live field (TTL) may be utilized, with LSA_BDPUs being examined for age. If one or more of these methods indicates that the LSA_BPDU is no longer useful, it is discarded (step 240). This helps to prevent LSA_BPDUs from continuing to circulate unnecessarily in the network when they are no longer useful (although, depending on the determining technique used, they may under certain conditions be processed more than once in the same bridge).

If the LSA_BDPU has not been received before, then it is forwarded (step 245) on each port of the bridge, except for the one on which it was

received, and the bridge's topology database is updated (step 250). As in Figure 5, each bridge, having updated its topology database, then determines if a tree change is necessary (step 255) and, if so, performs a tree calculation (step 260) with the newly acquired network information.

In SPC forwarding path control, each bridge (or at least each bridge though which data is received from an end component) calculates it own tree, assuming for itself the position of root branch, based on its topology database. In an alternate embodiment of the present invention, a master bridge calculates all necessary active topologies.

The tree calculation process undertaken by the root bridge according to one embodiment of the present invention is shown in Figure 1. Figure 1 is a flow diagram illustrating a method 35 of tree calculation using a master bridge according to an embodiment of the present invention. At START it is again presumed that an operable network is in place, and in this embodiment that a suitable master bridge has been selected and configured. The master bridge may be the most powerful bridge available in the network, or another bridge may be selected for other reasons. A master bridge may, in some application, simply be selected randomly, so long as it has the capabilities necessary to perform master-bridge functions. If a master bridge fails, of course, another may have to be selected, and a backup may be designated in advance for this purpose. The master bridge selection may be the bridge with the highest priority. In some cases the network operator may configure the priority of one or more bridges to influence the master bridge selection. An advantage of using a master bridge for all tree calculations is that only one (or perhaps only one plus one backup) bridge need have the necessary capabilities for the task. Other bridges that are not anticipated to have to perform tree calculation may be simpler in design, and perhaps less expensive to acquire.

The method 35 of Figure 1 begins with determining whether a change in the network topology that requires at least one new tree calculation has occurred (step 265). (Not all topology changes will require recalculation.) This determination may be performed at regular intervals, or upon the happening of a triggering event, or both. If a tree change is not necessary, of course, the

ordinary operation of the network may simply continue. If at least one tree change is determined to be necessary, then the method 35 proceeds to the calculation of all necessary new trees using the current topology database (step 270). The master bridge then configures discarding ports (step 275), followed by a configuration of forwarding ports (step 280). The tree configuration is then distributed throughout the network (step 285), for example using TA_BPDU messages.

In this embodiment, when a TA_BPDU (or other tree advertisement message) is received at a non-master bridge, it is processed and forwarded as necessary. Figure 7 is a flow diagram illustrating a method 40 of processing a tree configuration message in a non-master bridge according to an embodiment of the present invention. The process begins with determining whether a TA_BPDU has been received (step 290). If not, of course, the routing of data traffic, if any, may continue normally. If so, the TA_BPDU is read (step 295), and the discarding ports, of any, are configured (step 305). The discarding ports do not have to be configured first, but his order is preferred and helpful to creating a loop-free topology. The forwarding ports, if any, are then configured (step 310). Assuming at least one forwarding port has been configured, the TA_BPDU is then forwarded on each of the bridge's forwarding ports (step 315).

As each bridge in the network is made aware of the network topology though the neighbor discovery and LSA distribution described above, each may be set to anticipate the arrival of a TA_BPDU originating from the master bridge when a topology change is detected (or within a certain interval thereafter). In this case, when a TA_BPDU does not arrive when expected, a master bridge failure may be indicated and a new master will have to be selected according to the backup procedure for the network in question. Alternately, upon detecting a master bridge failure, the bridge may perform its own tree calculation. Figure 2 is a flow diagram illustrating a method 50 of tree configuration in a non-master bridge according to an embodiment of the present invention. This approach is referred to as RCOPB (Root Controlled Optimal Path Bridging). The process begins with determining that a change in the bridge's owned tree necessitating

a new tree calculation has occurred (step 450). Not all topology changes, of course, will necessitate a tree recalculation. Where the removal of a link or bridge is detected, for example, that was not part of the tree owned by the bridge, a new tree need not be calculated. If no tree calculation is needed, of course, the routing of data traffic, if any, may continue normally. If recalculation is necessary, however, the bridge proceeds to calculate its owned tree with itself as the root bridge (step 455). The forwarding ports are then configured (step 460), followed by configuration of the discarding ports, of any (step 465). The discarding ports may also be configured first, before the forwarding ports. A TA_BPDU advertising the calculated tree is then forwarded on each of the bridge's forwarding ports (step 470). Note the method of Figure 2, RCOPB, may also be used for reasons other than master bridge failure, and of course may be used in networks where no master bridge designation is made.

When the TA_BPDU (or other tree advertisement message) is received at a non-root bridge of the particular tree, it is processed and forwarded as necessary. Figure 8 is a flow diagram illustrating a method 55 of tree configuration in a non-root bridge according to an embodiment of the present invention. The process begins with determining whether a TA_BPDU has been received (step 475). If not, of course, the routing of data traffic, if any, may continue normally. If so, the TA_BPDU is read (step 480), and the discarding ports, of any, are configured (step 485). The discarding ports do not have to be configured first, but his order is preferred and helpful to creating a loop-free topology. The forwarding ports, if any, are then configured (step 490). Assuming at least one forwarding port has been configured, the TA_BPDU is then forwarded on each of the bridge's forwarding ports (step 495).

The format of BPDU messages follows generally the MSTP BPDU format of IEEE 802, with certain differences and special considerations as set forth below and in Figures 9 though 17. Figure 9 depicts a LSSTP_BPDU (link state spanning tree protocol BPDU) 45, according to an embodiment of the present invention. Regarding the fields of the LSSTP_BPDU especially related to implementation of the present invention, the BPDU type field 320 will reflect that this is an LSSTP_BPDU message. The Hello Time field 325 is used only

in ND_BPDUs, not in other types of LSSTP_BPDUs. Note that the tree configuration identifier field 330 is the same as is proposed in IEEE 802.1ap Virtual Bridged Local Area Networks – Amendment 9: Shortest Path Bridging, Draft D0.3, dated May 9, 2006. The LSSTP Parameters 335 vary according to the type of LSSTP_BPDU, as described in more detail below.

For an ND_BPDU according to the present invention, the LSSTP Parameters 335 are empty, as the necessary corresponding-link bridge and port identifiers are currently specified in the proposed IEEE 802 MSTP BPDU. In other words, the neighbor discovery process does not require additional LSSTP parameters for successful operation.

For LSA_BPDUs according to the present invention, there are several options available. These options correspond generally with the different embodiments, described above in the context of discarding LSAs when they are outdated or no longer useful. These options are illustrated in Figures 10a through 10c, which show LSSTP parameters for an LSA_BDPU according to three embodiments of the present invention. In each of these embodiments, the respective LSSTP parameters include a Flags field 340 and a LSSTP Computation Protocol Identifier field 345, followed by link description information 355. In Figure 10a, an optional LSA-removal field 350a includes a Sequence Number along with the Bridge ID of the sending bridge. In Figure 10b, optional LSA-removal field 350b includes a Path Vector. The LSA removal options associated with each of these LSA-removal fields has been described above (in reference to Figure 2). In a third embodiment, the TTL information already part of the standard BPDU message is used instead – and therefore no LSA-removal field is necessary - as shown in Figure 10c. Note that more than one of these LSA-removal options may be used, with appropriate changes made to the LSSTP parameters if needed, but using more than one option is not presently preferred. In an alternate embodiment, no LSA removal option is used at all, but this is not recommended.

Figure 11 illustrates LSSTP parameters for a TA_BDPU (tree advertisement BPDU) according to an embodiment of the present invention. This type of message also includes a Flags field 360 and a LSSTP

Computation Protocol Identifier field 365. A Tree ID field 370 follows, and a Number of Bridges field 375, which of course specifies the number of bridges in the tree. The individual Bridge fields 380 follow, one for each bridge in the identified tree. The individual Bridge fields each identify the port configuration for one of the bridges in the tree. In this embodiment, the designated forwarding ports for each bridge are specified, and the remaining ports at that bridge are assumed to be discarding ports and configured accordingly. As mentioned above, these discarding ports are preferably configured first, prior to setting the forwarding ports. The root port is also specified, in this embodiment immediately following each Bridge ID. An alternate port toward the root bridge may also be listed (but is not shown in Figure 11).

Figure 12 illustrates simplified LSSTP parameters for a TA_BDPU according to another embodiment of the present invention. Here, following the Tree ID field 370, a Number of Links field 385 relates the number of links in the tree rather than the number of bridges. Links fields 390 therefore follow the Number of Links field 385, and describe the tree in terms of the links between bridges. This reduces the size of he TA_BPDU, but increases the processing complexity as each bridge has to determine the how to set their ports based on the received tree description. Whether this trade-off is desirable may vary from network to network.

An even more abbreviated form may be useful in some applications. The exemplary tree topology of Figure 13, for example may be advertised using the TA_BPDU sequences illustrated in Figures 14-16. As can be seen in Figure 13, Bridge 1 has two designated ports and hence two branches leading from it, specifically to Bridges 2 and 3. Bridge 2, in turn, has three branches leading out to Bridges 4, 5, and 6, and Bridge 3 had one branch leading to Bridge 7. Bridges 4 through 7 are leaf bridges, and therefore have no branches leading from them.

According to this embodiment, a first TA_BPDU message is received in Bridge 1; this message (or rather the relevant portion thereof) is illustrated in Figure 10. This type of message also includes a Flags field 405 and a LSSTP Computation Protocol Identifier field 410. A Tree ID field 415 follows. After the

Tree ID field 415 is a first Tree Description field 420. The first Tree Description field includes the information needed at Bridge 1, namely, the relationship of the links and bridges that follow. As represented in Figure 14, a first set of brackets encloses the identities of all of the links and bridges associated with port 1 of Bridge 1, and a second set of brackets encloses the identities of all the links and bridges associated with port 2 of bridge 1. Regarding the latter, port 2 is designated to a link to Bridge 3, and port 1 of Bridge 3 is designated to link to bridge 7. Port 1 of bridge 1 is also designated to link to a single bridge, Bridge 2, but since Bridge 2 has three ports, each designated to link to a respective one of Bridges 4, 5, and 6, the designations of Bridge 2's ports are each isolated in a set of interior brackets in the Tree Description field 420 of Figure 10.

Naturally, Bridge 1 uses the information in the Tree Description field 420 to configure its own ports 1 and 2. It then transmits one modified version of TA_BPDU message to Bridge 2 and another to Bridge 3 using the appropriate ports. In each case, the port designations not applicable to the recipient bridge are removed by Bridge 1 prior to sending. The TA_BPDU sent to Bridge 2 is illustrated in Figure 15. In Figure 15, it can be seen that in the Tree Description field 420-2, all that remains are the port assignments for Bridge 2; port 1, for example is designated to link to Bridge 4. Similarly, Figure 16 illustrates the TA_BPDU, which is transmitted from Bridge 1 to Bridge 3. In Figure 16, it can be seen that in the Tree Description field 420-3, all that remains is the port assignment for port 1 of Bridge 3, which is designated to link to Bridge 7. For example is designated to link to Bridge 4. Note that Bridges 4 through 7 are leaf bridges and as such have no forwarding ports to set. The TA _BPDUs (not shown) sent to these Bridges are similar to those depicted in Figures 14 through 16, but they will have a NULL value for their Tree Description field.

Figure 17 illustrates a bridge architecture according to an embodiment of the present invention. Note that IP link state routing protocols such as OSPF or IS-IS are not suitable without modification for application to forwarding control in Ethernet networks because loop prevention is not assured. In accordance with the present invention, however, the routing protocol entity is implemented

as a higher layer as shown in Figure 17. Operation of the STP (spanning tree protocol) entity as a higher layer entity is illustrated in Figure 18. The routing protocol entity is attached to each port of a bridge similarly to the SPT entity. In this way, the routing protocol entity is able to send and receive frames on each port. For a routing protocol messages that is to flood the network, the routing protocol entity sends out the message on each port except the port on which the message was received. Other messages are processed by the routing protocol entity and only sent out on designated ports.

As mentioned above, the routing protocols are not applicable without some modification. First, IP addresses are replaced by MAC addresses in the routing protocol and each frame is them forwarded according to the MAC address in its header. Second, tree advertisement is inserted into routing protocol messages and the processing of tree advertisements is implemented in network bridges, as described in more detail above. Note that tree advertisements may be carried in routing protocol objects that are prepared for routing-protocol extensions. For OSPF routing, for example, a new type of Opaque LSA may be required. Opaque LSAs in OSPF provide a generalized for protocol extensions, and tree advertisement may be considered a type of extension. In a preferred embodiment, type-9 (link local) is used for tree advertisement Opaque LSAs. Tree advertisement may also be implemented in an IS-IS routing protocol using similar protocol extensions applicable there.

In accordance with the present invention, tree advertisement messages (TA_BPDUs, for example) should not be flooded, to avoid accidental loops. Rather, they are forwarded only on links that are part of the tree that they advertise, and are sent from a root bridge toward the leaves on its owned tree. Whether the tree topology is calculated in a master bridge or in each root bridge (as in RCOPB), all other affected bridges configure their ports according to the received tree advertisement.

Note that processing of the tree advertisement messages accordance with the present invention may represent a new functionality implemented in the bridges' routing protocol entity. This new functionality includes the proper configuration of the ports that are part of the tree. That is, the bridges send the

tree advertisement only to selected neighbor bridges instead of flooding all ports. The bridge may in some embodiments (as mentioned above) have to adjust the tree advertisement before forwarded it on the designated ports. The tree description schemes described above may also be implemented in routing protocols with the proper functionality.

In this manner, link state protocols may be applied for the control of forwarding, that is, configuration of the active topology, in SPB Ethernet networks.

As mentioned above, for MAC learning to function properly in an SPB environment it is important that the respective trees provide symmetric paths between any edge bridge pairs. In accordance with the present invention, any tree calculation algorithm may be used, for example the Dijkstra algorithm, so long as it provides symmetric-path assurance. This will occur if the path costs between the bridge pairs in question are unique. The present invention provides a manner of ensuring that this is the case.

In accordance with the present invention, the list of Bridge IDs used for path calculation is compiled in a way that provides a unique path cost for each path. This is accomplished by using compound path costs. As used herein, the term compound path costs means that each cost figure includes an integer part and a fractional part. The integer part represents the sum of the costs of each link in the path. The fractional part is not calculated, but rather takes a value that is actually a concatenation of the IDs of the bridges on either end of the path. For example, if two bridges with Bridge IDs 1201 and 239, respectively, are separated by two links with respective costs of 3 and 5, the path cost may be represented as 8.1201239. This path cost will always be unique because of the manner in which it is formed, even if another available path is formed by links whose actual costs sum to 8. Path symmetry between the two bridges is assured because the path cost is unique for most if not all topology optimization algorithms.

The present invention also offers support for protection switching in the network. In protection switching, protection trees are calculated in advance for use in the event of a failure. Depending on the network, these protection trees

may be calculated by a master bridge, in individual root bridges, or in a separate network management entity if one is used. The calculation is preferably done during a period of time when the network is stable and stored for later use. In one embodiment, the backup trees are not advertised immediately, but in the event a failure is detected, the stored alternate topologies are described in TA_BPDUs and distributed so that they may be implemented quickly.

In another embodiment, when the backup trees are calculated, they are then described in TA_BPDUs that are identified as advertising backup trees and distributed. TA_BPDUs that are identified as backup trees are received and stored in the affected bridges, but not acted upon immediately. In the event that a failure is detected, a topology controller entity (for example, a master bridge) simply sends a BDPU message specially configured to announce that the backup trees should be considered. If multiple alternate topologies are created, the specially-configured BPDU could also indicate which one is to be implemented.

In another embodiment, the probability of occurrence of different failure events is calculated, and alternate topologies are calculated only for those events having a probability exceeding a certain threshold value. Alternately, a desired number of alternate topologies is determined, and alternate topologies are calculated until the desired number is reached. Where multiple alternate topologies are calculated, some could be distributed according to one distribution method and others according to different distribution methods. In addition to the methods according to the present invention and described above, of course, any existing method may be used.

Although the present invention has been described in detail, those skilled in the art should understand that they can make various changes, substitutions and alterations herein without departing from the spirit and scope of the invention in its broadest form.

WHAT IS CLAIMED IS:

1. For use in a computer network having a plurality of bridges, a method for facilitating the flow of data through the network, said method comprising:

collecting neighbor information;

distributing link state advertising messages advertising collected neighbor information;

building at least one topology database based on the advertised neighbor information; and

calculating at least one spanning tree for each of the plurality of bridges as a root bridge based on the at least one topology database.

2. The method of claim 1, wherein the network is an Ethernet network.

3. The method of claim 1, further comprising advertising the calculated trees to the plurality of bridges.

4. The method of claim 1, wherein the tree calculations are performed by the bridge that forms the root of the respective tree.

5. The method of claim 1, further comprising selecting a master bridge from the plurality of bridges.

6. The method of claim 5, wherein the master bridge is selected by a network operator.

7. The method of claim 1, further comprising calculating at least one backup topology.

8. The method of claim 7, wherein calculating at the at least one backup topology is performed in a master bridge.

9. The method of claim 7, further comprising storing the at least one backup topology for use if a failure condition is detected.

10. The method of claim 9, further comprising detecting a failure condition and activating the at least one backup topology.

11. The method of claim 9, wherein the at least one backup topology is calculated and stored in a master bridge.

12. The method of claim 9, wherein the at least one backup topology is calculated in a master bridge and distributed to each of the plurality of bridges for storage until an activation message is received.

13. The method of claim 1, further comprising analyzing potential failure conditions and assigning a probability value to each condition.

14. The method of claim 13, further comprising calculating a backup topology for the failure condition having the highest probability value.

15. The method of claim 1, further comprising examining at least one link state advertising message and determining if the link state advertising message should be discarded or forwarded.

16. A bridge for use in a computer network, said bridge arranged to build a topology database based on link state advertisements and containing neighbor discovery information for a plurality of other network bridges, for calculating an active topology from information in the topology database, the active topology including a tree for each bridge of the plurality of other network bridges acting as a root bridge, and for advertising the active topology to the plurality of other network bridges.

17. The bridge of claim 16, wherein the bridge is a master bridge selected by a topology coordinating entity.

18. The bridge of claim 17, wherein the bridge is a master bridge selected based on a predetermined priority value.

19. The bridge of claim 16, wherein the bridge is further arranged to calculate and store at least one backup topology.

20. The bridge of claim 19, wherein the at least one backup topology is stored in the bridge.

21. The bridge of claim 19, further comprising transferring the at least one backup topology to a network management entity for storage.

22. The bridge of claim 19, further comprising transferring the at least one backup topology to a plurality of network bridges for storage.

23. A computer network, comprising:

a master bridge; and

a plurality of non-master bridges;

wherein the master bridge is arranged to calculate an active topology based on a topology database populated with neighbor information obtained from link state advertising messages.

24. The computer network of claim 23, wherein the master bridge is further arranged to calculate at least one backup topology.

25. The computer network of claim 24, wherein the master bridge is further arranged to store the at least one backup topology for activation in the event of a failure condition.

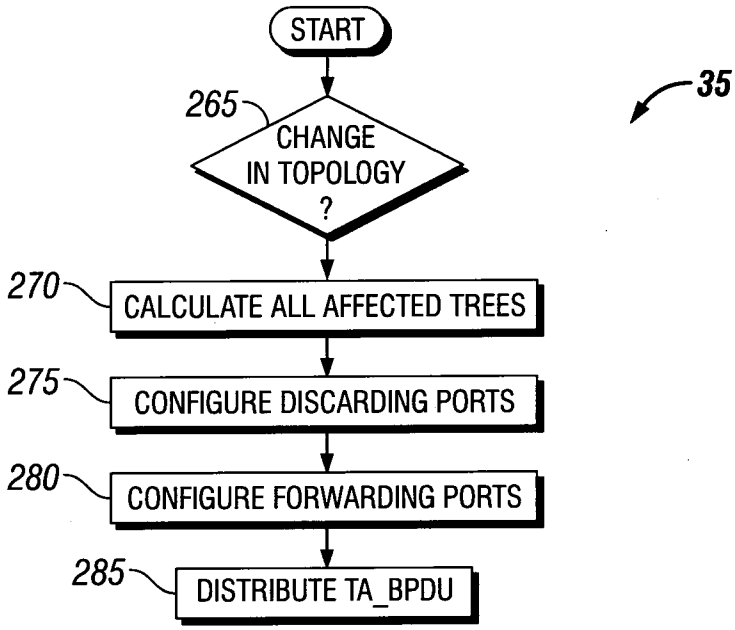
START
265
CHANGE IN TOPOLOGY ?
35
270
CALCULATE ALL AFFECTED TREES
275
CONFIGURE DISCARDING PORTS
280
CONFIGURE FORWARDING PORTS
285
DISTRIBUTE TA_BPDU

FIG. 1

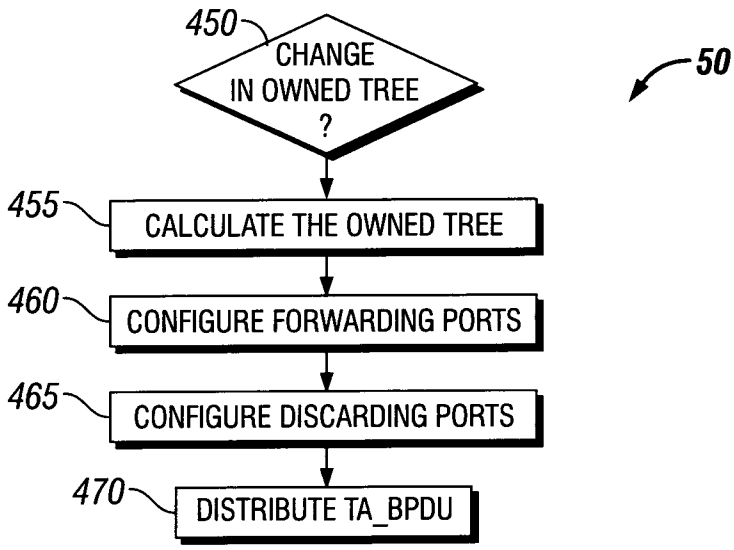
450
CHANGE IN OWNED TREE ?
50
455
CALCULATE THE OWNED TREE
460
CONFIGURE FORWARDING PORTS
465
CONFIGURE DISCARDING PORTS
470
DISTRIBUTE TA_BPDU

FIG. 2

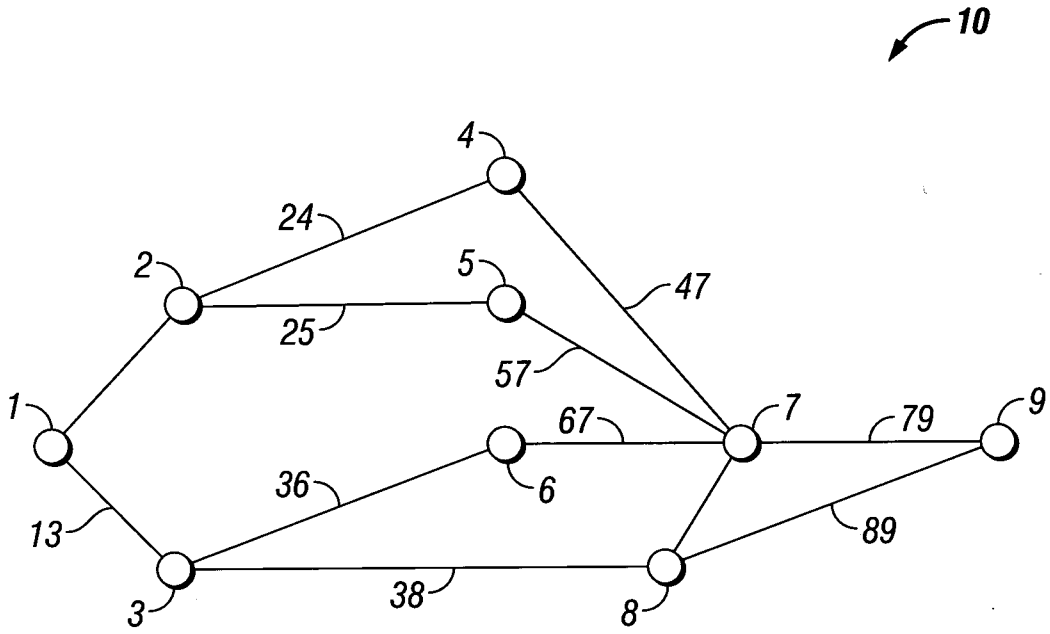

FIG. 3

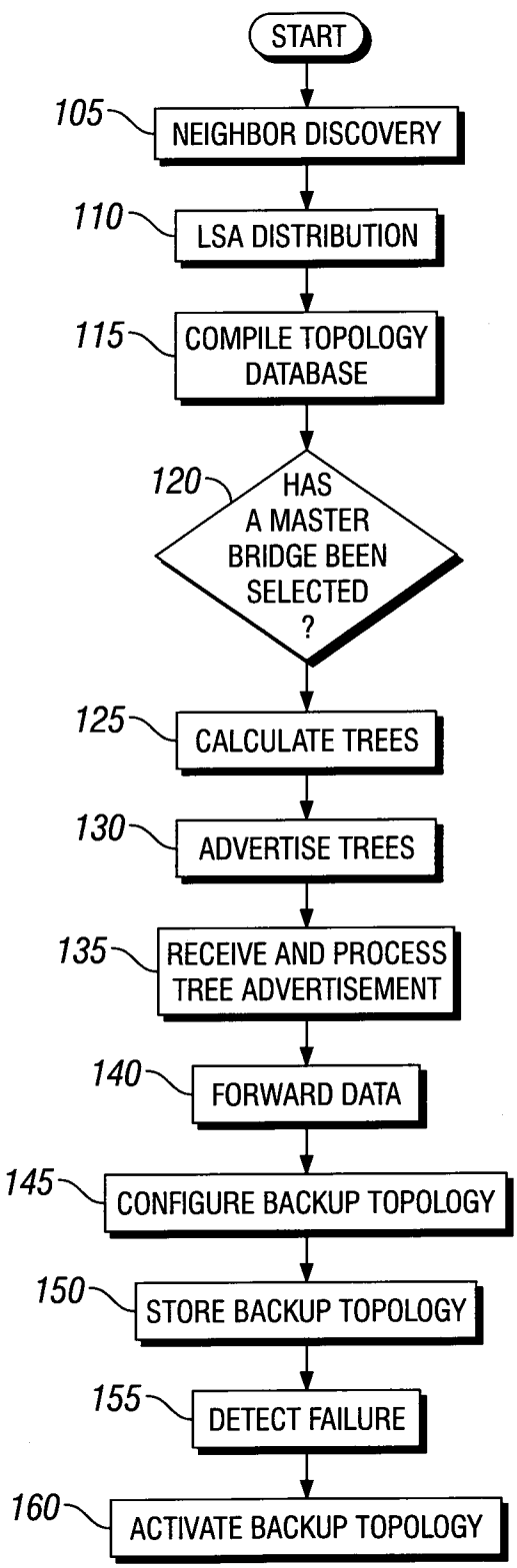
20
START
105
NEIGHBOR DISCOVERY
110
LSA DISTRIBUTION
115
COMPILE TOPOLOGY DATABASE
120
HAS A MASTER BRIDGE BEEN SELECTED ?
125
CALCULATE TREES
130
ADVERTISE TREES
135
RECEIVE AND PROCESS TREE ADVERTISEMENT
140
FORWARD DATA
145
CONFIGURE BACKUP TOPOLOGY
150
STORE BACKUP TOPOLOGY
155
DETECT FAILURE
160
ACTIVATE BACKUP TOPOLOGY

FIG. 4

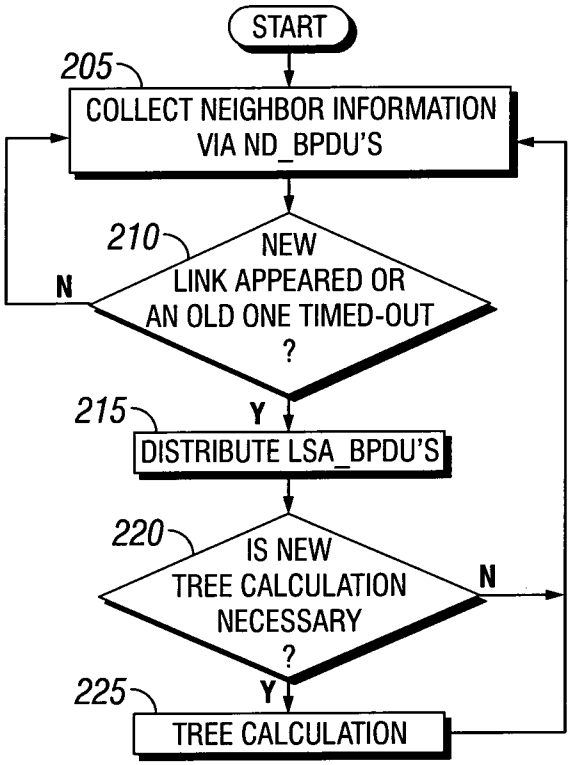
START
205
COLLECT NEIGHBOR INFORMATION VIA ND_BPDU'S
210
NEW LINK APPEARED OR AN OLD ONE TIMED-OUT ?
N
Y
215
DISTRIBUTE LSA_BPDU'S
220
IS NEW TREE CALCULATION NECESSARY ?
N
Y
225
TREE CALCULATION
25

FIG. 5

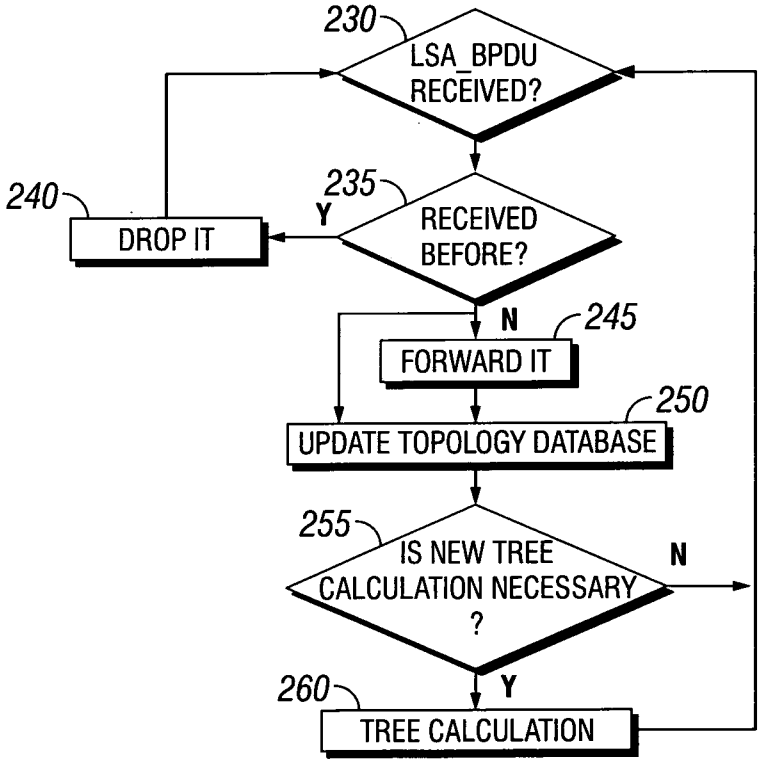
230
LSA_BPDU RECEIVED?
30
240
DROP IT
Y
235
RECEIVED BEFORE?
N
245
FORWARD IT
250
UPDATE TOPOLOGY DATABASE
255
IS NEW TREE CALCULATION NECESSARY ?
N
Y
260
TREE CALCULATION

FIG. 6

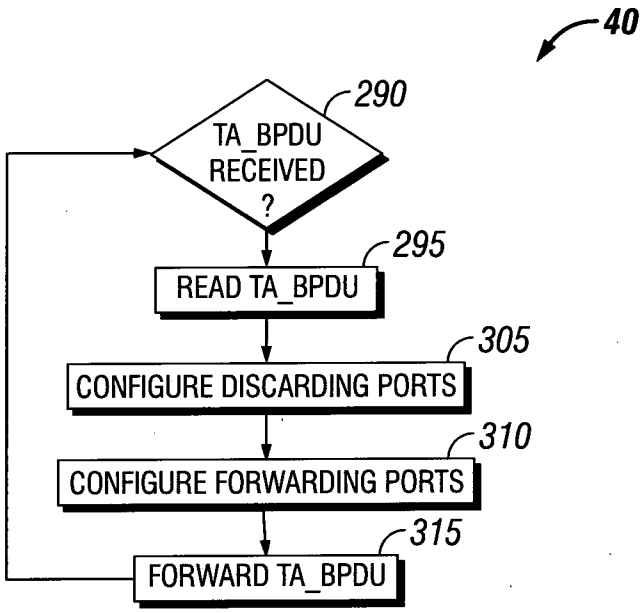
40
290
TA_BPDU RECEIVED ?
295
READ TA_BPDU
305
CONFIGURE DISCARDING PORTS
310
CONFIGURE FORWARDING PORTS
315
FORWARD TA_BPDU

FIG. 7

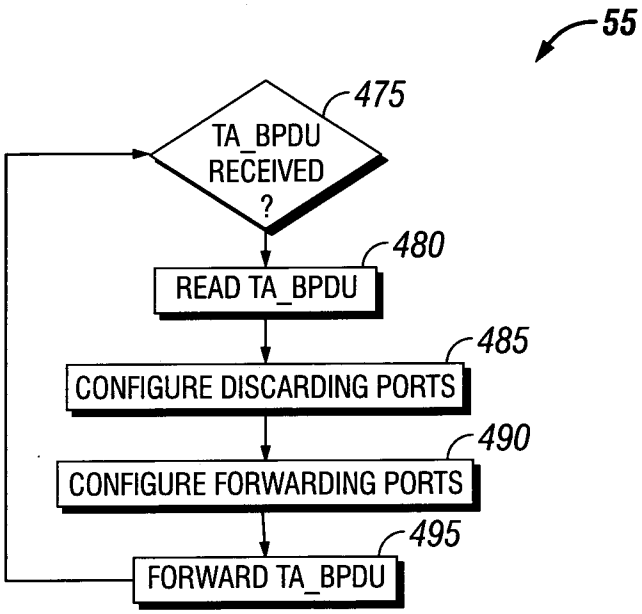
55
475
TA_BPDU RECEIVED ?
480
READ TA_BPDU
485
CONFIGURE DISCARDING PORTS
490
CONFIGURE FORWARDING PORTS
495
FORWARD TA_BPDU

FIG. 8

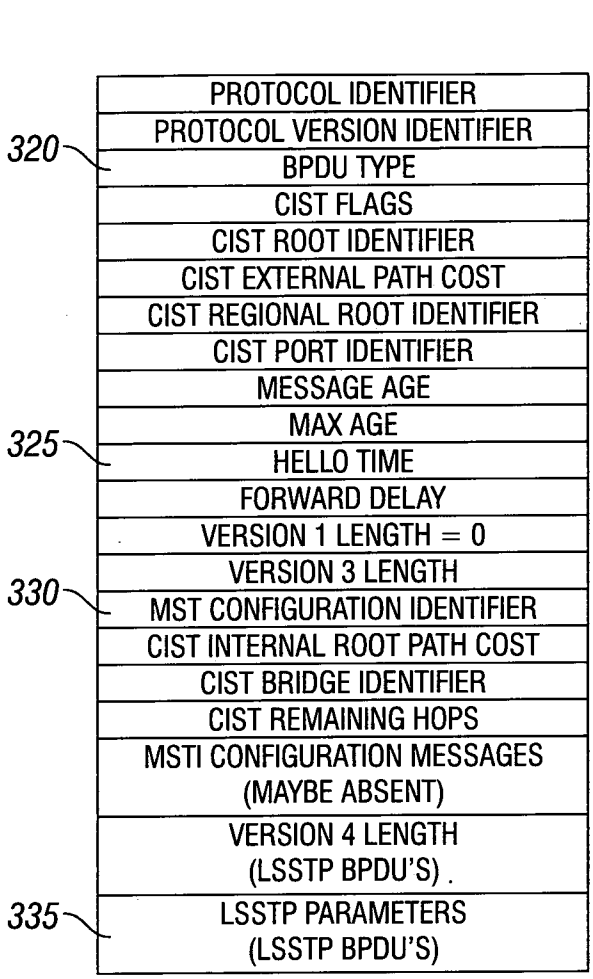

FIG. 9

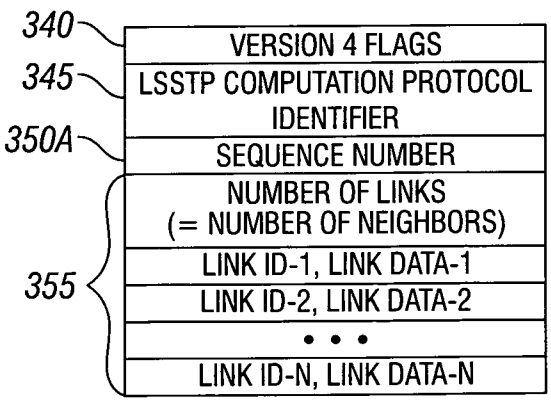

FIG. 10A

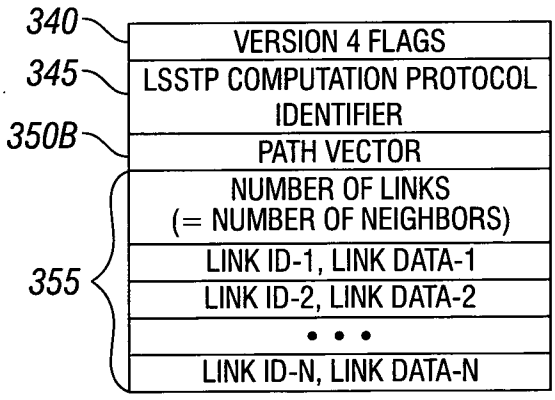

FIG. 10B

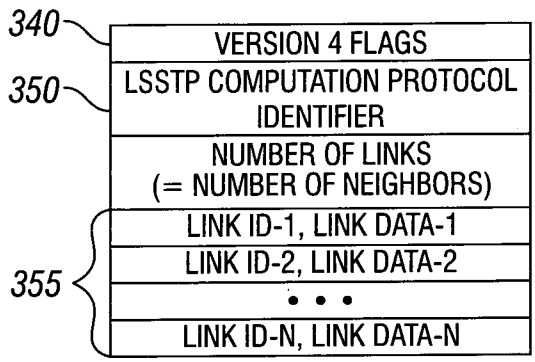

FIG. 10C

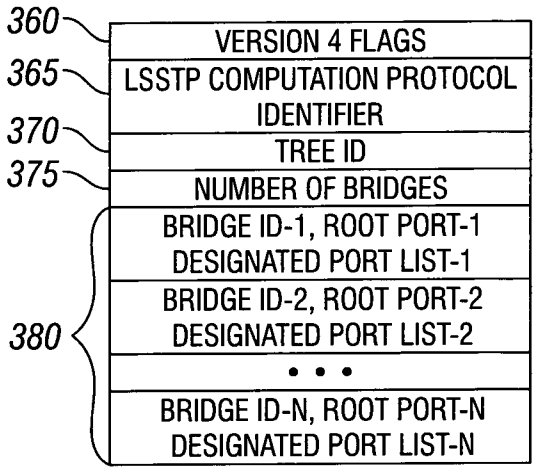

FIG. 11

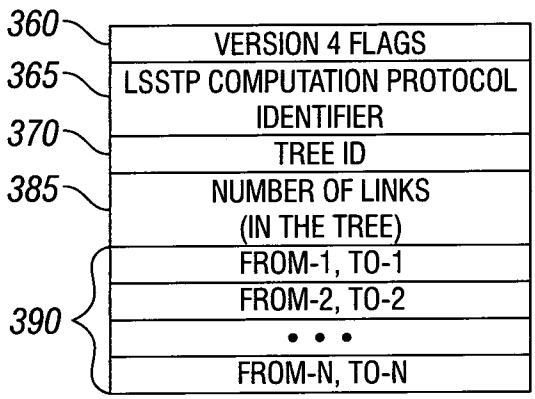

FIG. 12

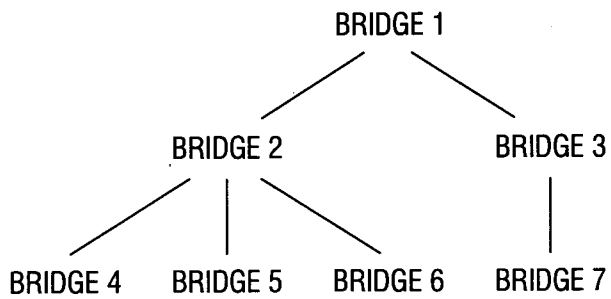

FIG. 13

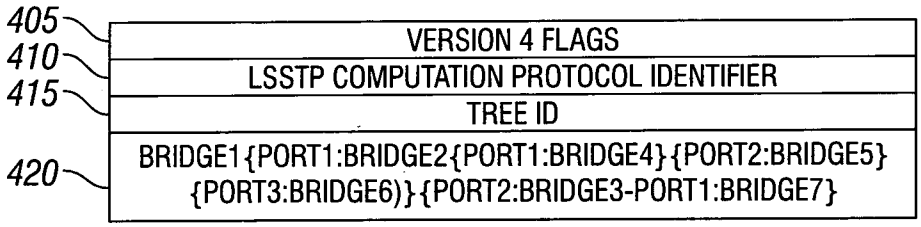

FIG. 14

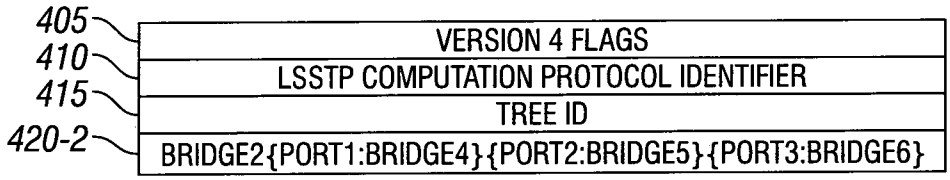

FIG. 15

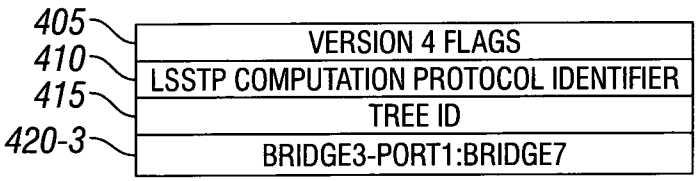

FIG. 16

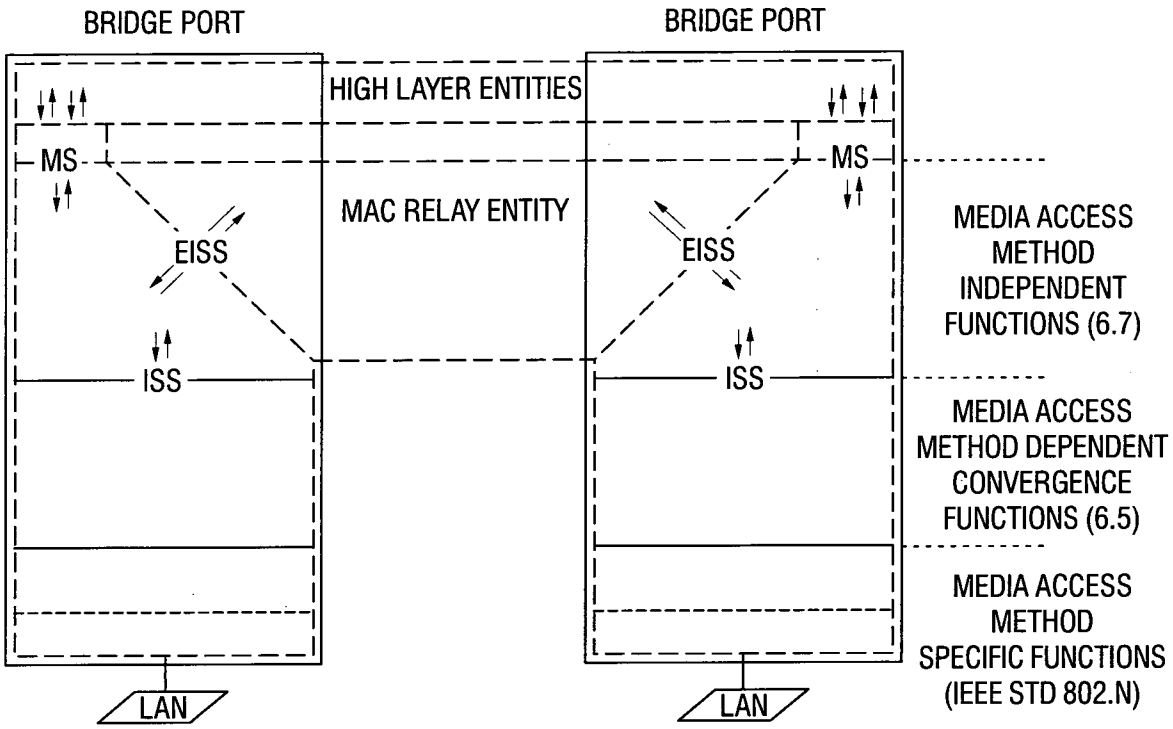
BRIDGE PORT
BRIDGE PORT
HIGH LAYER ENTITIES
MS
MS
MAC RELAY ENTITY
EISS
EISS
ISS
ISS
MEDIA ACCESS METHOD INDEPENDENT FUNCTIONS (6.7)
MEDIA ACCESS METHOD DEPENDENT CONVERGENCE FUNCTIONS (6.5)
MEDIA ACCESS METHOD SPECIFIC FUNCTIONS (IEEE STD 802.N)
LAN
LAN

FIG. 17

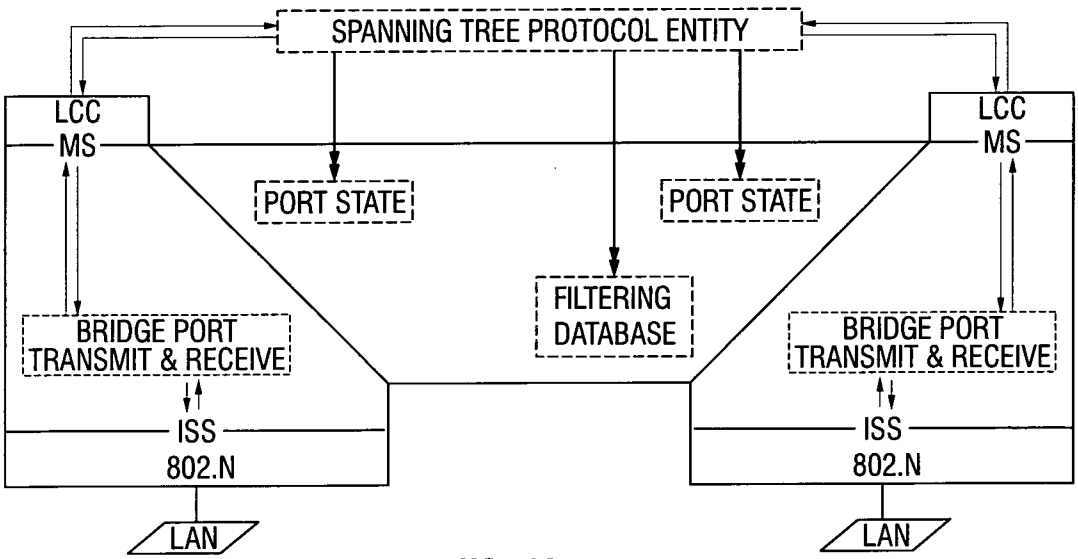
SPANNING TREE PROTOCOL ENTITY
LCC
MS
LCC
MS
PORT STATE
PORT STATE
FILTERING DATABASE
BRIDGE PORT TRANSMIT & RECEIVE
BRIDGE PORT TRANSMIT & RECEIVE
ISS
802.N
ISS
802.N
LAN
LAN

FIG. 18

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2008/000115

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L12/46 H04L12/56

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, PAJ, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 6 578 086 B1 (REGAN JOSEPH [US] ET AL) 10 June 2003 (2003-06-10) column 5, lines 33-37 column 6, line 46 - column 7, line 12 column 4, line 51 - column 5, line 4 column 1, line 61 - column 2, line 3 figure 1	1-25
X	NORMAN FINN: "SHORTEST PATH BRIDGING" INTERNET CITATION, [Online] 22 September 2005 (2005-09-22), pages 1-95, XP002471513 Retrieved from the Internet: URL:http://www.ieee802.org/1/files/public/ docs2005/aq-nfinn-shortest-path 0905.pdf> [retrieved on 2008-03-03] page 82 - page 91	1-25

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
3 June 2008	10/06/2008

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Siebel, Christian

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/IB2008/000115

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 6578086 B1	10-06-2003	NONE	