Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2536497 C 2009/04/14

(11)(21) 2 536 497

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(86) Date de dépôt PCT/PCT Filing Date: 2004/09/30
(87) Date publication PCT/PCT Publication Date: 2005/04/21
(45) Date de délivrance/Issue Date: 2009/04/14
(85) Entrée phase nationale/National Entry: 2006/02/21
(86) N° demande PCT/PCT Application No.: US 2004/032144
(87) N° publication PCT/PCT Publication No.: 2005/036838
(30) Priorité/Priority: 2003/10/02 (US10/677,797)

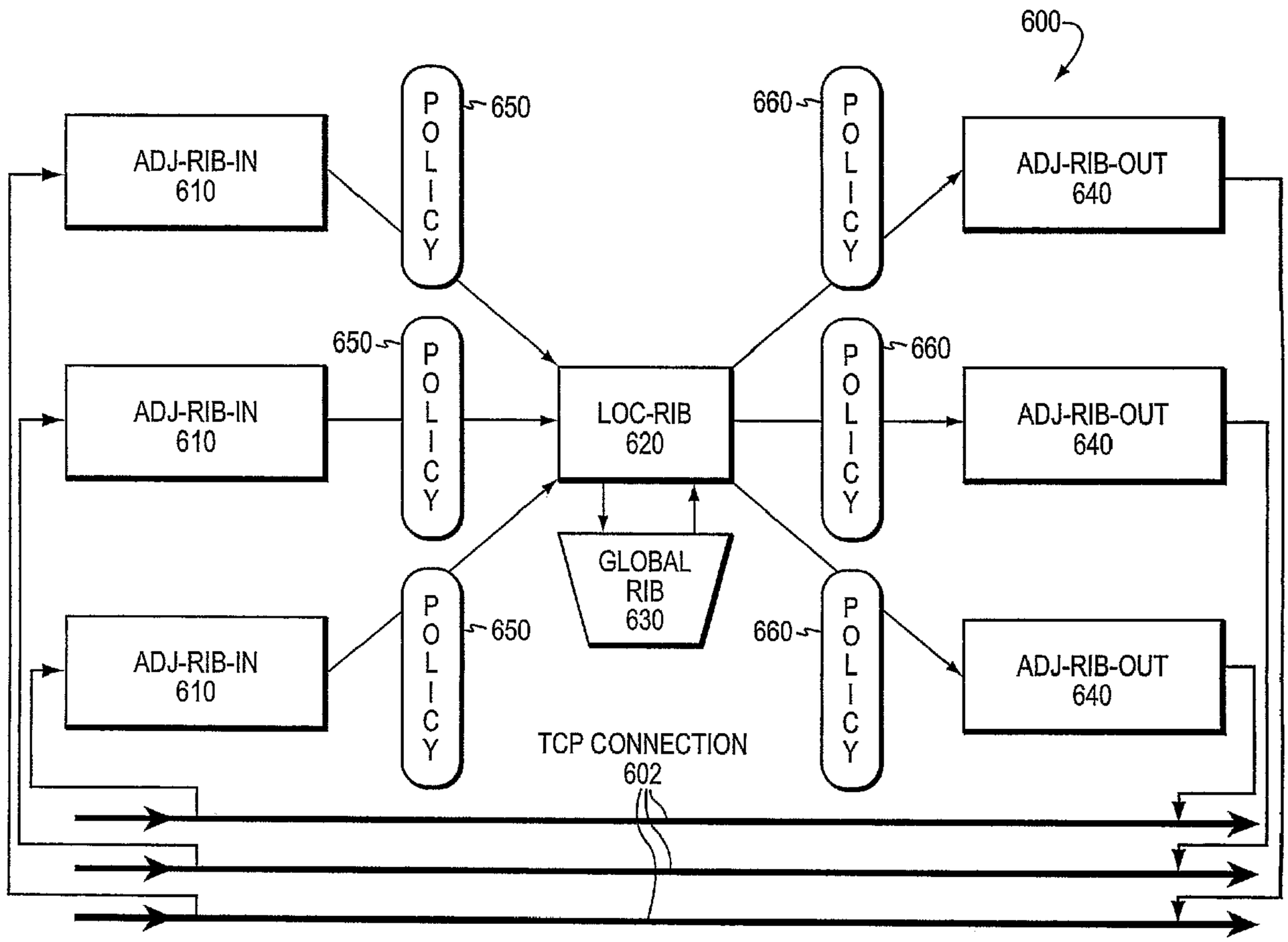
(51) Cl.Int./Int.Cl. *H04L 29/06* (2006.01),
H04L 12/56 (2006.01)

(72) Inventeurs/Inventors:
BALL, DAVID ALEXANDER, GB;
BENNETT, R. ERIC, US;
HESKETH, MARTIN, GB;
SCUDDER, JOHN GALEN, US;
WARD, DAVID D., US

(73) Propriétaire/Owner:
CISCO TECHNOLOGY, INC., US

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : ARCHITECTURE LOGICIELLE REPARTIE POUR L'IMPLEMENTATION DU PROTOCOLE BGP
(54) Title: DISTRIBUTED SOFTWARE ARCHITECTURE FOR IMPLEMENTING THE BORDER GATEWAY PROTOCOL (BGP)



(57) Abrégé/Abstract:
A distributed software architecture implements a routing protocol as a set of processes running on a set of processors of a router. The distributed processes cooperate in a manner that internally exploits the distributed set of processors, yet externally presents an

CA 2536497 C 2009/04/14

(11)(21) 2 536 497

(13) C

(57) Abrégé(suite)/Abstract(continued):
appearance/behavior of a single routing protocol process communicating with its peers in the network. The distributed nature of the architecture is achieved without altering the fundamental routing protocol, but by apportioning certain functions/tasks of the protocol among various processes in the multiprocessor router.

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
21 April 2005 (21.04.2005)

PCT

(10) International Publication Number
WO 2005/036838 A1

(51) International Patent Classification[7]: H04L 12/56

(21) International Application Number:
PCT/US2004/032144

(22) International Filing Date:
30 September 2004 (30.09.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/677,797 2 October 2003 (02.10.2003) US

(71) Applicant *(for all designated States except US)*: **CISCO TECHNOLOGY, INC.** [US/US]; 170 West Tasman Drive, San Jose, CA 95134-1706 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **BALL, David, Alexander** [GB/GB]; 2 Haddon Court, Shakespeare Road, Harpenden Herts. AL5 5NB (GB). **BENNETT, Eric, R.** [US/US]; 2410 Hickman Road, Ann Arbor, MI 48105 (US). **HESKETH, Martin** [GB/GB]; 16 Davys Close, Wheathampstead, St. Albans, Hertfordshire AL4 8TL (GB). **SCUDDER, John, Galen** [US/US]; 900 Spring Street, Ann Arbor, MI 48103 (US). **WARD, David, D.** [US/US]; 301 221st Avenue, Somerset, WI 54025 (US).

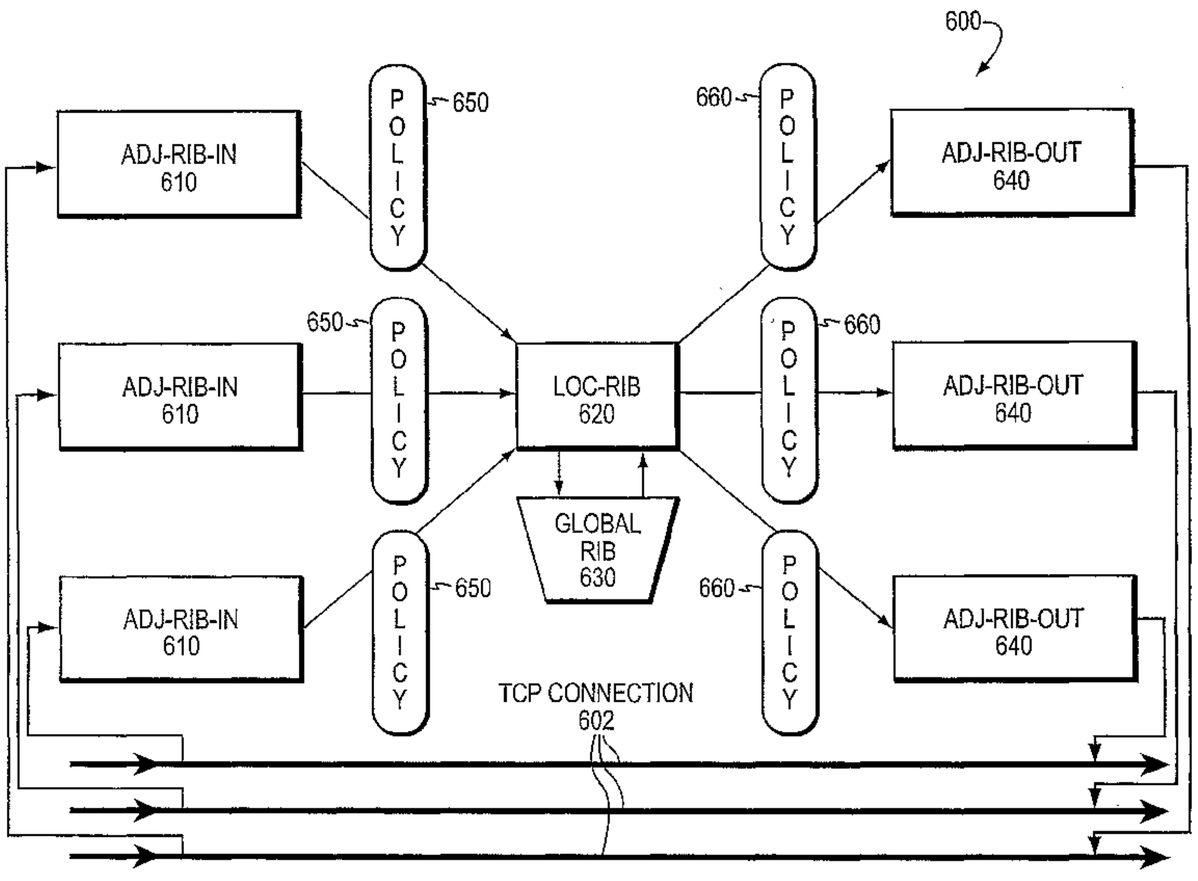
(74) Agents: **BARBAS, Charles, J.** et al.; Cesari and McKenna, LLP, 88 Black Falcon Avenue, Boston, MA 02210 (US).

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

[Continued on next page]

(54) Title: DISTRIBUTED SOFTWARE ARCHITECTURE FOR IMPLEMENTING THE BORDER GATEWAY PROTOCOL (BGP)



(57) Abstract: A distributed software architecture implements a routing protocol as a set of processes running on a set of processors of a router. The distributed processes cooperate in a manner that internally exploits the distributed set of processors, yet externally presents an appearance/behavior of a single routing protocol process communicating with its peers in the network. The distributed nature of the architecture is achieved without altering the fundamental routing protocol, but by apportioning certain functions/tasks of the protocol among various processes in the multiprocessor router.

WO 2005/036838 A1

WO 2005/036838 A1

FR, GB, GR, HU, IE, IT, LU, MC, NL, 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*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

DISTRIBUTED SOFTWARE ARCHITECTURE FOR IMPLEMENTING THE BORDER GATEWAY PROTOCOL (BGP)

DISTRIBUTED SOFTWARE ARCHITECTURE FOR IMPLEMENTING BGP

FIELD OF THE INVENTION

The invention relates generally to routing protocols used in computer networks and, more particularly, to an efficient and scalable implementation of a routing protocol.

BACKGROUND OF THE INVENTION

A computer network is a geographically distributed collection of interconnected communication links used to transport data between nodes, such as computers. Many types of computer networks are available, with the types ranging from local area networks (LANs) to wide area networks (WANs). The nodes typically communicate by exchanging discrete packets or messages of data according to pre-defined protocols. In this context, a protocol consists of a set of rules defining how the nodes interact with each other.

Computer networks may be further interconnected by an intermediate node, such as a router, to extend the effective "size" of each network. Since management of a large system of interconnected computer networks can prove burdensome, smaller groups of computer networks may be maintained as routing domains or autonomous systems. The networks within an autonomous system are typically coupled together by conventional "intradomain" routers. Yet it still may be desirable to increase the number of nodes capable of exchanging data; in this case, *interdomain* routers executing interdomain routing protocols are used to interconnect nodes of the various autonomous systems.

An example of an interdomain routing protocol is the Border Gateway Protocol version 4 (BGP), which performs routing between autonomous systems by exchanging routing and reachability information among neighboring interdomain routers of the systems. An adjacency is a relationship formed between selected neighboring (peer) routers for the purpose of exchanging routing information messages and abstracting the network topology. Before transmitting such messages, however, the peers cooperate to

establish a logical "peer" connection (session) between the routers. BGP generally operates over a reliable transport protocol, such as the Transmission Control Protocol (TCP), to establish a TCP connection/session.

The routing information exchanged by BGP peer routers typically includes destination address prefixes, i.e., the portions of destination addresses used by the routing protocol to render routing ("next hop") decisions. Examples of such destination addresses include Internet Protocol (IP) version 4 (IPv4) and version 6 (IPv6) addresses. The BGP routing protocol is well known and described in detail in *Request For Comments (RFC) 1771*, by Y. Rekhter and T. Li (1995), *Internet Draft <draft-ietf-idr-bgp4-20.txt>* titled, *A Border Gateway Protocol 4 (BGP-4)* by Y. Rekhter and T. Li (April 2003) and *Interconnections, Bridges and Routers*, by R. Perlman, published by Addison Wesley Publishing Company, at pages 323-329 (1992).

The interdomain routers configured to execute an implementation of the BGP protocol, referred to herein as *BGP routers*, perform various routing functions, including transmitting and receiving routing messages and rendering routing decisions based on routing metrics. Each BGP router maintains a routing table that lists all feasible paths to a particular network. Periodic refreshing of the routing table is generally not performed; however, BGP peer routers residing in the autonomous systems exchange routing information under certain circumstances. For example, when a BGP router initially connects to the network, the peer routers exchange the entire contents of their routing tables. Thereafter when changes occur to those contents, the routers exchange only those portions of their routing tables that change in order to update their peers' tables. These *update messages* are thus incremental update messages sent in response to changes to the contents of the routing tables and advertise only a best path to a particular network.

Broadly stated, a BGP router generates routing update messages for an adjacency or peer router by "walking-through" the routing table and applying appropriate routing policies. A routing policy is information that enables a BGP router to rank routes according to filtering and preference (i.e., the "best route"). Routing updates provided by the update messages allow BGP routers of the autonomous systems to construct a con-

struct a consistent view of the network topology. The update messages are typically sent using a reliable transport, such as TCP, to ensure reliable delivery. TCP is a transport protocol implemented by a transport layer of the IP architecture; the term *TCP/IP* is commonly used to denote this architecture. The TCP/IP architecture is well known and described in *Computer Networks, 3rd Edition*, by Andrew S. Tanenbaum, published by Prentice-Hall (1996).

A common implementation of the BGP protocol is embodied as a single process executing on a single processor, e.g., a central processing unit (CPU), of the BGP router, while another known implementation provides multiple instances of the BGP process running on a single CPU. In this latter implementation, each BGP instance has its own routing table and chooses its own best path for a given prefix. From the perspective of the protocol, each BGP instance is a separate router; yet, each router instance shares the same resources, e.g., the single CPU. Both BGP implementations store and process update messages received from their peer routers, and create and process update messages for transmission (advertisement) to those peers. However, the amount of processing time (i.e., bandwidth) available on the single CPU is finite which, in turn, results in limitations on the number of routes the BGP implementations can handle and limitations on the number of peers/adjacencies that the BGP implementations can support.

Examples of factors that limit the number of adjacencies and routes that a BGP implementation can support include the physical amount of memory in the BGP router. A router typically employs a 32-bit CPU that enables support of, at most, 4Gigabytes (GB) of memory. The amount of memory the BGP router can support is important because secondary storage, such as disks, cannot be efficiently used to store update messages given the substantial read/write latencies involved with accessing the disks. Moreover, each adjacency maintained by the router has a certain minimum CPU cost associated therewith. Examples of such cost include sending "KeepAlive" messages at predetermined intervals, processing received update messages, and deciding whether to send update messages to peers whenever a change is made to the routing table.

In general, it is desirable to increase the number of peers a BGP implementation can support. Yet as the number of peers increases, the amount and quantity of process-

ing increases correspondingly. In addition, convergence time increases as the number of routing peers increases. As used herein, the convergence time is the time needed for a BGP router to receive and process update messages from all its routing peers, select best paths for each prefix included in those messages, install those best paths into the routing table and advertise those best paths back to its routing peers via update messages. As a result, CPU, memory and even communication scaling becomes an issue with the BGP implementation.

One solution to the scaling issue is to provide a BGP implementation that spans a plurality of routers, wherein each router includes a processor that maintains a subset of the supported peers. Such a multi-processor implementation has a fundamental limitation that, from the point of view of a peer, each processor resembles a separate router. This results in a cognitive and operational model wherein the multiple routers interact separately instead of functioning as a single router to the network. The multiple-router model is operationally more complex than a single router; that is, it is easier and more cost effective, from an operational cost point of view, to operate a single router than it is to configure a plurality of routers to interoperate.

Accordingly, there is a need to provide additional CPU bandwidth to a BGP implementation that enables scaling to support larger numbers of peers and routes, without incurring similar increases in convergence time. The present invention is directed to an architecture that enables scaling of a BGP implementation to allow support of such additional peers and routes.

SUMMARY OF THE INVENTION

The present invention overcomes the disadvantages of the prior art by providing a distributed software architecture that implements a routing protocol as a set of processes running on a set of processors of a router. The distributed processes cooperate in a manner that internally exploits the distributed set of processors, yet externally presents an appearance/behavior of a single routing protocol process communicating with its peers in the network. The distributed nature of the architecture is achieved without altering the fundamental routing protocol, but by apportioning certain functions/tasks of the protocol among various processes in the multiprocessor router.

In the illustrative embodiment, the routing protocol is the Border Gateway Protocol version 4 (BGP). A BGP implementation of the distributed software architecture comprises multiple processes including BGP speakers, each of which is responsible for managing a set of routing peers, and a BGP Routing Information Base ("bRIB"). The BGP speakers are responsible for the majority of processing costs in the BGP implementation. The use of multiple BGP speakers provides a substantial scaling feature of the invention by enabling cost effective processing of tasks, such as packet reception, packet transmission and packet formatting.

Each BGP speaker preferably executes on a different processor and is generally responsible for, among other things, handling (terminating) one or more BGP peering connections, receiving and storing routes from each peer, and applying inbound policy to the routes received from each peer. Each BGP speaker is also responsible for downloading all routes received from its peers (except those "filtered" by policy) to the bRIB, which preferably executes on a processor different from that executing a speaker. The bRIB performs a first stage of route selection to compute a set of best routes from among the routes downloaded from all of the BGP speakers of the router and, thereafter, downloads each route selected as the best route to another process, i.e., the global RIB, which performs a second (and final) stage of route selection. The bRIB also sends the computed best routes to each BGP speaker, which applies outbound policy (per peer) to those routes prior to sending them to the peers.

Advantageously, the inventive architecture allows the workload of the distributed software implementation to be apportioned among multiple processes, effecting a more scalable BGP implementation capable of allowing a user the ability to dedicate resources to particular groups of peers, while maintaining the external appearance of a single BGP protocol instance. As noted, the BGP implementation may be further apportioned among several processors in a multiprocessor router, such that the total required processing is distributed among the processors, instead of concentrated on a single processor. As the number of routing peers increases, additional processors can be added to the router to handle the extra processing required, thereby avoiding overloading of a single processor and, hence, adversely affecting the convergence time of the protocol.

A secondary advantage of the invention is improved fault-tolerance. If a particular processor running a BGP speaker in the router fails, only the routing peers assigned to that speaker are affected. If the failing processor is running the bRIB process, no peers are affected and the router can recover simply by having each speaker resend all of its paths to the bRIB when it restarts. In the absence of the inventive distributed architecture, a failure to the processor running the concentrated BGP implementation would affect all peers of that implementation.

A tertiary advantage of the invention is that groups of peers can be co-located on given processors, separate from peers on the other processors, to effect feature separation or resource isolation. Furthermore, the inventive architecture maintains the autonomy of the peers, such that each peer can be configured ("placed") in a speaker arbitrarily, with the actual placement policy being determined by the user. For example, the user could place all peers exchanging routes for IPv4 on one processor, while peers exchanging routes for IPv6 could be placed on a different processor. Churn in the topology of a network will only slightly impact another network, effectively isolating the delivery of each service from perturbations in the churned network.

BRIEF DESCRIPTION OF THE DRAWINGS

The above and further advantages of the invention may be better understood by referring to the following description in conjunction with the accompanying drawings in which like reference numbers indicate identical or functionally similar elements:

Fig. 1 is a schematic block diagram of a computer network comprising a plurality of autonomous systems interconnected by intermediate nodes, such as Border Gateway Protocol (BGP) interdomain routers;

Fig. 2 is a schematic block diagram of an embodiment of an interdomain router that may be advantageously used with the present invention;

Fig. 3 is a schematic block diagram of a conventional protocol stack, such as the Internet communications protocol stack, within the interdomain router of Fig. 2;

Fig. 4 is a schematic block diagram of an update message, such as a Border Gateway Protocol (BGP) update message that may be advantageously used with the present invention;

Fig. 5 is a schematic block diagram of a path attributes field of the BGP update message that may be advantageously used with the present invention;

Fig. 6 is a schematic block diagram illustrating the architecture of the BGP protocol;

Fig. 7 is a schematic block diagram illustrating a BGP implementation of a distributed software architecture according to the present invention;

Fig. 8 is a schematic block diagram of a routing table having a plurality of routing table entries; and

Fig. 9 is a flowchart illustrating a sequence of steps pertaining to data flow through the BGP implementation of the distributed software architecture according to the present invention.

DETAILED DESCRIPTION OF AN ILLUSTRATIVE EMBODIMENT

Fig. 1 is a schematic block diagram of a computer network 100 comprising a plurality of routing domains or autonomous systems interconnected by intermediate nodes, such as conventional intradomain routers 120 and interdomain routers 200. The autonomous systems may include various routing domains (AS_{1-4}) interconnected by the interdomain routers. The interdomain routers 200 are further interconnected by shared medium networks, such as local area networks (LANs) 104, and point-to-point links 102, such as frame relay links, asynchronous transfer mode links or other serial links. Communication among the routers is typically effected by exchanging discrete data packets or messages in accordance with pre-defined protocols, such as the Transmission Control Protocol/Internet Protocol (TCP/IP). It will be understood to those skilled in the art that other protocols, such as the Internet Packet Exchange (IPX) protocol, may be advantageously used with the present invention.

Fig. 2 is a schematic block diagram of an interdomain router 200 that may be advantageously used with the present invention. The interdomain router 200 comprises a plurality of loosely coupled processors 210 connected to a plurality of ingress and egress line cards (line cards 260) via a high-speed switch fabric 250 such as, e.g., a crossbar interconnection or high-speed bus. Those skilled in the art will recognize that other router platforms such as, e.g., a plurality of independent nodes interconnected as a

multi-node cluster, could be used in accordance with the invention. In this context, the term "node" denotes a chassis adapted to hold a plurality of modules, including processors and line cards.

The processors 210 are illustratively route processors (RPs), each having a dedicated memory 230. The memory 230 may comprise storage locations addressable by the processor for storing software programs and data structures associated with the inventive distributed routing protocol architecture. The processor 210 may comprise processing elements or logic for executing the software programs and manipulating the data structures. A router operating system 232, portions of which are typically resident in memory 230 and executed by the processor, functionally organizes the router by, *inter alia*, invoking network operations in support of software processes (described herein) executing on the processor. It will be apparent to those skilled in the art that other processor and memory means, including various computer readable media, may be used for storing and executing program instructions pertaining to the inventive architecture described herein.

In the illustrative embodiment, each RP 210 comprises two central processing units (CPUs 220), e.g., Power-PC 7460 chips, configured as a symmetric multiprocessing (SMP) pair. The CPU SMP pair is adapted to run a single copy of the router operating system 232 and access its memory space 230. As noted, each RP has a memory space that is separate from the other RPs in the router 200. The processors communicate using an interprocess communication (IPC) mechanism. In addition, each line card 260 comprises an interface 270 having a plurality of ports coupled to a receive forwarding processor (FP Rx 280) and a transmit forwarding processor (FP Tx 290). The FP Rx 280 renders a forwarding decision for each packet received at the router on interface 270 of an ingress line card in order to determine to which RP 210 to forward the packet. To that end, the FP Rx renders the forwarding decision using an internal forwarding information base, IFIB, of a FIB 275. Likewise, the FP Tx 290 performs lookup operations (using FIB 275) on a packet transmitted from the router via interface 270 of an egress line card.

A key function of the interdomain router 200 is determining the next node to which a packet is sent; in order to accomplish such "routing" the interdomain routers

cooperate to determine best paths through the computer network 100. The routing function is preferably performed by an internetwork layer of a conventional protocol stack within each router. Fig. 3 is a schematic block diagram of a conventional network protocol stack, such as the Internet communications protocol stack 300. The architecture of the Internet protocol stack is represented by 4 layers termed, in ascending interfacing order, the network interface layer 308, the internetwork layer 306, the transport layer 304 and the application layer 302.

The lower network interface layer 308 is generally standardized and implemented in hardware and firmware, whereas the higher layers are typically implemented in the form of software. The primary internetwork layer protocol of the Internet architecture is the IP protocol. IP is primarily a connectionless protocol that provides for internetwork routing, fragmentation and reassembly of exchanged packets - generally referred to as "datagrams" in an Internet environment - and which relies on transport protocols for end-to-end reliability. An example of such a transport protocol is the TCP protocol, which is implemented by the transport layer 304 and provides connection-oriented services to the upper layer protocols of the Internet architecture. The term *TCP/IP* is commonly used to denote the Internet architecture.

In particular, the internetwork layer 306 concerns the protocol and algorithms that interdomain routers utilize so that they can cooperate to calculate paths through the computer network 100. An interdomain routing protocol, such as the Border Gateway Protocol version 4 (BGP), is used to perform interdomain routing (for the internetwork layer) through the computer network. The interdomain routers 200 (hereinafter "peer routers") exchange routing and reachability information among the autonomous systems over a reliable transport layer connection, such as TCP. An adjacency is a relationship formed between selected peer routers for the purpose of exchanging routing messages and abstracting the network topology. The BGP protocol uses the TCP transport layer 304 to ensure reliable communication of routing messages among the peer routers.

In order to perform routing operations in accordance with the BGP protocol, each interdomain router 200 maintains a routing table 800 that lists all feasible paths to a particular network. The routers further exchange routing information using routing

update messages 400 when their routing tables change. The routing update messages are generated by an updating router to advertise best paths to each of its neighboring peer routers throughout the computer network. These routing updates allow the BGP routers of the autonomous systems to construct a consistent and up-to-date view of the network topology.

Fig. 4 is a schematic block diagram of a conventional BGP update message 400 comprising a plurality of fields appended to a header 410. An unfeasible routes length field 402 indicates the total length of a withdrawn routes field 404, which illustratively contains a list of IP address prefixes for the routes being withdrawn from service. A total path attribute length field 406 indicates the total length of a path attributes field 500 and a network layer reachability information field 408 illustratively contains a list of IP (IPv4 or IPv6) address prefixes. Note that the combination of a set of path attributes and a prefix is referred to as a "route"; the terms "route" and "path" may be used interchangeably herein. The format and function of the update message 400 is described in *RFC 1771* and *Interconnections, Bridges and Routers*.

Specifically, the path attributes field 500 comprises a sequence of fields, each describing a path attribute in the form of a triple (i.e., attribute type, attribute length, attribute value). Fig. 5 is a schematic block diagram of the path attributes field 500 comprising a plurality of subfields including a flags subfield 502, an attribute type subfield 504, an attribute length subfield 506 and an attribute value subfield 508. In particular, the attribute type subfield 504 specifies a plurality of attribute type codes, examples of which include an autonomous system (AS) path, a multi-exit discriminator (MED) code and a communities attribute, which is a set of opaque 32-bit tags that can apply to a route. The MED is an optional non-transitive attribute having a value that may be used by an updating BGP router's decision algorithm to discriminate among multiple exit points to a neighboring autonomous system, as described further herein. Note that the path attributes are derived from a combination of configuration and protocol (i.e., propagated from the BGP protocol) information.

BGP Architecture

Fig. 6 is a schematic block diagram illustrating the architecture of the BGP protocol. Peers announce routing updates via TCP connections 602. The BGP protocol "listens" for routing update messages 400 and stores all learned routes for each connection in a BGP database. The BGP database is illustratively organized as Adjacency RIB In (Adj-RIB-In 610), Adjacency RIB Out (Adj-RIB-Out 640) and local RIB (loc-RIB 620). Each peer/TCP connection 602 is associated with an Adj-RIB-In 610 and an Adj-RIB-Out 640. Note that this association is a conceptual data construct; there is typically not a separate Adj-RIB-In/-Out database for each peer.

The BGP protocol runs inbound policy on all routes "learned" for each connection 602 and those routes that match are stored in an Adj-RIB-In 610 unique to that connection. Additional inbound policy 650 (filtering) is then applied to those stored routes, with a potentially modified route being installed in the loc-RIB 620. The loc-RIB 620 is generally responsible for selecting the best route per prefix from the union of all policy-modified Adj-RIB-In routes, resulting in routes referred to as "best paths". The set of best paths is then installed in the global RIB 630, where they may contend with routes from other protocols to become the "optimal" path ultimately selected for forwarding. Thereafter, the set of best paths have outbound policy 660 run on them, the result of which is placed in appropriate Adj-RIB-Outs 640 and announced to the respective peers via the same TCP connections 602 from which routing update messages 400 were learned.

Many of the functions or tasks performed within the BGP protocol are performed on distinct subsets of routing data, independently from one another. These tasks include (1) tracking the state of each peer according to the BGP Finite State Machine (FSM), described in *draft-ietf-idr-bgp4-20.txt (Section 8)*, and responding to FSM events, (2) parsing update messages 400 received from each peer and placing them in an Adj-RIB-In 610 for that peer *(Section 3)*, and (3) applying inbound policy 650 for the peer to filter or modify the received updates in the Adj-RIB-In. The BGP implementation also (4) calculates the best path for each prefix in the set of Adj-RIB-Ins and places those best paths in the loc-RIB 620 *(Section 9)*. As the number of peers increases, the number of paths per-prefix also increases and, hence, this calculation becomes more complex. Additional tasks performed by the BGP implementation include

(5) applying outbound policy 660 for each peer on all the selected paths in the loc-RIB to filter or modify those paths, and placing the filtered and modified paths in an Adj-RIB-Out 640 for that peer, as well as (6) formatting and sending update messages 400 to each peer based on the routes in the Adj-RIB-Out for that peer.

Tasks (1), (2), and (3) are defined per peer and operate on routing data learned only from that peer. Performing any of these tasks for a given peer is done independently of performing the same task for any other peers. Task (4) examines all paths from all peers, in order to insert them into the loc-RIB and determine the best path for each prefix. Tasks (5) and (6), like tasks (1), (2) and (3), are defined per peer. While both tasks (5) and (6) must access the set of best paths determined in task (4), they generate routing data for each peer independently of all of the other peers. Thus, the autonomy of each subset of the data and the tasks performed on them lend themselves to distribution across processes or threads in an *n*-way SMP router, or across nodes in a cluster, so long as each task has access to the required data. The required data includes (i) inbound routes from the peer for tasks (1), (2) and (3); (ii) all paths in all the Adj-RIBs-Ins for task (4); and (iii) a set of best paths for tasks (5) and (6).

According to the present invention, a distributed software architecture is provided that implements a routing protocol, such as BGP, as a set of processes running on a set of processors of a router. The distributed processes cooperate in a manner that internally exploits the distributed set of processors, yet externally presents an appearance/behavior of a single routing protocol process communicating with its peers in the network. The distributed nature of the architecture is achieved without altering the fundamental BGP routing protocol, but by apportioning certain functions/tasks of the protocol among various processes in the multiprocessor router.

BGP Implementation of Distributed Software Architecture

Fig. 7 is a schematic block diagram illustrating a BGP implementation 700 of the distributed software architecture according to the present invention. The distributed BGP implementation comprises multiple processes including one or more BGP speaker processes 710, each of which is responsible for managing a set of routing peers, and a BGP Routing Information Base ("bRIB") process 720. The BGP speakers 710 are re-

sponsible for the majority of processing costs in the BGP implementation. The use of multiple BGP speakers provides a substantial scaling feature of the invention by enabling cost effective processing of tasks, such as packet reception, packet transmission and packet formatting. Each BGP speaker is generally responsible for, among other things, handling (terminating) one or more BGP peering connections, receiving and storing routes from each peer, and applying inbound policy to the routes received from each peer.

Specifically, each BGP speaker (i) establishes and maintains a reliable TCP connection to each routing peer and handles FSM events for the peer, (ii) receives and processes update messages 400 received from the peers, places the paths in the Adj-RIB-In 610 and applies inbound policy 650, (iii) sends all paths in the Adj-RIBs-In 650 to the bRIB 720, and (iv) receives a best path for each prefix from the bRIB 720 and advertises it to each routing peer after applying outbound policy 660 for that peer. In the distributed software architecture, policy computations are preferably handled by a separate software component, e.g., a library, to which the BGP speaker "binds", although these computations could alternately be implemented "in-line" as part of the BGP code. Each BGP speaker 710 is illustratively a multithreaded process; policy is thus preferably handled as a library function call initiated by one of the BGP speaker threads. As such, policy computations occur within the BGP process space.

Policy may be used to limit the reception or distribution of routing information from and to a BGP speaker (i.e., a form of access control or filtering) and to manipulate the data in the routing information. Examples of routing policy include "match if prefix is 10/8" or "match if prefix starts with 192.168 and AS path starts with 690". One or both of these policies may be applied to filtering on a peering session in an inbound fashion, such that the BGP speaker only accepts those routes that match the policy. Policy can also apply to filtering in an outbound fashion, such that only routes that match one of the policies are sent to the peers. Moreover, policy may be used for "go or no-go" decisions on whether to pass a route and to manipulate the route. For example, assume a policy "if the route contains AS number 1800, then add community 42 to the route". This manipulates the data comprising the route according to the policy control.

Several processors 210 may be used to run the speakers 710, wherein each processor runs entirely independently of the other processors. The reason for distributing functions, such as policy, to the BGP speaker 710 as opposed to handling it in the bRIB 720 is that executing the policy code is one of the most expensive operations in the entire BGP protocol. As noted, there is only one bRIB 720 in the distributed software architecture, but potentially many speakers 710. By distributing the policy code function/task to the speakers, that task can be apportioned into many smaller subtasks and the collective strength of the multiple processors may be applied to execute the code. In addition, each BGP speaker is illustratively assigned many routing peers (e.g., 1000) to manage and every routing peer configured on the router is assigned to one speaker. Therefore, as the number of BGP routing peers increases, extra processors can be added to the router to handle the extra processing needed.

Each BGP speaker 710 is responsible for downloading all routes received from its peers (except those "filtered" by policy) to the bRIB 720, as described further herein. The bRIB is illustratively a process executing on a processor (RP 210) of the BGP router 200 that may be separate from those processors functioning as speakers; alternatively, the bRIB may share a processor with one of the speakers. It will be understood to those of skill in the art that other implementations are contemplated by the invention, including implementations wherein more than two (or all) processes of the distributed BGP architecture run on the same processor.

The bRIB process 720 (i) receives and stores routes received from each speaker process, (ii) performs a first stage of route selection to compute a set of best routes from among the routes (prefixes) downloaded from all of the BGP speakers, (iii) installs the best routes/paths into a "local" routing table (i.e., loc-RIB 620) and (iv) sends the computed best paths back to all the speakers 710 so that they can be advertised to their routing peers. It should be noted that the speakers must not announce the routes they learn from the bRIB back to the bRIB. Moreover, since all paths in all Adj-RIBs-Ins 610 are sent to the bRIB 720, the correct best path for each network is selected by the bRIB, according to the BGP protocol standard.

The global RIB 730 illustratively maintains a "system" routing table for the router. The system routing table ("routing table 800") is a database that contains rout-

ing information used to construct a forwarding table of the FIB 275 used by the FPs of the router 200 when performing forwarding decisions on packets. The routing table 800 typically denotes a database containing all available routes, including ones that have been selected for forwarding (optimal paths) as well as backup routes that are not currently selected for forwarding, while the forwarding table denotes those optimal best paths that have actually been selected for forwarding.

The loc-RIB 620 denotes a table storing routes that are similar to the routes in the forwarding table. The bRIB 720 maintains the loc-RIB 620, including processing and downloading to the global RIB 730 each route/path in the loc-RIB selected as the best path. The global RIB 730 maintains a copy of those downloaded best paths, along with other paths/routes downloaded from other routing protocols, in order to compute a set of optimal best paths/routes for installation in the routing table 800. The global RIB 730 preferably interacts with another software component to download those optimal routes to all the line cards 260 of the router 200, each of which maintains its own copy as a forwarding table.

Fig. 8 is a schematic block diagram of a routing table 800 comprising a plurality of route entries 810, each of which includes a plurality of fields. Specifically, route entry 810 includes a network field 812 containing a network portion of an IP address identifying a network, a mask/length field 814 containing a mask for differentiating between the network portion of the IP address and a host portion, and an entry version number field 816 containing a version number of the entry. A path field 820 contains one or more paths, wherein each path describes the "next hop" address or interface 270 of the peer or other path attributes 500 of routes in the computer network, while an optimal path field 818 contains the optimal best path from among the paths described in field 820 based on pre-specified route selection criteria.

The routing table 800 further includes a table version number 830 that is used to indicate a version (level) of the routing table. The table version number 830 is incremented each time there is a change to the routing table 800. The entry version number 816 is used for incremental update operations. Thus, each time there is a change to an entry 810, such as when the entry is added or deleted or when there is a best path

change, the table version number 830 is incremented and the entry version number 816 is set to that incremented value.

In the illustrative embodiment, the distributed BGP software architecture is organized such that each BGP speaker process 710 executes on a different RP. In addition, the bRIB process 720 executes on an RP 210 separate from an RP executing a BGP speaker 710, to thereby avoid contention between the bRIB and speaker for similar resources. Illustratively, the bRIB 720 executes on the same RP 210 as the global RIB 730, but this is not a requirement and those processes could execute on different RPs. However, when configuring the bRIB 720 to execute on the same RP as the global RIB 730, the performance of the router increases because the processes communicate, e.g., with respect to route selection, via message exchanges that occur faster on the same RP 210 rather than across the switch fabric 250. It will be understood to those skilled in the art that alternative configurations are contemplated, including allowing all processes to run on the same RP 210, as well as allowing the bRIB and global RIB to be the same process.

As noted, the BGP processes of the distributed software architecture cooperate in a manner that externally presents an appearance/behavior of a single routing protocol process despite having those processes run on various RPs 210 of the router. To make the distributed RPs appear as a single-processor BGP, a local packet transport service is used to distribute TCP sessions to the RPs, even TCP sessions with identical destination IP addresses. Thus, from the perspective of an "outsider", all RPs share the same IP address or addresses. This is different from the typical way of dealing with a collection of processors/routers, where each would have its own unique IP address. An example of a local packet transport service that may be advantageously used with the present invention is described in U.S. Patent Application Serial No. 10/293,180, titled *System and Method for Local Packet Transport Services within Distributed Routers*, filed on November 12, 2002, which application is hereby incorporated by reference as though fully set forth herein.

Route Selection

Route selection, as described herein, utilizes a distance vector (Bellman-Ford) algorithm or, more specifically, a BGP best path selection (path vector) algorithm. According to the BGP standard, every BGP router announces to all of its peers the routes it uses for its own forwarding. As a result of these announcements, a particular router may gather from its peers two or more routes for some networks. For example, the router may have learned two or more different ways to reach network 10.1.1.0/24; the best path selection computation is a way of choosing one of those routes as "best" and using it to render forwarding decisions for the router. Note that in the case of multi-path BGP, more than one path may be chosen as best by the algorithm. However, it should be further noted that these multiple chosen paths are only downloaded to the global RIB and there is only ever one best path for each prefix sent back to the speakers.

Broadly stated, the illustrative BGP best path selection algorithm comprises the following steps:

1. Prefer the path with the largest WEIGHT; note that WEIGHT is a locally specified parameter, i.e., local to the router on which it is configured;

2. Prefer the path with the largest LOCAL_PREF;

3. Prefer the path that was locally originated via a network or aggregate BGP subcommand, or through redistribution from an interior gateway protocol (IGP);

4. Prefer the path with the shortest AS_PATH;

5. Prefer the path with the lowest origin type, e.g., IGP is lower than exterior gateway protocol (EGP), and EGP is lower than INCOMPLETE;

6. Prefer the path with the lowest MED among routes with identical AS;

7. Prefer external (eBGP) over internal (iBGP) paths;

8. Prefer the path with the lowest IGP metric to the BGP next hop;

9. Prefer the route coming from the BGP router with the lowest router ID (BGP identifier);

10. If the originator or router ID is the same for multiple paths, prefer the path with the minimum cluster ID length; and

11. Prefer the path coming from the lowest neighbor (peer) address.

It should be noted that the full best path computation is preferably performed where the router has fast access to all paths for a given prefix; thus, in the illustrative embodiment, the full BGP best path selection algorithm is performed in the bRIB 720. The loc-RIB 620 conceptually comprises the output of the BGP selection algorithm, i.e., the bRIB 720 and loc-RIB 620 are not quite identical. The bRIB 720 contains all routes downloaded by the speakers that are considered for selection into the loc-RIB 610; the bRIB then performs the first stage of route selection.

Once the bRIB computes the loc-RIB 620, the next function in the route selection procedure is to generate the forwarding tables of FIB 275 for the line cards 260. The bRIB abstracts the best paths/routes of the loc-RIB and downloads them to the global RIB 730. Since there are protocols other than BGP running on the router 200, the global RIB gathers abstracted routes from other routing protocols, e.g., OSPF and IS-IS routes, as well as locally configured routes and static routes, and performs a second (and final) stage of route selection to compute a set of optimal best paths for all routing protocols executing on the router. For example, the global RIB 730 examines a BGP best path/route and determines whether it is the only route for a particular destination; if so, the global RIB selects that route as an optimal best path. However, if there are final best paths to a destination offered from both BGP and, e.g., OSPF, (a "conflict") the global RIB must select one.

Specifically, the global RIB 730 selects optimal best paths from among various protocols where there may be conflicts between the outputs of the different protocols. By examining the route selection outputs from the different protocols, the global RIB 730 is the final arbiter of which routes get selected as optimal paths to destinations. Routes with different destinations are never in conflict, so the problem arises when there are two or more routes that have the same destination. For example, assume there is a route from OSPF for 10/8 and a route from BGP for 10/8; the global RIB must then select one for installation in the routing table 800. The criteria that the global RIB 730

may apply to determine which route to install may be, e.g., always use OSPF over BGP. Once the global RIB has rendered its conflict resolution, it essentially selects routes for installation in the FIB. Other software components in the router then download the routes from the global RIB into the FIB 275 of the line cards 260.

When generating update messages 400 to send to its peers, each BGP speaker 710 may apply policy configured for redistribution of routes from other protocols into BGP; redistribution of routes occurs by the global RIB 730 uploading (communicating) those optimal best paths into the bRIB 720. For example, redistribution may occur from OSPF into BGP, which means all active OSPF optimal best paths (those that have made it into the global RIB) are copied into the BGP routing table (i.e., the loc-RIB 620). These redistributed protocol routes do not supersede those routes in the loc-RIB, but rather augment them to essentially factor into the BGP best path selection algorithm. Note that the best paths in the loc-RIB that have been downloaded to the global RIB are not thereafter uploaded back to the bRIB. Moreover, if a redistributed path is selected as the best path by the bRIB and installed into the loc-RIB 620, it is not then downloaded to the global RIB (since that is where it came from originally).

The bRIB 720 transmits a copy of the loc-RIB 620 to each BGP speaker 710, which performs outbound policy operations on those loc-RIB best paths/routes. As a result of the policy operations, the speaker computes a subset of routes for the Adj-RIB-Out 640 for a peer router. The BGP speaker then creates one or more BGP update messages 400 based on internal data representations of the routes in the Adj-RIB-Out 640 and transmits those update messages to the peer. As noted, the BGP protocol is an incremental protocol in that the update messages are incremental. Despite having an Adj-RIB-Out 640 with many (e.g., a million) routes, only routes that have changed (including withdrawn) are included in the update messages. The BGP speaker 710 may also perform some kind of manipulation/change to the data before transmitting it in the update messages 400. Once created, the BGP updates messages are passed to the TCP layer and other lower layers of the network protocol stack, where the messages are formatted and transmitted over the communication links as packets to the peer routers.

Fig. 9 is a flowchart illustrating a sequence of steps pertaining to data flow throughout the BGP implementation of the distributed software architecture according

to the present invention. Data flow in the BGP implementation 700 occurs in response to update messages 400 received at and transmitted from the router 200. These update messages are, in turn, used in connection with route selection in the router. The sequence starts at Step 900 and proceeds to Step 902 where each BGP speaker receives update messages 400 from its peers and, in Step 904, processes those received messages by applying inbound policy to the routes announced in those messages. The speaker then downloads all routes received from its peers (except those "filtered" by policy) to the bRIB 720 in Step 906.

The bRIB, in turn, examines all the routes that it receives from the various BGP speakers and, in Step 908, performs a first stage of route selection to compute a set of best paths/routes. In Step 910, the bRIB 720 downloads those best routes to the global RIB 730 for the router which, in Step 912, performs a second (and final) stage of route selection to compute optimal best path routes. In Step 914, the bRIB uploads the best routes for each prefix to each BGP speaker. In Step 916, the BGP speaker 710 performs further processing by applying outbound policy on those best routes and, in Step 918, determines whether the applied policy blocks transmission of one or more routes that had been previously transmitted. If so, those routes are withdrawn from service using the withdrawn routes field 404 of update message 400 (Step 920). Otherwise, the speaker transmits (advertises) the best routes to its peers as update messages in Step 922 and the sequence ends at Step 924.

The distributed software architecture described herein overcomes conventional CPU and memory constraints to provide a scalable routing protocol mechanism. The architecture also exploits the frequency of update message processing by distributing the routing protocol functions across processing resources of the router. Because the computer network 100 is not entirely stable, each event that alters the network topology (e.g., a communication link or segment going offline) is transformed into a BGP update message 400 that a BGP router 200 receives and may need to transmit. There is an average frequency of update messages that the protocol must handle and that translates into a CPU load. A BGP distributed implementation that operates within its scaling envelope is, on average, able to process update messages substantially as soon as they are received to thereby keep the data flow moving through the router.

As for scalability and convergence, there is a certain amount of extra latency that is incurred by going to the distributed architecture because of the IPC mechanism. This latency is "traded off" for total volume supported by the router. On the convergence spectrum, minimum average latency (as opposed to minimum latency) is a goal. Since all speakers 710 provide all (filtered) routes to the bRIB 720, the distributed architecture is asynchronous and eventually converges to the same correct state depending on timing issues.

Advantageously, the inventive architecture allows the workload of the distributed software implementation to be apportioned among multiple processes, effecting a more scalable BGP implementation capable of allowing a user the ability to dedicate resources to particular groups of peers, while maintaining the external appearance of a single BGP protocol instance. As noted, the BGP implementation may be further apportioned among several processors in a multiprocessor router (or nodes in a multi-node cluster), such that the total required processing is distributed among the processors, instead of concentrated on a single processor. As the number of routing peers increases, additional processors can be added to the router to handle the extra processing required, thereby avoiding overloading of a single processor and, hence, adversely affecting the convergence time of the protocol.

A secondary advantage of the invention is improved fault-tolerance. If a particular processor running a BGP speaker in the router fails, only the routing peers assigned to that speaker are affected. If the failing processor is running the bRIB process, no peers are affected and the router can recover simply by having each speaker resend all of its paths to the bRIB when it restarts. In the absence of the inventive distributed architecture, a failure to the processor running the concentrated BGP implementation would affect all peers of that implementation.

A tertiary advantage of the invention is that groups of peers can be co-located on given processors, separate from peers on the other processors, to effect feature separation or resource isolation. Furthermore, the inventive architecture maintains the autonomy of the peers, such that each peer can be configured ("placed") in a speaker arbitrarily, with the actual placement policy being determined by the user. For example, the user could place all peers exchanging routes for IPv4 on one processor, while

peers exchanging routes for IPv6 could be placed on a different processor. Churn in the topology of a network will only slightly impact another network, effectively isolating the delivery of each service from perturbations in the churned network.

In sum, the inventive architecture increases the scalability (and thus performance under load) of the BGP routing protocol, while simultaneously making the protocol more fault-tolerant. Because the invention is directed to performance of a BGP implementation with a large number of peers, it has the greatest applicability to large service providers; however, the invention is not intrinsically limited to that space.

The foregoing description has been directed to specific embodiments of this invention. It will be apparent, however, that other variations and modifications may be made to the described embodiments, with the attainment of some or all of their advantages. For instance, it is expressly contemplated that the teachings of this invention, including the various processes described herein, can be implemented as software, including a computer-readable medium having program instructions executing on a computer, hardware, firmware, or a combination thereof. In addition, it is understood that the data structures described herein can include additional information while remaining within the scope of the present invention. Furthermore, the inventive distributed software architecture may apply generally to distance vector routing protocols, e.g., IGRP, EIGRP or RIP, as well as to a label distribution protocol (LDP). Accordingly this description is to be taken only by way of example and not to otherwise limit the scope of the invention. Therefore, it is the object of the appended claims to cover all such variations and modifications as come within the true spirit and scope of the invention.

What is claimed is:

CLAIMS

1. A router configured to implement a routing protocol, the router comprising:

a plurality of processors;

a switch fabric interconnecting the processors; and

a plurality of processes running on the processors, the processes including two or more speakers and a protocol routing information base (RIB), each speaker configured to (i) handle one or more connections to peer routers of the router, (ii) receive and store routes from the peer routers, (iii) apply inbound policy to the routes received from the peer routers and (iv) download all the routes received from the peer routers, except those filtered by the inbound policy, to the protocol RIB, the protocol RIB configured to perform a first stage of route selection to compute a set of best routes from among the routes downloaded from all of the speakers of the router.

2. The router of Claim 1 further comprising a local RIB (loc-RIB) maintained by the protocol RIB and configured to store the set of best routes computed by the protocol RIB.

3. The router of Claim 2 wherein the processes further include a global RIB configured to maintain a routing table for the router.

4. The router of Claim 3 wherein the protocol RIB is configured to download the set of best routes from the loc-RIB to the global RIB, the global RIB further configured to use the downloaded set of best routes from the loc-RIB, along with other sets of best routes downloaded from other routing protocols, to perform a second stage of route selection that computes optimal routes for installation in the routing table.

5. The router of Claim 4 further comprising one or more line cards connected to the switch fabric, each line card configured to render forwarding decisions on packets received at the router using a forwarding information base (FIB) constructed from the optimal routes installed in the routing table.

6. The router of Claim 2 wherein the protocol RIB is further configured to upload the set of best routes to the speakers to allow the speakers to advertise the best routes to the peer routers.

7. The router of Claim 6 wherein each speaker is further configured to apply outbound policy to the best routes prior to advertising them to the peer routers.

8. The router of Claim 7 wherein each speaker is further configured to advertise the best routes using update messages.

9. The router of Claim 8 wherein the routing protocol is a Border Gateway Protocol (BGP) and wherein the protocol RIB is a BGP RIB (bRIB).

10. The router of Claim 1 wherein the routing protocol is a distance vector routing protocol.

11. A method for implementing a routing protocol in a router , the method comprising the steps of:

providing a plurality of processors of the router;

interconnecting the processors;

running at least two speakers on at least two first processors of the plurality of processors, each speaker:

handling one or more connections to peers of the router,

receiving and storing routes from the peers,

applying inbound policy to the routes received from the peers, and

running a protocol routing information base (RIB) on a second processor of the plurality of processors, each speaker downloading all the routes received from the peers, except those filtered by the inbound policy, to the protocol RIB, the protocol RIB performing a first stage of route selection to compute best routes from among the routes downloaded from all of the speakers of the router.

12. The method of Claim 11 further comprising the steps of:

maintaining a local RIB (loc-RIB) at the protocol RIB; and

storing the best routes computed by the protocol RIB in the loc-RIB.

13. The method of Claim 12 further comprising the step of running a global RIB on a third processor of the plurality of processors, the global RIB maintaining a routing table for the router.

14. The method of Claim 13 wherein the second and third processors are the same processor.

15. The method of Claim 13 further comprising the steps of:

downloading the best routes from the loc-RIB to the global RIB;

performing a second stage of route selection at the global RIB using the downloaded best routes from the loc-RIB, along with other sets of best routes downloaded from other routing protocols, the second stage of route selection computing optimal routes for installation in the routing table.

16. The method of Claim 15 further comprising the steps of:

interconnecting one or more line cards to the plurality of processors;

constructing a forwarding information base (FIB) at each line card, the FIB constructed from the optimal routes installed in the routing table; and

rendering forwarding decisions on packets received at each line card using the FIB.

17. The method of Claim 12 further comprising the steps of:

uploading the best routes from the bRIB to each speaker;

applying outbound policy to the uploaded best routes; and

advertising resulting best routes to the peers.

18. An apparatus adapted to implement a Border Gateway Protocol (BGP) routing protocol in a router , the apparatus comprising:

means for running a BGP speaker on a first processor of a plurality of interconnected processors, the BGP speaker including:

means for handling one or more connections to peers of the router,

means for receiving and storing routes from those peers,

means for applying inbound policy to the routes received from the peers, and

means for running a BGP routing information base (bRIB) on a second processor of the plurality of interconnected processors, the BGP speaker further including means for downloading all the routes received from the peers, except those filtered by the inbound policy, to the bRIB, the bRIB including means for performing a first stage of route selection to compute best routes from among the routes downloaded from the BGP speaker.

19. The apparatus of Claim 18 further comprising:

means for maintaining a local RIB (loc-RIB) at the bRIB; and

means for storing the best routes computed by the bRIB in the loc-RIB.

20. A computer readable medium containing executable program instructions for implementing a routing protocol in a router , the executable program instructions comprising program instructions for:

running at least two speakers on at least two first processors of a plurality of interconnected processors, each speaker:

handling one or more connections to peers of the router,

receiving and storing routes from those peers,

applying inbound policy to the routes received from the peers, and

running a protocol routing information base (RIB) on a second processor of the plurality of interconnected processors, each speaker downloading all the routes received from the peers, except those filtered by the inbound policy, to the protocol RIB, the

protocol RIB performing a first stage of route selection to compute best routes from among the routes downloaded from all of the speakers of the router.

21. The computer readable medium of Claim 20 further comprising program instructions for:

maintaining a local RIB (loc-RIB) at the protocol RIB; and

storing the best routes computed by the protocol RIB in the loc-RIB.

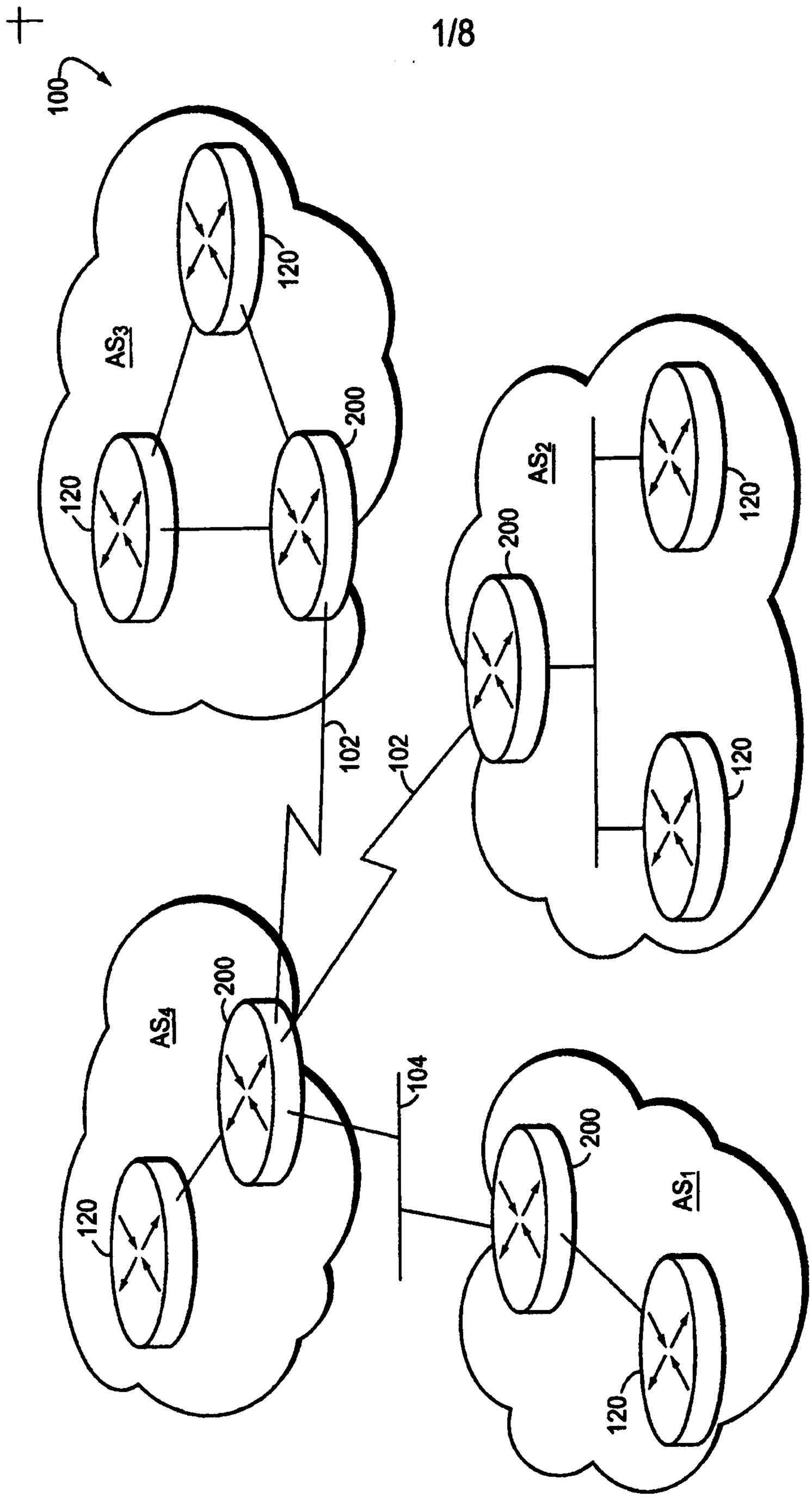


FIG. 1
PRIOR ART

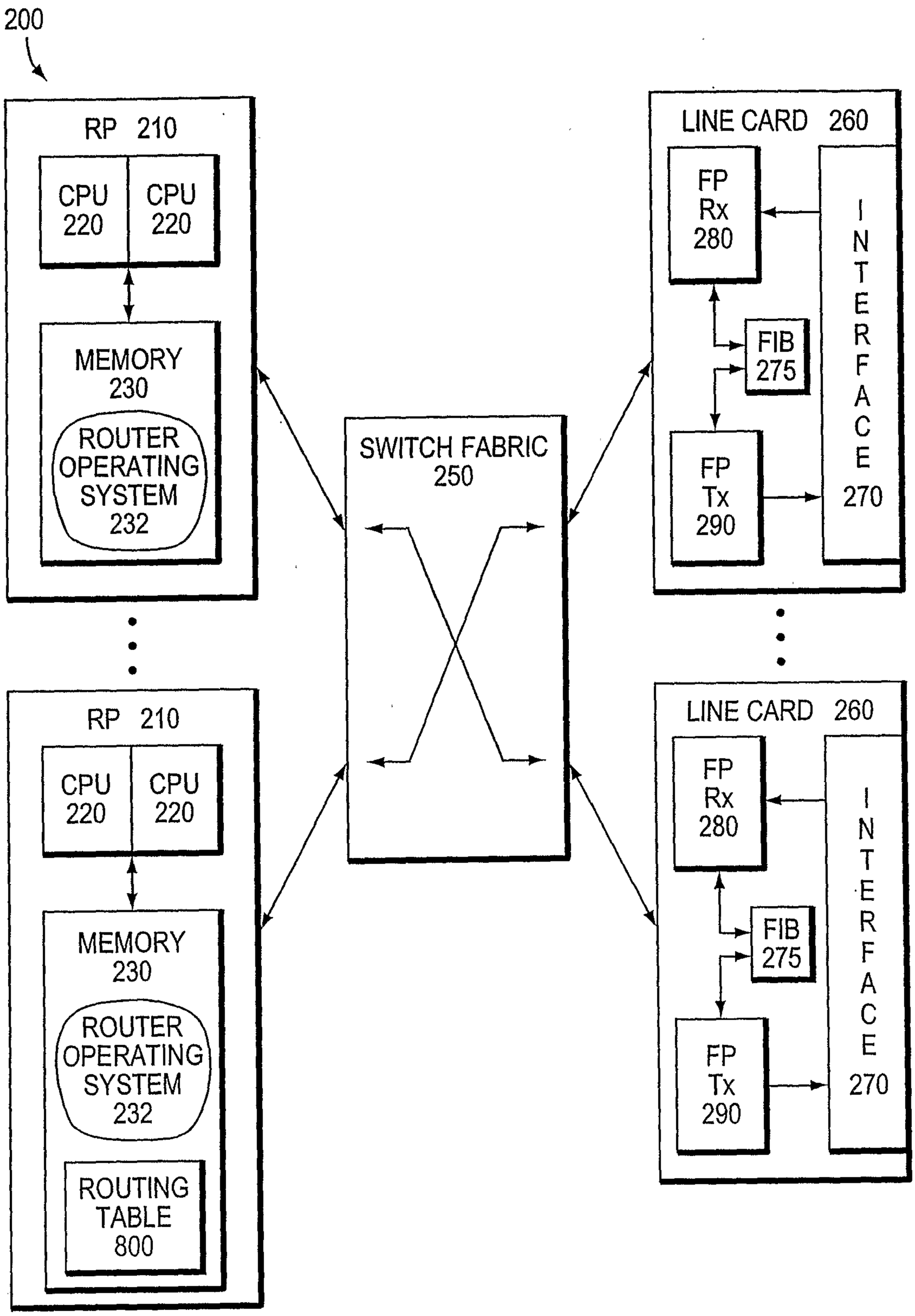

200
RP 210
CPU 220
CPU 220
MEMORY 230
ROUTER OPERATING SYSTEM 232
ROUTING TABLE 800
SWITCH FABRIC 250
LINE CARD 260
FP Rx 280
FIB 275
FP Tx 290
INTERFACE 270

FIG. 2

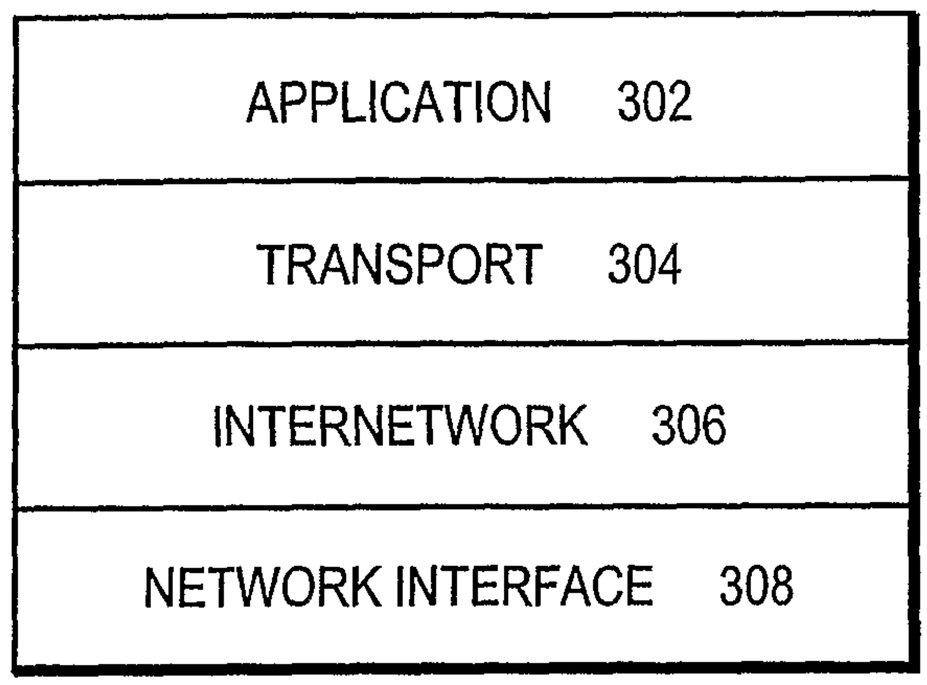

300
APPLICATION 302
TRANSPORT 304
INTERNETWORK 306
NETWORK INTERFACE 308

FIG. 3

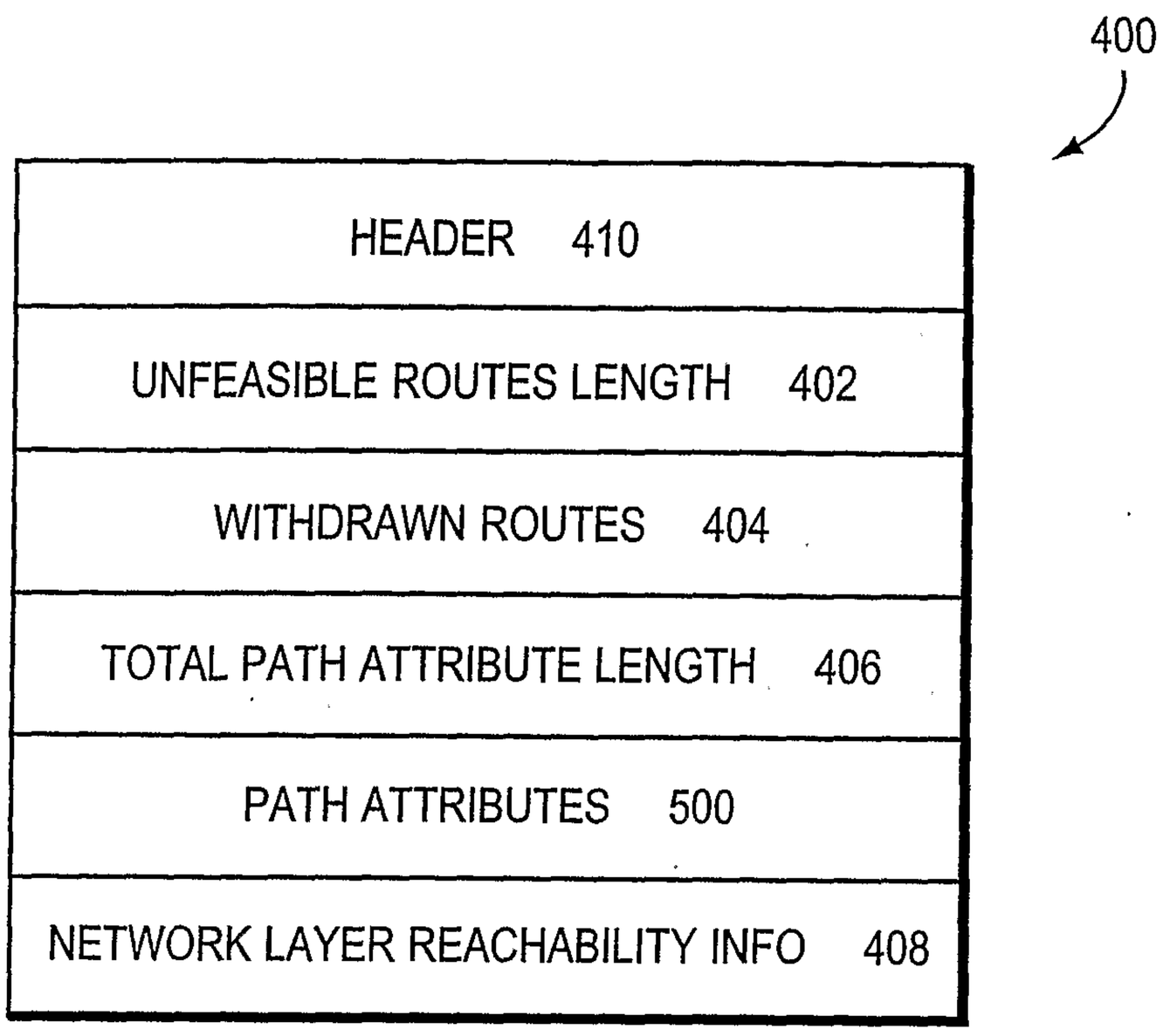

400
HEADER 410
UNFEASIBLE ROUTES LENGTH 402
WITHDRAWN ROUTES 404
TOTAL PATH ATTRIBUTE LENGTH 406
PATH ATTRIBUTES 500
NETWORK LAYER REACHABILITY INFO 408

FIG. 4

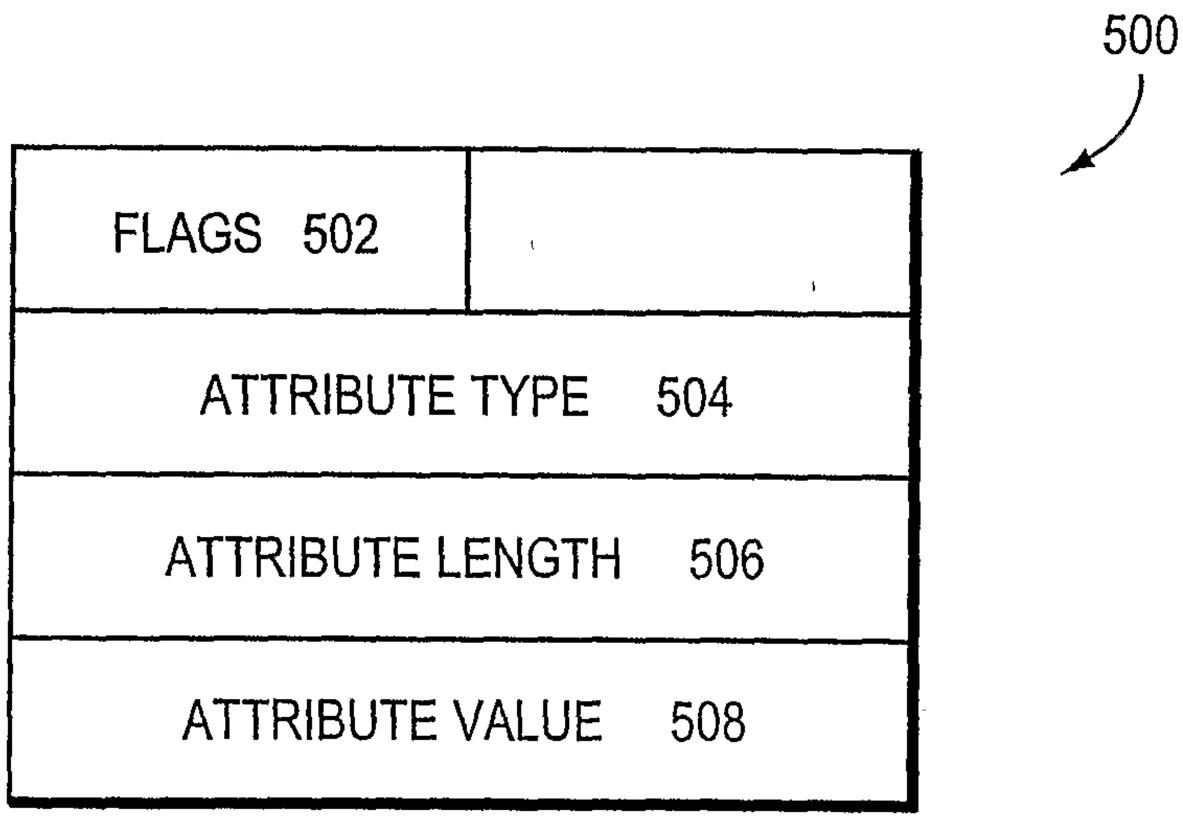

500
FLAGS 502
ATTRIBUTE TYPE 504
ATTRIBUTE LENGTH 506
ATTRIBUTE VALUE 508

FIG. 5

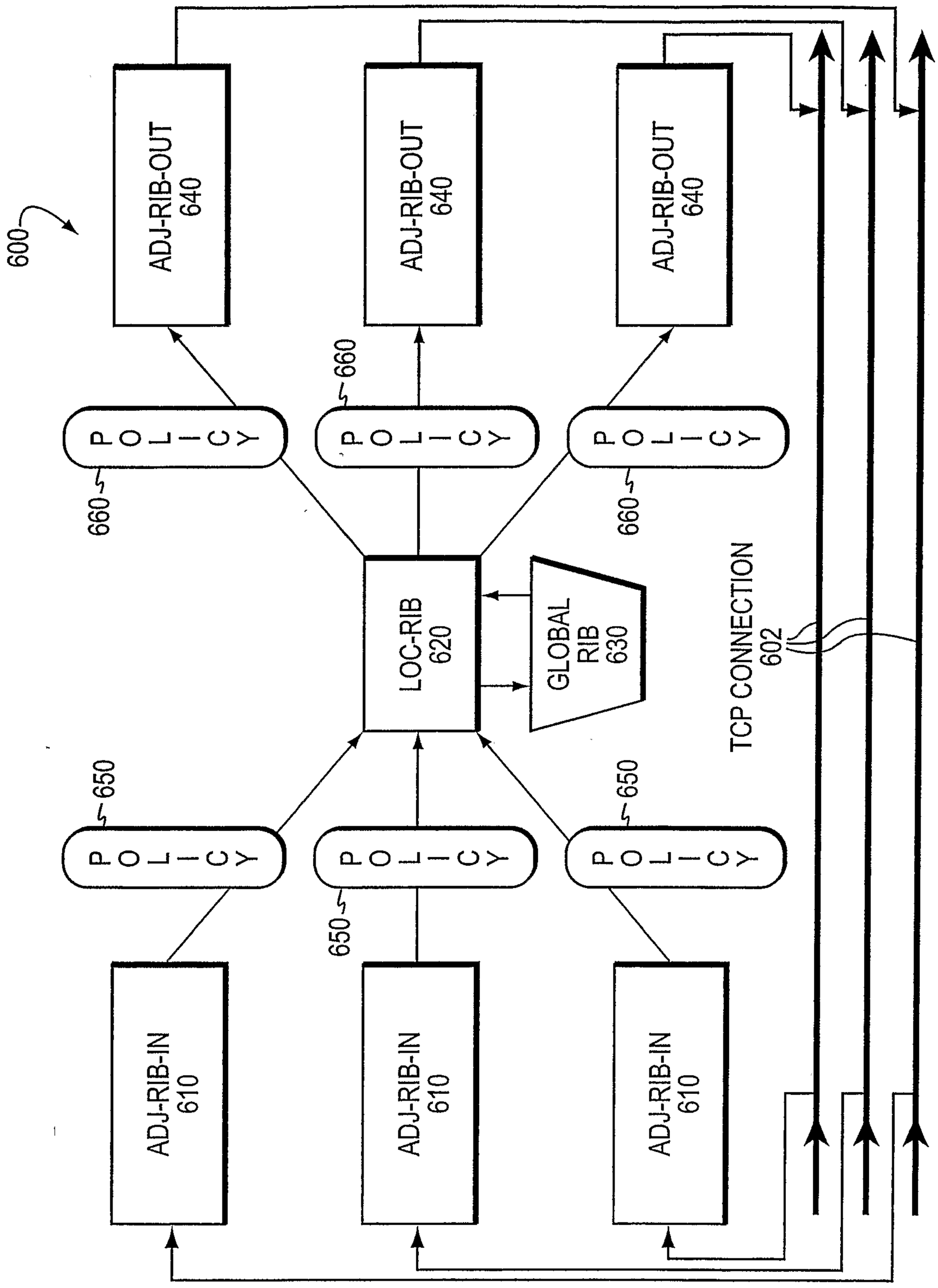

600
ADJ-RIB-OUT 640
ADJ-RIB-OUT 640
ADJ-RIB-OUT 640
660
POLICY
660
POLICY
660
POLICY
LOC-RIB 620
GLOBAL RIB 630
TCP CONNECTION 602
650
POLICY
650
POLICY
650
POLICY
ADJ-RIB-IN 610
ADJ-RIB-IN 610
ADJ-RIB-IN 610

FIG. 6

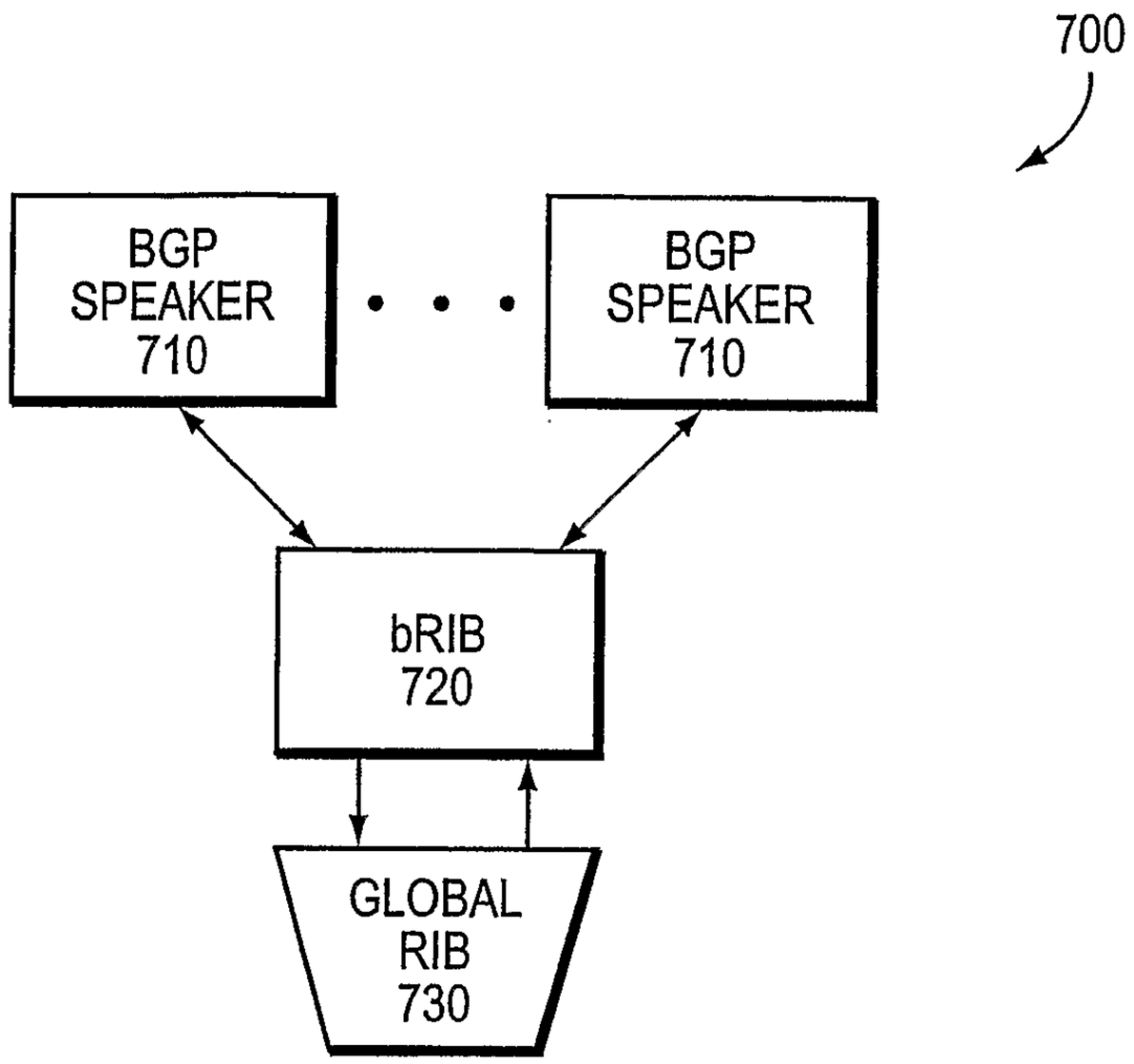


FIG. 7

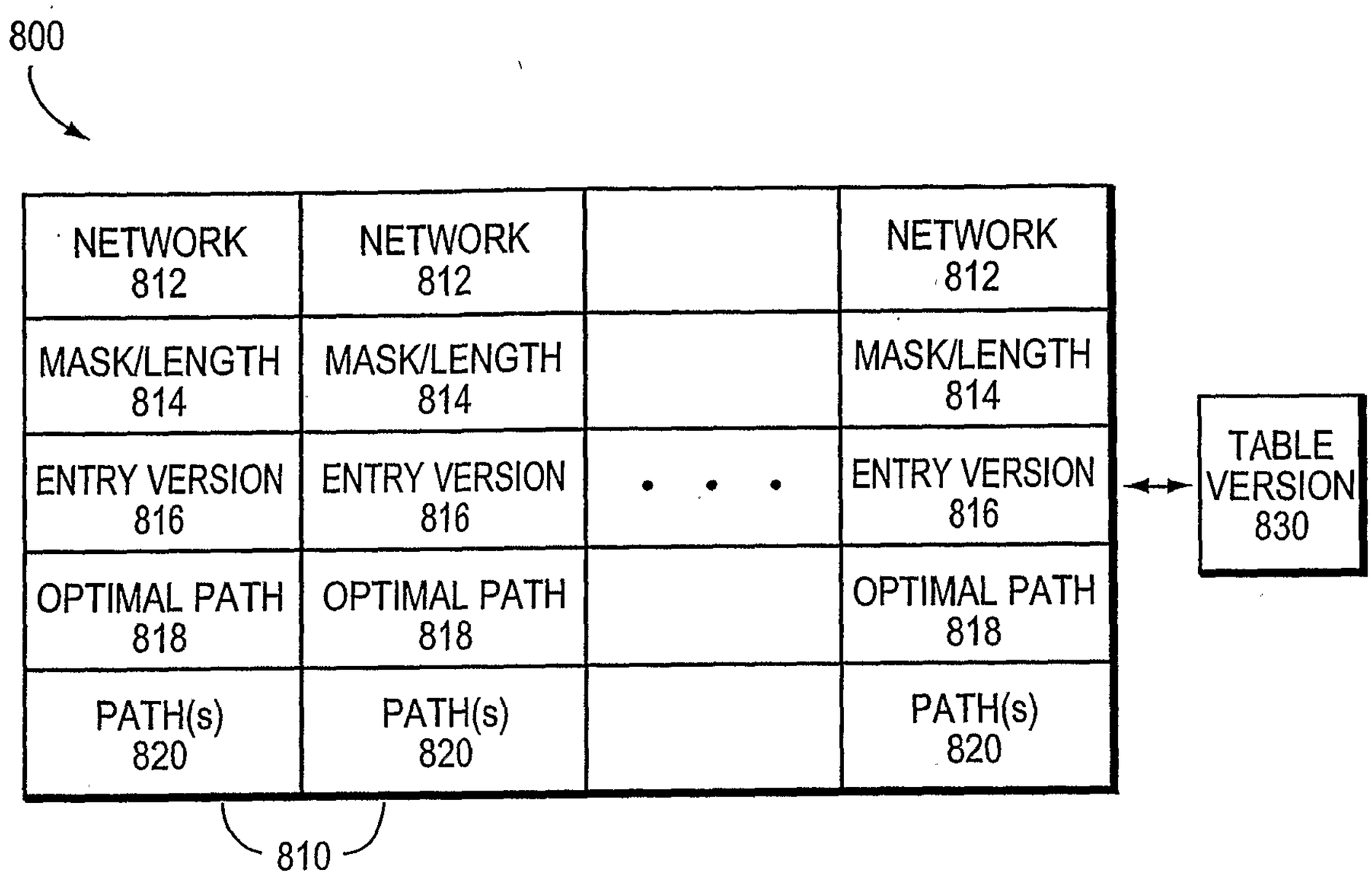

800
NETWORK 812
NETWORK 812
NETWORK 812
MASK/LENGTH 814
MASK/LENGTH 814
MASK/LENGTH 814
ENTRY VERSION 816
ENTRY VERSION 816
ENTRY VERSION 816
OPTIMAL PATH 818
OPTIMAL PATH 818
OPTIMAL PATH 818
PATH(s) 820
PATH(s) 820
PATH(s) 820
TABLE VERSION 830
810

FIG. 8

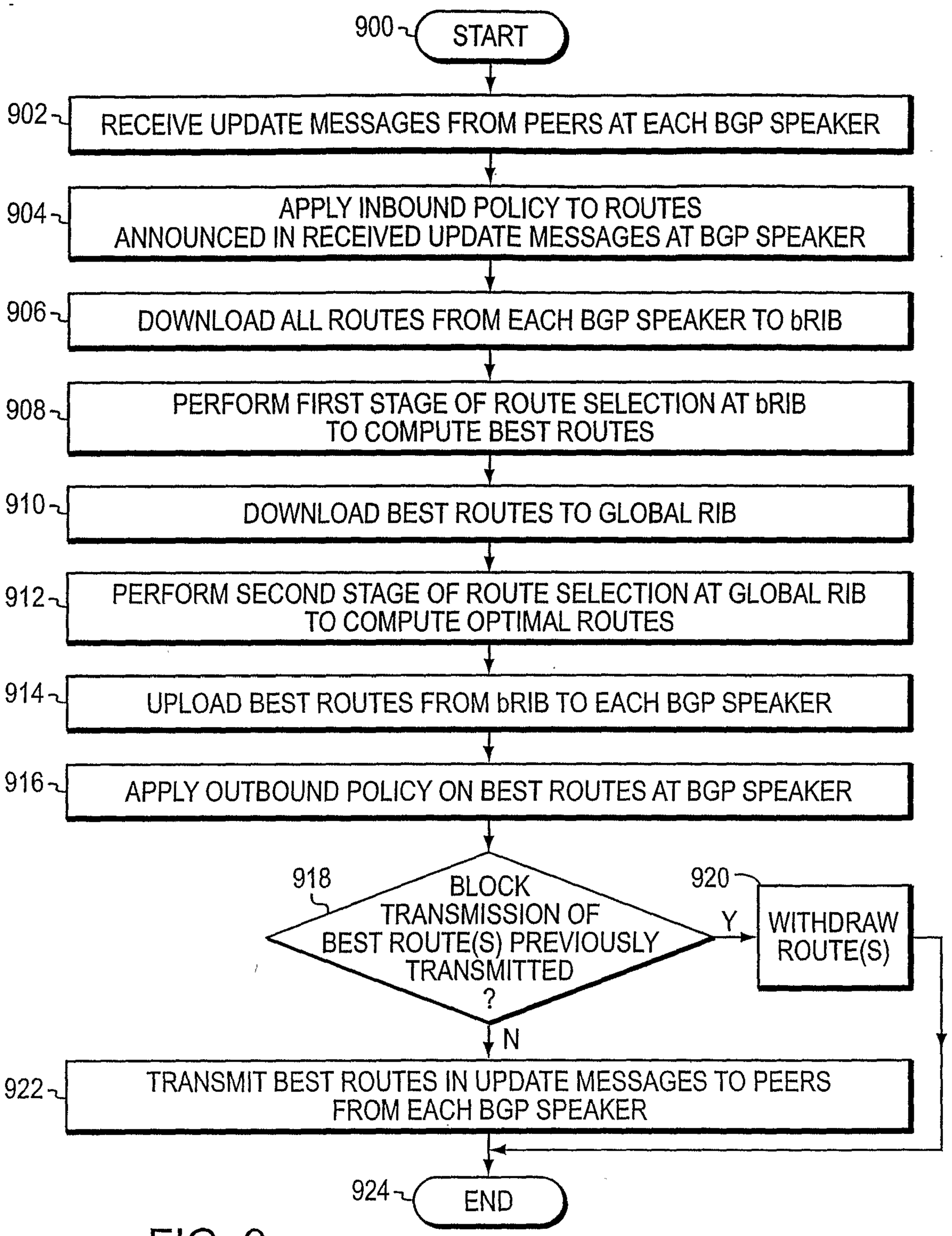
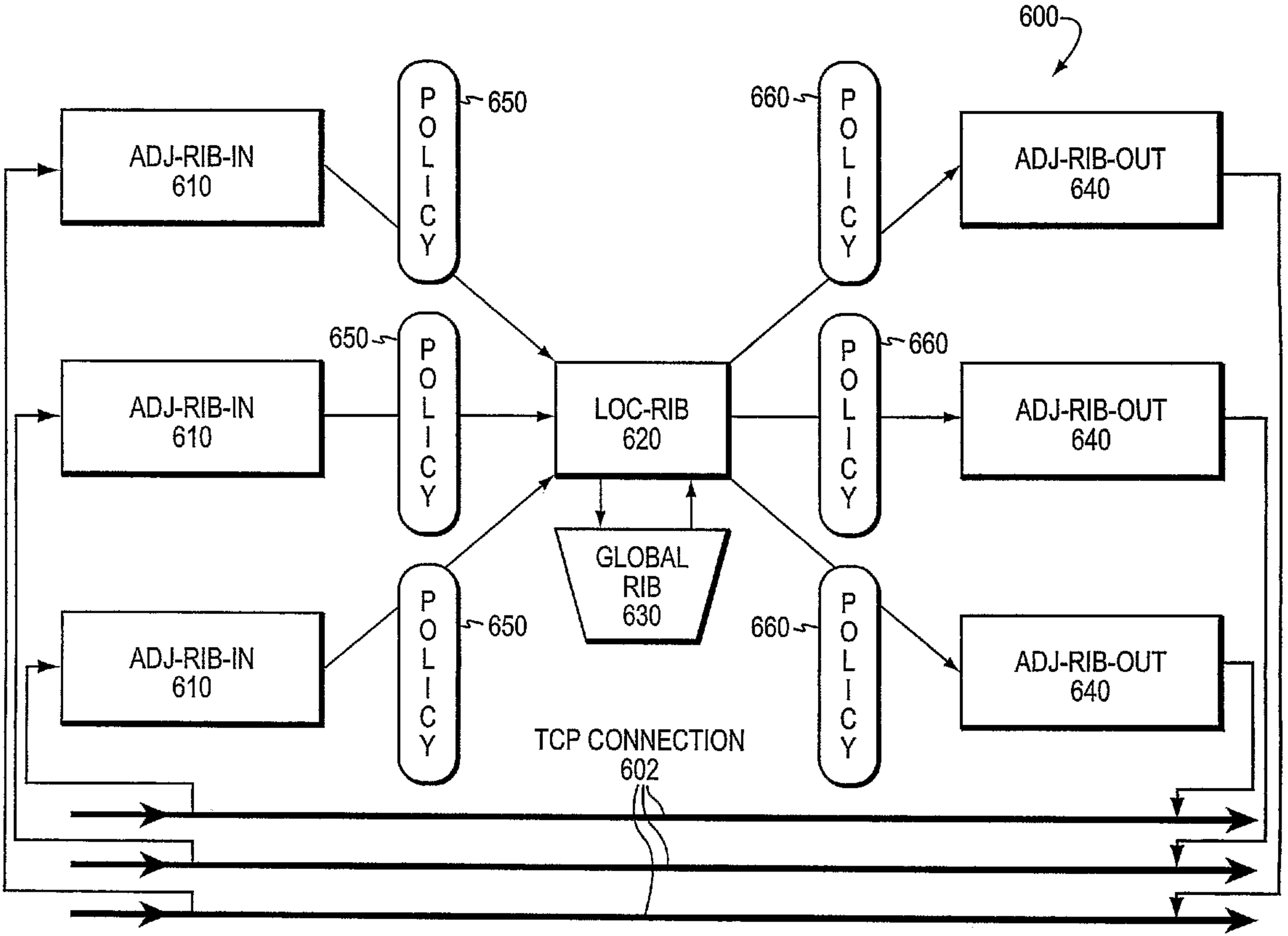

900
START
902
RECEIVE UPDATE MESSAGES FROM PEERS AT EACH BGP SPEAKER
904
APPLY INBOUND POLICY TO ROUTES ANNOUNCED IN RECEIVED UPDATE MESSAGES AT BGP SPEAKER
906
DOWNLOAD ALL ROUTES FROM EACH BGP SPEAKER TO bRIB
908
PERFORM FIRST STAGE OF ROUTE SELECTION AT bRIB TO COMPUTE BEST ROUTES
910
DOWNLOAD BEST ROUTES TO GLOBAL RIB
912
PERFORM SECOND STAGE OF ROUTE SELECTION AT GLOBAL RIB TO COMPUTE OPTIMAL ROUTES
914
UPLOAD BEST ROUTES FROM bRIB TO EACH BGP SPEAKER
916
APPLY OUTBOUND POLICY ON BEST ROUTES AT BGP SPEAKER
918
BLOCK TRANSMISSION OF BEST ROUTE(S) PREVIOUSLY TRANSMITTED ?
Y
920
WITHDRAW ROUTE(S)
N
922
TRANSMIT BEST ROUTES IN UPDATE MESSAGES TO PEERS FROM EACH BGP SPEAKER
924
END

FIG. 9


600
ADJ-RIB-IN
610
ADJ-RIB-IN
610
ADJ-RIB-IN
610
POLICY
650
650
650
LOC-RIB
620
GLOBAL
RIB
630
660
660
660
ADJ-RIB-OUT
640
ADJ-RIB-OUT
640
ADJ-RIB-OUT
640
TCP CONNECTION
602