(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2022년05월13일
(11) 등록번호 10-2397876
(24) 등록일자 2022년05월10일

(51) 국제특허분류(Int. Cl.)
H04L 47/00 (2022.01) *H04L 45/00* (2022.01)
H04L 49/00 (2022.01) *H04L 65/40* (2022.01)
(52) CPC특허분류
H04L 47/125 (2022.05)
H04L 45/48 (2022.05)
(21) 출원번호 10-2017-7003836
(22) 출원일자(국제) 2015년07월09일
심사청구일자 2020년07월09일
(85) 번역문제출일자 2017년02월10일
(65) 공개번호 10-2017-0029595
(43) 공개일자 2017년03월15일
(86) 국제출원번호 PCT/US2015/039767
(87) 국제공개번호 WO 2016/007760
국제공개일자 2016년01월14일
(30) 우선권주장
62/023,321 2014년07월11일 미국(US)
(뒷면에 계속)
(56) 선행기술조사문헌
US05138615 A*
US20130121154 A1*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
오라클 인터내셔날 코포레이션
미국, 캘리포니아 94065, 레드우드 쇼어스 엠에스 5오피7, 오라클 파크웨이 500
(72) 발명자
자히드 페로즈
노르웨이 엔-1325 리사케르 피.오. 박스 134
그란 에른스트 군나르
미국 캘리포니아 94065 레드우드 쇼오스 엠/에스 5오피7 오라클 파크웨이 500
(뒷면에 계속)
(74) 대리인
박장원

전체 청구항 수 : 총 18 항 심사관 : 김대성

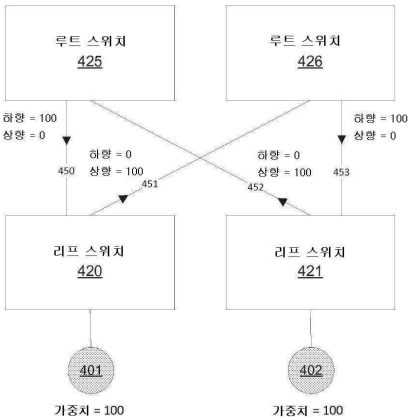
(54) 발명의 명칭 **고성능 컴퓨팅(HPC) 환경에서 효율적인 로드-발란싱을 지원하기 위한 시스템 및 방법**

(57) 요 약

네트워크 환경에서 트리 토폴로지로 정렬된 복수의 말단 노드들 및 복수의 스위치들 간에 효율적인 로드 발란싱을 지원하기 위한 방법들 및 시스템들이 제공된다. 본 방법들 및 시스템들은 복수의 말단 노드들을 정렬시킬 수 있고, 여기서 복수의 말단 노드들은 수신 가중치가 감소하는 내림 차순으로 정렬된다. 본 방법 및 시스템은

(뒷면에 계속)

대 표 도 - 도9



또한, 복수의 말단 노드들을 수신 가중치들이 감소하는 내림 차순으로 라우팅할 수 있고, 여기서 라우팅하는 것은 적어도 하나의 하향-진행 포트 및 적어도 하나의 상향-진행 포트를 선택하는 것을 포함한다. 또한, 본 방법 및 시스템은, 각각의 선택된 하향-진행 포트 상의 누적 하향 가중치를 라우팅되는 말단 노드의 수신 가중치만큼 증가시킬 수 있고, 그리고 각각의 선택된 상향-진행 포트 상의 누적 상향 가중치를 라우팅되는 말단 노드의 수신 가중치만큼 증가시킬 수 있다.

(52) CPC특허분류
H04L 49/25 (2022.05)
H04L 67/1001 (2022.05)
(30) 우선권주장
62/049,466 2014년09월12일 미국(US)
14/792,070 2015년07월06일 미국(US)

(72) 발명자
보그단스키 바르토츠
노르웨이 엔-0275 오슬로 호프 테라쎄 15 에이치0203
존슨 비요른 대그
노르웨이 엔-0687 오슬로 빌베르그그렌다 9

명 세 서

청구범위

청구항 1

네트워크 환경(network environment)에서 트리 토폴로지(tree topology)로 정렬된 복수의 말단 노드(end node)들 및 복수의 스위치(switch)들 간에 효율적인 로드 발란싱(load balancing)을 지원하기 위한 방법으로서, 상기 방법은,

상기 복수의 말단 노드들을 정렬(sorting)시키는 단계와, 여기서 상기 복수의 말단 노드들은 상기 복수의 스위치들 중 하나 이상의 스위치에 연결되고, 상기 복수의 말단 노드들은 수신 가중치(receive weight)가 감소하는 내림 차순(decreasing order)으로 정렬되며;

상기 복수의 말단 노드들을 수신 가중치들이 감소하는 상기 내림 차순으로 라우팅(routing)하는 단계와, 여기서 상기 라우팅하는 단계는 적어도 하나의 하향-진행 포트(down-going port) 및 적어도 하나의 상향-진행 포트(up-going port)를 선택하는 것을 포함하고;

각각의 선택된 하향-진행 포트 상의 누적 하향 가중치(accumulated downward weight)를 상기 라우팅되는 말단 노드의 수신 가중치만큼 증가시키는 단계와; 그리고

각각의 선택된 상향-진행 포트 상의 누적 상향 가중치(accumulated upward weight)를 상기 라우팅되는 말단 노드의 수신 가중치만큼 증가시키는 단계를 포함하며,

상기 적어도 하나의 하향-진행 포트를 선택하는 것은,

복수의 하향-진행 포트들을 비교하는 것, 그리고 가장 적은 누적 하향 가중치를 갖는 하향-진행 포트를 선택하는 것;

둘 이상의 하향-진행 포트들이 가장 적은 누적 하향 가중치를 갖는 것에 응답하여, 가장 적은 누적 하향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들을 비교하는 것, 그리고 가장 적은 누적 하향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들로부터, 가장 적은 누적 상향 가중치를 갖는 하향-진행 포트를 선택하는 것; 그리고

상기 둘 이상의 하향-진행 포트들이 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 것에 응답하여, 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들을 비교하는 것, 그리고 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들로부터, 가장 작은 글로벌 고유 식별자를 갖는 하향-진행 포트를 선택하는 것을 포함하는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 방법.

청구항 2

제1항에 있어서,

상기 적어도 하나의 상향-진행 포트를 선택하는 것은, 복수의 상향-진행 포트들을 비교하는 것과, 그리고 가장 적은 누적 상향 가중치를 갖는 상향-진행 포트를 선택하는 것을 포함하는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 방법.

청구항 3

제1항 또는 제2항에 있어서,

상기 방법은 또한, 상기 복수의 말단 노드들 각각에 대한 수신 가중치를 수신하는 단계를 포함하는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 방법.

청구항 4

제3항에 있어서,

상기 복수의 말단 노드들 각각에 대한 수신 가중치는, 관리자(administrator)로부터의 입력으로부터 수신되는

것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 방법.

청구항 5

제3항에 있어서,

상기 복수의 말단 노드들 각각에 대한 수신 가중치는, 상기 복수의 말단 노드들 각각 상의 모니터링되는 트래픽(monitored traffic)과 각각 관련된 입력으로부터 수신되는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 방법.

청구항 6

제1항에 있어서,

상기 복수의 스위치들 및 상기 복수의 말단 노드들은 고-성능 컴퓨팅(High-Performance Computing, HPC) 클러스터(cluster)에서의 사용을 위한 팻-트리 토폴로지(fat-tree topology)로 정렬되는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 방법.

청구항 7

제3항에 있어서,

상기 복수의 말단 노드들 각각에 대한 수신 가중치는, 계산을 통해 계산되는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 방법.

청구항 8

제7항에 있어서,

상기 계산은 적어도 선형 변환(linear transformation)을 포함하는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 방법.

청구항 9

삭제

청구항 10

네트워크 환경에서 트리 토폴로지로 정렬된 복수의 말단 노드들 및 복수의 스위치들 간에 효율적인 로드 발란싱을 지원하기 위한 시스템으로서, 상기 시스템은,

하나 이상의 마이크로프로세서들과; 그리고

상기 하나 이상의 마이크로프로세서들 상에서 실행되는 프로세서를 포함하고,

상기 프로세서는,

상기 복수의 말단 노드들을 정렬시키는 단계와, 여기서 상기 복수의 말단 노드들은 상기 복수의 스위치들 중 하나 이상의 스위치에 연결되고, 상기 복수의 말단 노드들은 수신 가중치가 감소하는 내림 차순으로 정렬되며;

상기 복수의 말단 노드들을 수신 가중치들이 감소하는 상기 내림 차순으로 라우팅하는 단계와, 여기서 상기 라우팅하는 단계는 적어도 하나의 하향-진행 포트 및 적어도 하나의 상향-진행 포트를 선택하는 것을 포함하고;

각각의 선택된 하향-진행 포트 상의 누적 하향 가중치를 상기 라우팅되는 말단 노드의 수신 가중치만큼 증가시키는 단계와; 그리고

각각의 선택된 상향-진행 포트 상의 누적 상향 가중치를 상기 라우팅되는 말단 노드의 수신 가중치만큼 증가시키는 단계를

포함하는 단계들을 수행하도록 동작하며,

상기 적어도 하나의 하향-진행 포트를 선택하는 것은,

복수의 하향-진행 포트들을 비교하는 것, 그리고 가장 적은 누적 하향 가중치를 갖는 하향-진행 포트를 선택하는 것;

둘 이상의 하향-진행 포트들이 가장 적은 누적 하향 가중치를 갖는 것에 응답하여, 가장 적은 누적 하향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들을 비교하는 것, 그리고 가장 적은 누적 하향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들로부터, 가장 적은 누적 상향 가중치를 갖는 하향-진행 포트를 선택하는 것; 그리고

상기 둘 이상의 하향-진행 포트들이 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 것에 응답하여, 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들을 비교하는 것, 그리고 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들로부터, 가장 작은 글로벌 고유 식별자를 갖는 하향-진행 포트를 선택하는 것을 포함하는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 시스템.

청구항 11

제10항에 있어서,

상기 복수의 스위치들 및 상기 복수의 말단 노드들은 고-성능 컴퓨팅(High-Performance Computing, HPC) 클러스터(cluster)에서의 사용을 위한 팻-트리 토폴로지(fat-tree topology)로 정렬되는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 시스템.

청구항 12

제10항 또는 제11항에 있어서,

상기 프로세서는, 상기 복수의 말단 노드들 각각에 대한 수신 가중치를 수신하도록 동작하는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 시스템.

청구항 13

제12항에 있어서,

상기 복수의 말단 노드들 각각에 대한 수신 가중치는, 관리자로부터의 입력으로부터 수신되는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 시스템.

청구항 14

제12항에 있어서,

상기 복수의 말단 노드들 각각에 대한 수신 가중치는, 계산을 통해 계산되는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 시스템.

청구항 15

제14항에 있어서,

상기 계산은 적어도 선형 변환을 포함하는 것을 특징으로 하는 효율적인 로드 발란싱을 지원하기 위한 시스템.

청구항 16

삭제

청구항 17

비-일시적 컴퓨터 판독가능 저장 매체(non-transitory computer readable storage medium)로서, 상기 비-일시적 컴퓨터 판독가능 저장 매체는 상기 비-일시적 컴퓨터 판독가능 저장 매체 상에 저장되는 명령들을 포함하고, 상기 명령들은 네트워크 환경에서 트리 토폴로지로 정렬된 복수의 말단 노드들 및 복수의 스위치들 간에 효율적인 로드 발란싱을 지원하기 위한 명령들이고, 상기 명령들은 실행될 때 시스템으로 하여금,

상기 복수의 말단 노드들을 정렬시키는 단계와, 여기서 상기 복수의 말단 노드들은 또한, 상기 트리 토폴로지에

서 상기 복수의 스위치들 중 하나 이상의 스위치에 연결되고, 상기 복수의 말단 노드들은 수신 가중치가 감소하는 내림 차순으로 정렬되며;

상기 복수의 말단 노드들을 수신 가중치들이 감소하는 상기 내림 차순으로 라우팅하는 단계와, 여기서 상기 라우팅하는 단계는 적어도 하나의 하향-진행 포트 및 적어도 하나의 상향-진행 포트를 선택하는 것을 포함하고;

각각의 선택된 하향-진행 포트 상의 누적 하향 가중치를 상기 라우팅되는 말단 노드의 수신 가중치만큼 증가시키는 단계와; 그리고

각각의 선택된 상향-진행 포트 상의 누적 상향 가중치를 상기 라우팅되는 말단 노드의 수신 가중치만큼 증가시키는 단계를

포함하는 단계들을 수행하도록 하며,

상기 적어도 하나의 하향-진행 포트를 선택하는 것은,

복수의 하향-진행 포트들을 비교하는 것, 그리고 가장 적은 누적 하향 가중치를 갖는 하향-진행 포트를 선택하는 것;

둘 이상의 하향-진행 포트들이 가장 적은 누적 하향 가중치를 갖는 것에 응답하여, 가장 적은 누적 하향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들을 비교하는 것, 그리고 가장 적은 누적 하향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들로부터, 가장 적은 누적 상향 가중치를 갖는 하향-진행 포트를 선택하는 것; 그리고

상기 둘 이상의 하향-진행 포트들이 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 것에 응답하여, 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들을 비교하는 것, 그리고 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 상기 둘 이상의 하향-진행 포트들로부터, 가장 작은 글로벌 고유 식별자를 갖는 하향-진행 포트를 선택하는 것을 포함하는 것을 특징으로 하는 비-일시적 컴퓨터 판독가능 저장 매체.

청구항 18

제17항에 있어서,

상기 적어도 하나의 상향-진행 포트를 선택하는 것은, 복수의 상향-진행 포트들을 비교하는 것과, 그리고 가장 적은 누적 상향 가중치를 갖는 상향-진행 포트를 선택하는 것을 포함하는 것을 특징으로 하는 비-일시적 컴퓨터 판독가능 저장 매체.

청구항 19

제17항에 있어서,

상기 단계들은 또한, 상기 복수의 스위치들 중의 스위치 상의 상기 복수의 말단 노드들 각각에 대한 수신 가중치를 수신하는 단계를 포함하는 것을 특징으로 하는 비-일시적 컴퓨터 판독가능 저장 매체.

청구항 20

제17항에 있어서,

상기 복수의 스위치들 중의 스위치 상의 상기 복수의 말단 노드들 각각에 대한 수신 가중치는, 관리자로부터의 입력으로부터 수신되는 것을 특징으로 하는 비-일시적 컴퓨터 판독가능 저장 매체.

청구항 21

삭제

청구항 22

삭제

발명의 설명

기 술 분 야

[0001] 저작권 고지(Copyright Notice)

[0002] 본 특허 문서의 개시내용의 일부분은 저작권 보호를 받는 자료를 포함한다. 저작권 소유자는 특허 및 상표 관리국의 특허 파일 또는 기록에 나타나 있는 바와 같이 본 특허 문서 또는 특허 개시내용을 임의의 사람이 복사(facsimile reproduction)하는 것에 대해 반대하지 않지만, 한편으로는 어떤 경우라도 모든 저작권 권리들을 보유한다.

[0003] 본 발명은 일반적으로 컴퓨터 시스템들에 관한 것이고, 특히 네트워크 환경에 관한 것이다.

배 경 기 술

[0004] 팻-트리 토폴로지(fat-tree topology)는, 고성능 컴퓨팅(High Performance Computing, HPC) 클러스터(cluster)들을 위해, 그리고 인피니밴드(InfiniBand™)(IB) 기술에 기반을 둔 클러스터들을 위해, 사용된다. 예를 들어, 팻-트리 토폴로지는 톈허 2호(Tianhe-2)와 같은 가장 빠른 슈퍼컴퓨터들에서 사용된다. 또한, 팻-트리 IB 시스템들은 스탬피드(Stampede), TGCC 퀴리(TGCC Curie) 및 슈퍼MUC(SuperMUC)와 같은 대규모 설치환경들을 포함한다.

[0005] 이것들은 본 발명의 실시예들이 처리하고자 의도하는 일반적인 영역들이다.

발명의 내용

[0006] 네트워크 환경(network environment)에서 트리 토폴로지(tree topology)로 정렬된 복수의 말단 노드(end node)들 및 복수의 스위치(switch)들 간에 효율적인 로드 발란싱(load balancing)을 지원하기 위한 시스템들 및 방법들이 본 명세서에서 설명된다. 본 시스템들 및 방법들은, 복수의 말단 노드들을 정렬(sort)시킬 수 있고, 여기서 복수의 말단 노드들은 트리 토폴로지에서 하나 이상의 리프 스위치(leaf switch)들 상에 있고, 복수의 말단 노드들은 수신 가중치(receive weight)가 감소하는 내림 차순(decreasing order)으로 정렬된다. 본 시스템들 및 방법들은, 복수의 말단 노드들을 수신 가중치들이 감소하는 내림 차순으로 라우팅(routing)할 수 있고, 여기서 라우팅하는 것은 적어도 하나의 하향-진행 포트(down-going port) 및 적어도 하나의 상향-진행 포트(up-going port)를 선택하는 것을 포함한다. 본 시스템들 및 방법들은, 각각의 선택된 하향-진행 포트 상의 누적 하향 가중치(accumulated downward weight)를 라우팅되는 말단 노드의 수신 가중치만큼 증가시킬 수 있다. 마지막으로, 본 시스템들 및 방법들은, 각각의 선택된 상향-진행 포트 상의 누적 상향 가중치(accumulated upward weight)를 라우팅되는 말단 노드의 수신 가중치만큼 증가시킬 수 있다.

[0007] 일 실시예에서, 본 명세서에서 설명되는 시스템들 및 방법들은, 복수의 스위치들 및 복수의 말단 노드들이 고-성능 컴퓨팅(High-Performance Computing, HPC) 클러스터(cluster)에서의 사용을 위한 팻-트리 토폴로지(fat-tree topology)를 가질 수 있게 할 수 있다. 추가적으로, 일 실시예에서, 하나 이상의 리프 스위치들 상에 있는 복수의 말단 노드들에 대한 수신 가중치들이 시스템에 의해 수신될 수 있다.

[0008] 일 실시예에서, 본 방법들 및 시스템들에 의해 수행되는 적어도 하나의 하향-진행 포트의 선택은, 복수의 하향-진행 포트들을 비교하는 것과, 그리고 가장 적은 누적 하향 가중치를 갖는 하향-진행 포트를 선택하는 것을 포함할 수 있다.

[0009] 일 실시예에서, 본 방법들 및 시스템들에 의해 수행되는 적어도 하나의 하향-진행 포트의 선택은, 복수의 하향-진행 포트들을 비교하는 것과, 그리고 가장 적은 누적 상향 가중치를 갖는 하향-진행 포트를 선택하는 것을 포함한다.

[0010] 일 실시예에서, 본 방법들 및 시스템들에 의해 수행되는 적어도 하나의 하향-진행 포트의 선택은, 복수의 하향-진행 포트들을 비교하는 것과, 그리고 가장 작은 글로벌 고유 식별자(global unique identifier)를 갖는 하향-진행 포트를 선택하는 것을 포함한다.

[0011] 일 실시예에서, 본 방법들 및 시스템들에 의해 수행되는 적어도 하나의 하향-진행 포트의 선택은, 복수의 하향-진행 포트들을 비교하는 것, 그리고 가장 적은 누적 하향 가중치를 갖는 하향-진행 포트를 선택하는 것을 포함한다. 둘 이상의 하향-진행 포트들이 가장 적은 누적 하향 가중치를 갖는 경우, 본 방법들 및 시스템들은 또한, 가장 적은 누적 하향 가중치를 갖는 둘 이상의 하향-진행 포트들을 비교할 수 있고, 그리고 가장 적은 누적 하향 가중치를 갖는 둘 이상의 하향-진행 포트들로부터, 가장 적은 누적 상향 가중치를 갖는 하향-진행 포트를 선택할 수 있다. 둘 이상의 하향-진행 포트들이 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 경우, 본 방법들 및 시스템들은, 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 둘 이

상의 하향-진행 포트들을 비교할 수 있고, 그리고 가장 적은 누적 하향 가중치 및 가장 적은 누적 상향 가중치를 갖는 둘 이상의 하향-진행 포트들로부터, 가장 작은 글로벌 고유 식별자를 갖는 하향-진행 포트를 선택할 수 있다.

도면의 간단한 설명

[0012] 도 1은 본 발명의 일 실시예가 실시될 수 있는 네트워크 환경에서의 팻-트리 라우팅의 블록도이다.

도 2는 본 발명의 일 실시예가 실시될 수 있는 네트워크 환경에서의 팻-트리 라우팅의 블록도이다.

도 3은 본 발명의 일 실시예가 실시될 수 있는 네트워크 환경에서의 팻-트리 라우팅의 블록도이다.

도 4는 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 보여주는 블록도이다.

도 5는 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 보여주는 블록도이다.

도 6은 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 보여주는 블록도이다.

도 7은 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 보여주는 블록도이다.

도 8은 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 보여주는 블록도이다.

도 9는 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 보여주는 블록도이다.

도 10은 본 발명의 일 실시예에 따른, 네트워크 환경에서 트리 토폴로지로 정렬된 복수의 말단 노드들 및 복수의 스위치들 간에 효율적인 로드 발란싱을 지원하기 위한 방법을 보여주는 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0013] 다음의 상세한 설명에서는 본 발명이 예시되는바, 이것은 첨부되는 도면들의 그림들에서 한정적 의미가 아닌 예시적 의미로 제시될 것이다. 본 개시내용에서 "임의의" 혹은 "하나의" 혹은 "일부" 실시예(들)로 언급되는 것들은 반드시 동일한 실시예를 지칭하는 것이 아니며, 이러한 언급은 "적어도 하나"의 의미를 갖고 있음에 유의해야 한다. 특정 구현예들이 논의되지만, 이러한 특정 구현예들은 오로지 예시적 목적으로 제공되는 것임을 이해해야 한다. 관련 기술분야에서 숙련된 자는 본 발명의 범위 및 사상으로부터 벗어남이 없이 다른 컴포넌트들 및 구성들이 사용될 수 있음을 인식할 것이다.

[0014] 상세한 설명 및 도면들 전반에 걸쳐 유사한 요소들을 표시하기 위해 공통의 참조 번호들이 사용되고, 따라서 도면에서 사용되는 참조 번호들은 만약 해당 요소가 다른 곳에서 설명되는 경우 이러한 도면에 특정된 상세한 설명에서 언급될 수 있거나 혹은 언급되지 않을 수 있다.

[0015] 본 발명의 다음의 설명은 고성능 네트워크에 대한 예로서 인피니밴드(InfiniBand™)(IB)를 사용한다. 이에 한정됨이 없이 고성능 네트워크들의 다른 타입들이 사용될 수 있음은 본 발명의 기술분야에서 숙련된 자들에게 명백할 것이다. 다음의 설명은 또한 패브릭 토폴로지(fabric topology)에 대한 예로서 팻-트리 토폴로지를 사용한다. 이에 한정됨이 없이 패브릭 토폴로지들의 다른 타입들이 사용될 수 있음은 본 발명의 기술분야에서 숙련된 자들에게 명백할 것이다.

[0016] **인피니밴드(InfiniBand™)**

[0017] 인피니밴드(InfiniBand™)(IB)는 인피니밴드 동업 협회(InfiniBand™ Trade Association, IBTA)에 의해 개발된 개방형 표준 무손실 네트워크 기술(open standard lossless network technology)이다. 이러한 기술은 높은 처리량(high throughput) 및 낮은 레이턴시(low latency) 통신을 제공하는 (특히 HPC 애플리케이션들 및 데이터센터들에 맞도록 설계된) 직렬 포인트-투-포인트 풀-듀플렉스 상호연결(serial point-to-point full-duplex interconnect)에 기반을 두고 있다.

[0018] 인피니밴드 아키텍처(InfiniBand™ Architecture, IBA)는 2-계층 토폴로지 디비젼(two-layer topological division)을 지원한다. 하위-계층에서, IB 네트워크들은 서브넷(subnet)들로서 지칭되고, 여기서 서브넷은 스위치(switch)들 및 포인트-투-포인트 링크(point-to-point link)들을 사용하여 상호연결되는 호스트(host)들의 세트를 포함할 수 있다. 상위 레벨에서, IB 패브릭은 하나 이상의 서브넷들을 구성하고, 이들은 라우터(router)들을 사용하여 상호연결될 수 있다.

[0019] 서브넷 내에서, 호스트들은 스위치들 및 포인트-투-포인트 링크들을 사용하여 연결된다. 추가적으로, 하나의 마

스터 관리 엔티티(master management entity), 서브넷 매니저(Subnet Manager, SM)가 존재하고, 이것은 서브넷 내의 지정된 서브넷 디바이스 상에 상주한다. 서브넷 매니저는 IB 서브넷을 구성하는 것, 활성화시키는 것, 그리고 유지시키는 것에 대한 책임을 지고 있다. 추가적으로, 서브넷 매니저(SM)는 IB 패브릭 내의 라우팅 테이블 계산(routing table calculation)들을 수행하는 것에 대한 책임을 지고 있을 수 있다. 여기서, 예를 들어, IB 네트워크의 라우팅은 로컬 서브넷 내에서 모든 소스 및 목적지 쌍(source and destination pair)들 간의 적절한 로드 발란싱을 목표로 한다.

[0020] 서브넷 관리 인터페이스(subnet management interface)를 통해, 서브넷 매니저는 제어 패킷(control packet)들(이것은 서브넷 관리 패킷(Subnet Management Packet, SMP)들로서 지칭됨)을 서브넷 관리 에이전트(Subnet Management Agent, SMA)들과 교환한다. 서브넷 관리 에이전트들은 모든 IB 서브넷 디바이스 상에 상주한다. SMP들을 사용함으로써, 서브넷 매니저는 패브릭을 발견할 수 있고, 말단 노드들 및 스위치들을 구성할 수 있고, 그리고 SMA들로부터 통지(notification)들을 수신할 수 있다.

[0021] 일반적으로, 마스터 서브넷 매니저(master subnet manager)를 제외한 다른 모든 서브넷 매니저들은 오류-허용(fault-tolerance)을 위해서 대기 모드(standby mode)에서 동작한다. 그러나, 마스터 서브넷 매니저가 고장이 난 상황에서, 대기상태의 서브넷 매니저들은 새로운 마스터 서브넷 매니저를 협상한다. 마스터 서브넷 매니저는 또한, 임의의 토폴로지 변화들을 검출하기 위해 그리고 이에 따라 네트워크를 재구성하기 위해, 서브넷의 주기적인 스위프(sweep)들을 수행한다.

[0022] 더욱이, 서브넷 내의 호스트들 및 스위치들은 로컬 식별자(Local IDdentifier, LID)들을 사용하여 어드레싱(addressing)될 수 있으며, 단일의 서브넷은 49151개의 LID들로 제한될 수 있다. 서브넷 내에서 유효한 로컬 어드레스들인 LID들 외에도, 각각의 IB 디바이스는 자신의 비-휘발성 메모리에 버닝(burning)되는 64-비트 글로벌 고유 식별자(Global Unique IDentifier, GUID)를 가질 수 있다. GUID는 IB 계층 3(Layer 3, L3) 어드레스인 글로벌 식별자(Global IDentifier, GID)를 형성하기 위해 사용될 수 있다. IPv6과-같은 128-비트 어드레스(IPv6-like 128-bit address)를 형성하도록 64-비트 서브넷 식별자(ID)를 64-비트 GUID와 결부(concatenating)시킴으로써 GID가 생성될 수 있다. 예를 들어, 상이한 포트 GUID들이 IB 패브릭에 연결된 포트들에 할당될 수 있다.

[0023] SM은 네트워크 초기화 시간에 라우팅 테이블들(즉, 트리 내의 노드들의 각각의 쌍들 간의 연결들/루트들(connections/routes))을 계산할 수 있다. 더욱이, 라우팅 테이블들은 최적의 성능을 보장하기 위해 토폴로지가 변경될 때마다 업데이트(update)될 수 있다. 보통의 동작들 동안, SM은 토폴로지 변경들을 점검하기 위해 네트워크의 주기적인 라이트 스위프(light sweep)들을 수행할 수 있다. 만약 라이트 스위프 동안 임의의 변경이 발견되면, 혹은 만약 네트워크 변경을 시그널링(signaling)하는 메시지(message)(트랩(trap))를 SM이 수신한다면, SM은 그 발견된 변경들에 따라 네트워크를 재구성할 수 있다.

[0024] 예를 들어, SM은 네트워크 토폴로지가 변경될 때, 예를 들어, 링크(link)가 하향(down)으로 진행할 때, 혹은 디바이스가 추가될 때, 혹은 링크가 제거될 때, 네트워크를 재구성할 수 있다. 이러한 재구성 단계는 네트워크 초기화 동안 수행되는 단계들을 포함할 수 있다. 더욱이, 재구성들은 (네트워크 변경들이 일어난) 서브넷들에 한정되는 로컬 범위(local scope)를 가질 수 있다. 또한, 라우터들을 갖는 커다란 패브릭의 세그먼트화(segmenting)는 재구성 범위를 한정할 수 있다.

[0025] **팻-트리 라우팅(Fat-tree Routing)**

[0026] 팻-트리 토폴로지는 범용 네트워크 토폴로지들의 스케일링가능한 클래스(scalable class)이다. 팻-트리 토폴로지를 지지하는 초기 개념은 네트워크 토폴로지를 말단 노드들이 리프 스위치들에 상주하는 스위치들의 계층화된 다중-루트 트리 구조(layered, multi-rooted tree structure)로서 정렬하는 것이었다. 팻-트리의 루트(root)들을 향해 점점 더 비대해지는 링크들을 사용함으로써, 전체 이분화 대역폭(full bisection bandwidth)이 유지될 수 있고, 잠재적으로 혼잡(congestion)이 일어나지 않는다. 추가적으로 이것은 이용가능하게 된 임의의 대역폭을 사용하는 혜택을 제공할 수 있다.

[0027] 팻-트리 토폴로지는 예를 들어, HPC 환경들 내에서, 고성능 상호연결들을 지원하기 위한 다양한 혜택들을 제공할 수 있다. 이러한 혜택들은 데드락 없음(deadlock freedom), 내재적 오류-허용(inherent fault-tolerance), 및 전체 이분화 대역폭(full bisection bandwidth)을 포함할 수 있다. 데드락 없음은 트리 구조의 사용이 데드락 회피를 위한 특별한 고려들 없이 팻-트리들을 라우팅하는 것을 가능하게 함을 나타낸다. 개개의 소스 목적지 쌍들 간의 다중 경로들의 존재로 인한 내재적 오류-허용은 네트워크 오류들의 효율적인 핸들링을 가능하게 한다. 전체 이분화 대역폭은 네트워크로 하여금 네트워크의 두 개의 절반들 간의 전체 속도 통신(full speed

communication)을 유지할 수 있게 한다.

[0028] 팻-트리 라우팅 알고리즘들은 네트워크 패브릭 내의 링크들에 걸쳐 가장 짧은 경로의 루트들을 균등하게 분산(spread)시키는 선형 포워딩 테이블(Linear Forwarding Table, LFT)들을 발생시키는 것을 목표로 할 수 있다. 이러한 알고리즘은 인덱싱 순서(indexing order)에서 패브릭을 이동(traverse)할 수 있고, 말단 노드들의 목표 LID들을, 그리고 이에 따른 대응하는 루트들을 각각의 스위치 포트에 할당할 수 있다.

[0029] 더욱이, 팻-트리 라우팅 알고리즘들은 기반이 되는 팻-트리 토폴로지의 효율적인 사용을 지원하기 위해 사용될 수 있다. 아래와 같은 알고리즘 1(Algorithm 1)이 예시적인 팻-트리 라우팅 알고리즘이다.

[0030] ---

[0031] **Algorithm 1** *route_to_cns() function*

[0032] ---

[0033] **Require:** Addressing is completed

[0034] **Ensure:** All *hca_ports* are routed

[0035] 1: **for** *swleaf = 0* **to** *max_leaf_sw* **do**

[0036] 2: **for** *swleaf.port = 0* **to** max_ports **do**

[0037] 3: *hca_lid = swleaf.port-> remote_lid*

[0038] 4: *swleaf.routing table[hca_lid]* = swleaf.port

[0039] 5: *route_downgoing_by_going_ up()*

[0040] 6: **end for**

[0041] 7: **end for**

[0042] 위에서 제시되는 바와 같이, 라우팅 함수(routing function) route_to_cns()는 리프 스위치들의 어레이에 관해 반복될 수 있다(라인 1(Line 1) 내지 라인 7(Line 7)). 각각의 선택된 리프 스위치에 대해, 라우팅 함수는 그 선택된 리프 스위치에 연결된 각각의 말단-노드 포트를 라우팅시킬 수 있다(예를 들어, 포트 번호부여 시퀀스(port numbering sequence)에서)(라인 2 내지 라인 6).

[0043] 더욱이, 특정 LID와 관련된 말단-노드 포트를 라우팅할 때, 라우팅 함수는 하향-진행 경로들을 라우팅하기 위해 네트워크 토폴로지 내에서 한 레벨 상향으로 진행할 수 있고, 그리고 각각의 스위치 포트를 라우팅할 때, 라우팅 함수는 상향진행 경로들을 라우팅하기 위해 하향으로 진행할 수 있다. 이러한 프로세스는 루트 스위치 레벨(root switch level)에 도달할 때까지 반복될 수 있다. 이후 모든 노드들을 향하는 경로들이 라우팅될 수 있고 패브릭 내의 모든 스위치들의 선형 포워딩 테이블(LFT)들 내에 삽입될 수 있다.

[0044] 예를 들어, route_downgoing_by_going_up() 함수(라인 5)는, 경로들을 발란싱할 수 있는 그리고 route_upgoing_by going_down() 함수를 호출할 수 있는, 반복 함수(recurrence function)일 수 있고, 이것은 route_downgoing_by_going_up() 함수가 호출된 스위치를 통해 목적지를 향해 팻-트리 내에서 상향 경로들을 라우팅한다.

[0045] route_to_cns() 함수와 관련된 몇 가지 잠재적인 단점들이 존재할 수 있다. 첫째, route_to_cns() 함수는 비의식적(oblivious)이고, 말단-포트들이 어떤 말단-노드에 속하는지에 관한 어떠한 고려도 없이 말단-포트들을 라우팅한다. 둘째, route_to_cns() 함수는 라우팅을 위한 물리적 포트 개수에 따라 달라진다.

[0046] 도 1은 본 개시내용의 일 실시예가 실시될 수 있는 네트워크 환경에서의 팻-트리 라우팅의 예시를 보여준다. 도 1에서 제시되는 바와 같이, 하나 이상의 말단 노드들(101 내지 104)이 네트워크 패브릭(100) 내에서 연결될 수 있다. 네트워크 패브릭(100)은 팻-트리 토폴로지에 근거할 수 있고, 이것은 복수의 리프 스위치들(111 내지 114), 그리고 다수의 스파인 스위치(spine switch)들 혹은 루트 스위치(root switch)들(131 내지 134)을 포함한다. 추가적으로, 네트워크 패브릭(100)은 스위치들(121 내지 124)과 같은 하나 이상의 중간 스위치들을 포함할 수 있다.

[0047] 또한 도 1에서 제시되는 바와 같이, 말단 노드들(101 내지 104) 각각은 다중-홈 노드(multi-homed node)일 수

있는바, 즉 다수의 포트들을 통해 네트워크 패브릭(100)의 둘 이상의 부분들에 연결된 단일 노드일 수 있다. 예를 들어, 노드(101)는 포트(H1) 및 포트(H2)를 포함할 수 있고, 노드(102)는 포트(H3) 및 포트(H4)를 포함할 수 있고, 노드(103)는 포트(H5) 및 포트(H6)를 포함할 수 있고, 그리고 노드(104)는 포트(H7) 및 포트(H8)를 포함할 수 있다.

[0048] 추가적으로, 각각의 스위치는 다수의 스위치 포트들을 가질 수 있다. 예를 들어, 루트 스위치(131)는 스위치 포트(1) 내지 스위치 포트(2)를 가질 수 있고, 루트 스위치(132)는 스위치 포트(3) 내지 스위치 포트(4)를 가질 수 있고, 루트 스위치(133)는 스위치 포트(5) 내지 스위치 포트(6)를 가질 수 있고, 그리고 루트 스위치(134)는 스위치 포트(7) 내지 스위치 포트(8)를 가질 수 있다.

[0049] 도 2는 본 개시내용의 일 실시예가 실시될 수 있는 네트워크 환경(200)에서의 팻-트리 라우팅의 예시를 보여준다. 도 2는 k-진-n-트리(k-ary-n-tree)를 도시하는데, k-진-n-트리는 n-레벨의 팻-트리이며 k^n개의 말단 노드들을 갖고 그리고 $n \times k^{n-1}$개의 스위치들을 가지며 스위치들 각각은 2k개의 포트들을 갖는다. 더 구체적으로 살펴보면, 도 2는 4-진-2-트리를 도시하는바, 즉 팻-트리 토폴로지가 2개의 레벨들을 갖고, 16개의 말단 노드들(201 내지 216)을 갖고, 그리고 8개 스위치들(네 개의 리프 스위치들(220 내지 223) 및 네 개의 루트 스위치들(225 내지 228))을 갖고, 각각의 스위치는 8개의 포트들을 갖는다.

[0050] 레거시 팻-트리 라우팅 알고리즘(legacy fat-tree routing algorithm)(이것은 본 명세서에서 에프트리(FTree)로서 다양하게 지칭됨)은 네트워크 패브릭 내의 링크들에 걸쳐 가장 짧은 경로의 루트들을 균등하게 분산시키는 LFT들을 발생시키는 것을 목표로 한다. 이러한 알고리즘은 일반적으로 인덱싱 순서에서 패브릭을 이동하고, 말단 노드들의 목표 LID들을, 그리고 이에 따른 대응하는 루트들을 각각의 스위치 포트에 할당한다. 동일한 리프 스위치에 연결된 말단 노드들에 대해, 인덱싱 순서는 말단 노드가 연결된 스위치 포트에 따라 달라진다(포트 번호부여 시퀀스). 각각의 포트에 대해, 알고리즘은 포트 사용 카운터(port usage counter)를 유지할 수 있고, 그리고 (만약 하나보다 더 많은 옵션(option)이 이용가능하다면) 루트가 추가될 때마다 가장 적게 사용된 포트를 선택하기 위해 이것을 사용할 수 있다. 만약 동일한 두 개의 스위치들에 연결되는 다수의 포트들이 존재한다면, 이러한 포트들은 포트 그룹(port group)을 형성한다. 그러한 경우에, 가장 적게 로딩(loading)된 포트 그룹의 가장 적게 사용된 포트가 새로운 루트에 추가되도록 선택된다.

[0051] 일반적으로, LID들에 대한 포트 할당은 리프 스위치들에서 시작하여 두 개의 스테이지(stage)들에서 반복적으로 수행된다. 제 1 스테이지에서, 알고리즘은 아래로 이동하여 각각의 말단 노드로부터 위에 있는 트리 루트를 향해 이동하고, 하향-진행 포트를 LID에 할당한다. 하향-진행 포트들이 설정된 이후, 알고리즘은 트리를 내려감으로써 모든 연결된 하향 스위치들 상의 LID에게 상향 포트들을 할당한다. 그 다음에 이러한 프로세스는 트리의 다음 레벨까지 움직임으로써 반복적으로 되풀이된다.

[0052] 팻-트리 토폴로지에 대한 레거시 라우팅 메커니즘(즉, 에프트리 알고리즘)과 관련된 두 개의 단점들이 존재한다.

[0053] 첫째, 팻-트리 토폴로지에 대한 표준 알고리즘에 의해 사용되는 로드-발란싱 기법은 노드들의 트래픽 특징(traffic characteristic)들 중 그 어느 것도 고려함이 없이 토폴로지 내의 링크들에 걸쳐 로드를 발란싱하려고 한다. 달리 말하면, 레거시 팻-트리 알고리즘은 네트워크 내의 모든 노드들에 대해 동일한 가중치를 가정한다. 그러나, HPC 클러스터들에서, 상이한 노드들은 종종 이들의 트래픽 프로파일(traffic profile)들을 결정하는 역할(role)들을 미리할당한다. 예를 들어, 저장 노드들 혹은 I/O 게이트웨이(gateway)들은 다른 노드들보다 더 많은 트래픽을 소비할 가능성이 높다. 따라서, 이러한 높은 트래픽 노드들을 향하는 루트들은 더 혼잡하게 될 가능성이 높고 네트워크 내에서 우선권(priority)을 필요로 할 가능성이 높다. 특정 노드들의 트래픽 필요들을 고려함이 없이 라우팅이 행해지는 경우, 이것은 결과적으로 일부 링크들은 과다가입(oversubscribe)되는 반면 다른 링크들은 과소이용(underutilize)됨에 따라, 최적화되지 못한 네트워크 처리량이 일어나게 할 수 있다.

[0054] 둘째, 팻-트리 토폴로지에 대한 레거시 알고리즘은 이것이 결과적으로 비-예측가능 성능을 일으킬 수 있기 때문에 바람직하지 않다. 이러한 비-예측가능 성능은 알고리즘이 인덱싱 순서에 따라 루트들을 링크들에 할당하기 때문에 일어나는 결과이다. 그러나, 인덱싱 순서는 구성가능하지 않고, 말단 노드들이 연결되는 리프 스위치들의 포트 개수들에 따라 달라진다. 이로 인해, 동일한 방식으로 케이블링(cabling)된 팻-트리 시스템들은 상이한 그리고 비-예측가능한 성능을 나타낼 수 있다. 예로서, 2-레벨 팻-트리에서, 만약 상이한 리프 스위치들에서의 두 개의 말단 노드들이, 동일한 인덱스 위치를 공유한다면, 이러한 두 개의 노드들을 향하는 트래픽은 동일한 루트 스위치를 통해 라우팅될 것이다. 결과적으로, 다른 리프 스위치들에서의 말단 노드들로부터 나왔지만 이러

한 두 개의 노드들을 향하고 있는 모든 트래픽은, 대안적 루트 스위치들을 통한 수 개의 덜 로딩된 경로들이 존재할 수 있을지라도, 단일 루트 스위치에 연결된 상향 링크들의 공통 세트에 대한 액세스를 위해 경쟁할 것이다.

[0055] 레거시 팻-트리 라우팅과 관련된 문제들을 더 잘 예시하기 위해, 도 2에서의 예의 라우팅을 고려하는 것이 도움을 준다. 도 2에서, 네 개의 수신기 노드들을 나타내기 위해(혹은 전체 네트워크 트래픽 중 많은 부분을 수신하는 것으로 알려진 노드들을 나타내기 위해) 노드들(201, 206, 210, 및 213)이 음영화(shade)되어 있다. 네 개의 리프 스위치들(220 내지 223) 각각은 네 개의 루트 스위치들(225 내지 228)에 연결된다. 노드들이 좌측으로부터 우측으로 인덱싱 순서로 있다고 가정하면(즉, 노드(201)는 인덱싱 순서 1을 가지고, 노드(206)는 인덱싱 순서 2를 가지고, 노드(210)는 인덱싱 순서 2를 가지고, 그리고 노드(213)는 인덱싱 순서 1을 갖는다고 가정하면), 이것은 노드(206)와 노드(213)가, 동일한 인덱싱 순서(즉, 1)를 공유함, 그리고 마찬가지로 노드(206)와 노드(210)가, 동일한 인덱싱 순서(즉, 2)를 공유함을 의미한다. 이것의 결과로서, 팻 트리-라우팅 알고리즘은 단지 두 개의 가장 좌측에 있는 루트 스위치들(225 및 226)만을 이용하여 이러한 네 개의 말단 노드들을 향해 트래픽을 라우팅할 것이다. 이것은 결과적으로 상향 방향에서 네 개의 잠재적으로 과다가입되는 링크들이 발생되게 한다(도 2에서 점선 라인(dashed line)들에 의해 보여짐). 도 2에서 "Up{a, b}"로서 라벨링(labelling)된 점선 라인들은 수신기 노드들 a 및 b를 향하는 상향 흐름이 해당 링크들 상의 대역폭에 대해 경쟁할 것임을 표시하기 위한 것이다.

[0056] 예로서, 수신기 노드(201 및 213)를 향하는 트래픽 흐름들 간의 간섭을 피하기 위해 토폴로지 내에서 이용가능한 링크들이 충분히 존재할지라도, 레거시 팻-트리 알고리즘은 여전히, 노드들(206 및 210)에 대한 두 개의 독립된 흐름들이 가장 좌측에 있는 리프 스위치(220)로부터의 동일한 상향 링크를 공유하게 할 것이다.

[0057] **k-진-n-트리들에 대한 인덱스 충돌 확률(Index Collision Probability for k-ary-n-trees)**

[0058] 앞에서 논의된 바와 같이, 만약 네트워크 내에서의 수신기 노드들(즉, 시스템 내에서의 트래픽 중 많은 부분을 고려하는 노드들)이 자신들의 각각의 리프 스위치들에서 인덱스 위치들을 공유한다면 에프트리의 성능은 저하(degrade)될 수 있다. 예를 들어, 도 2에서, 수신기 노드(201) 및 수신기 노드(213)가 인덱싱 위치 1을 공유하고, 수신기 노드(206) 및 수신기 노드(210)가 인덱싱 위치 2를 공유한다. 이로 인해, 로드-발란싱에 이르게 될 때 에프트리의 실현가능성(viability)을 평가하기 위해 이러한 인덱스 충돌(즉, 수신기 노드들이, 상이한 리프 스위치들에서, 동일한 인덱스 위치를 공유할 때)의 확률을 결정하는 것이 중요한다.

[0059] k-진-n-트리에 대해 상기하면, k-진-n-트리는 n-레벨의 트리이며 k^n개의 말단 노드들을 갖고 그리고 $n \times k^{n-1}$개의 스위치들을 가지며 스위치들 각각은 2k개의 포트들을 갖는다. 말단 노드들을 가지며 레벨 l=n인 완전히 채워진 트리를 가정하면,

[0060] • 각각의 말단 노드는 n-듀플(n-tuple) $\{0, 1, \ldots, k-1\}^n$에 의해 나타내어지고, 그리고 각각의 스위치는 순서 쌍 <s,l>에 의해 나타내어지며, 여기서 $s \in \{0, 1, \ldots, k-1\}^{n-1}$이고 레벨 $l \in = \{0, 1, \ldots, n-1\}$이며,

[0061] • 리프 스위치는 레벨 n에서 말단 노드들 $c_0, c_1, \ldots, c_{n-1}$에 대한 에지(edge)를 갖는 레벨 n-1 스위치 $<l_0, l_1, \ldots, l_{n-2}, n-1>$로서 정의된다.

[0062] y개의 노드들이 존재하고 이러한 y개의 노드들 각각은 각각의 리프 스위치에서의 k개의 말단 노드들 중에서 네트워크 내에서의 트래픽 수신 중 더 높은 비율을 갖는 그러한 노드인 상황에서(예를 들어, 수신기 노드들), 수신기 노드가 리프 스위치에서의 임의의 인덱스 위치 i에서 발견될 확률은 아래의 공식에 의해 주어진다.

[0063] $$p_i = \frac{y}{k}$$

[0064] 팻-트리가 $N = k^{n-1}$개의 리프 스위치들을 가지기 때문에, 수신기 노드들이 그들의 대응하는 스위치들에서 동일한 인덱스 위치를 공유할 확률을 찾는데 이항 분포(binomial distribution)가 사용될 수 있다. p_i의 확률을 갖는 임의의 인덱스 위치 i에서 정확히 r개의 수신기 노드들을 발견할 확률은 아래의 식에 의해 주어진다.

[0065] $$f(r, N, p_i) = \binom{N}{r} p_i^r (1 - p_i)^{N-r}$$

[0066] 위치 i에서 적어도 x개의 인덱스 충돌들을 얻을 확률을 계산하기 위해 여기에 제시된 바와 같은 모든 대응하는 해당 확률들의 합(sum)이 아래와 같이 취해진다.

[0067] $$f'(x,N,p_i) = \sum_{j=x+1}^{N} f(j,N,p_i), x < N$$

[0068] 각각의 리프 스위치에서의 R개의 연결된 말단 노드들을 갖는 팻-트리에 대해, R개의 위치들 중 임의의 위치에서의 인덱스 충돌은(i ∈ = {1, 2, 3, ..., R}) 결과적으로 네트워크 경합(network contention)이 증가되게 할 수 있음에 유의해야 한다.

[0069] **가중치 부여된 팻-트리 라우팅 알고리즘(Weighted Fat-Tree Routing Algorithm)**

[0070] 본 개시내용의 일 실시예에 따르면, 가중치 부여된 팻-트리 라우팅 알고리즘(이것은 본 명세서 전반에 걸쳐 더블유팻트리(wFatTree)로서 다양하게 지칭됨)은 앞서의 에프트리의 결점들을 극복하기 위해 사용된다. 더블유팻트리 내에서, 각각의 말단 노드에는 새로운 파라미터 *receive_weight*가 할당되는데, 이것은 시스템 내에서 루트들을 계산할 때, 알려진 혹은 학습된 트래픽 특징들을 고려하기 위해 사용될 수 있다.

[0071] 일 실시예에서, 각각의 말단 노드에 대한 *receive_weight* 파라미터의 값은, 라우팅 테이블을 계산할 때, 노드 수신을 향하는 흐름들의 우선권의 정도를 반영한다. 일 예로서, 범위 [1, 100] 내의 말단 노드에게 가중치들을 할당하도록 하는 구성이 존재할 수 있다. 각각의 노드는, 네트워크 내에서 노드가 얼마나 많은 트래픽을 수신하는 것으로 알려져 있는지에 따라 가중치를 수신하게 된다. 이러한 예에서, 말단 노드는 그 값이 1인 *receive_weight*를 할당받을 수 있다. 이것은 매우 적은 트래픽을 수신하는 노드(트래픽 발생 노드)를 나타내게 된다. 추가적으로, 링크 용량(link capacity)에 가까운 트래픽을 수신하는 말단 노드는 그 값이 100인 *receive_weight*를 할당받을 수 있다. 이 경우, 이러한 상황에서, 1과 100 사이에 있는 *receive_weight*의 값들은 노드가 네트워크 내에서 수신하는 트래픽의 비율을 나타내게 된다.

[0072] 또 하나의 다른 실시예에서, 노드는 그 값이 500인 *receive_weight*를 수신할 수 있고, 반면 네트워크 내의 다른 모든 노드들은 그 값이 1인 *receive_weight*를 부여받는다. 이것은 그 값이 500인 *receive_weight*를 갖는 말단 노드가 중요 노드(critical node)이며 이러한 중요 노드를 향해 흐르는 트래픽은 우선화(prioritize)돼야만 함을 표시하게 된다.

[0073] 일 실시예에서, 더블유팻트리 라우팅 알고리즘(아래에서 알고리즘 2로 보여짐)은 세 개의 국면(phase)들에서 반복적으로 작동한다. 이러한 실시예에서, 모든 루트들은 역행으로(backwards) 계산되는데, 목적지 노드에서 시작하여 역방향으로 작동한다. 아래와 같은 알고리즘 2(Algorithm 2)가 예시적인 더블유팻트리 라우팅 알고리즘(wFatTree routing algorithm)이다.

[0074] ---

[0075] **Algorithm 2** A wFatTree routing algorithm

[0076] ---

```
[0077] 1: procedure ROUTETOENDNODES
[0078] 2:     for all s ∈ leafSwitches[] do
[0079] 3:         sort end nodes connected to s by rcv_weight
[0080] 4:         for all c ∈ endNodes[] do
[0081] 5:             s.LFT[c.LID]←c.port
[0082] 6:             ROUTEDOWNGOINGBYASC(s, c)
[0083] 7:         end for
[0084] 8:     end for
[0085] 9: end procedure
[0086] 10: procedure ROUTEDOWNGOINGBYASC(s, c)
```

[0087] 11: *p* ← GETLEASTLOADEDPORT(*s*, *UpGroups*[])

[0088] 12: *rSwitch* ← *p.Switch*

[0089] 13: *rSwitch.LFT[c.LID]* ← *p*

[0090] 14: *p.Dwn* += *c.rcv_weight*

[0091] 15: ROUTEUPGOINGBYDESC(*s*, *c*)

[0092] 16: ROUTEDOWNGOINGBYASC(*rSwitch*, *c*)

[0093] 17: **end procedure**

[0094] 18: **procedure** ROUTEUPGOINGBYDESC(*s*, *c*)

[0095] 19: **for all** *g* ∈ *DownGroups*[] **do**

[0096] 20: *skip g if the LFT(c.LID) is part of this group*

[0097] 21: *p* ← GETLEASTLOADEDPORT(g)

[0098] 22: *rSwitch* ← *p.Switch*

[0099] 23: *rSwitch.LFT[c.LID]* ← *p*

[0100] 24: *p.Up* += *c.rcv_weight*

[0101] 25: ROUTEUPGOINGBYDESC(*rSwitch*, *c*)

[0102] 26: **end for**

[0103] 27: **end procedure**

[0104] 일 실시예에서, 예시적인 알고리즘 2의 제 1 국면 동안, 각각의 리프 스위치에서의 말단 노드들은 *receive_weights* 감소에 따라 정렬된다(라인 3). (알고리즘 2에서 *receive_weight*는 "*rcv_weight*"에 의해 약어로 표시되었음에 유의). 이전에 언급된 바와 같이, *receive_weights*는 관리자에 의해 공급될 수 있거나, 혹은 이들은 또한 계산될 수 있다. 이 주제에 관한 추가 논의는 이후 나타난다.

[0105] 일 실시예에서, 예시적인 알고리즘 2의 제 2 국면 동안, 더블유팻트리는 각각의 말단 노드(예를 들어, 목적지 노드 혹은 루트의 목적지)로부터 트리 상향으로 이동하고, 다음 레벨에서의 선택된 스위치에서 현재 노드에 대한 하향-진행 포트를 할당한다(ROUTEDOWGOINGBYASC, 예시적인 알고리즘 2의 라인 6). 하향-진행 포트가 선택되는 경우, 알고리즘은 대응하는 포트에 대한 누적 하향 가중치를 라우팅된 말단 노드의 *receive_weight*만큼 증가시킨다(라인 14). 이것은 새로운 (가중치 부여된) 루트가, 대응하는 포트에 추가되었음을 표시한다.

[0106] 일 실시예에서, 하향-진행 포트가 설정된 이후, 예시적인 알고리즘 2의 제 3 국면에서, 알고리즘은 트리를 내려감으로써 모든 연결된 하향 스위치들 상에서 말단 노드를 향하는 루트들에 대한 상향 포트들을 할당한다(그리고 라우팅된 말단 노드들의 receive_weight를 추가함으로써 포트들에 대한 대응하는 상향 가중치들을 업데이트함)(ROUTEUPGOIGNBYDESC). 그 다음에 전체 3-국면 프로세스는 트리 내에서 다음 레벨로 올라감으로써 반복된다(라인 16).

[0107] 일 실시예에서, 더블유팻트리 알고리즘인 임의의 알고리즘은 각각의 루트 계산에 대해 가장 적게 로딩된 포트를 선택한다. 선택 기준들은 먼저 하향 가중치에 근거한다. 두 개의 포트들이 동등한 하향 가중치들을 갖는 상황에서는, 가장 적은 하향 가중치를 갖는 포트가 선택된다. 추가적으로, 하향 가중치와 상향 가중치가 모두 동등한 상황에서, 알고리즘은 프로세스의 결정성(deterministic)을 유지하기 위해 가장 작은 GUID를 갖는 포트를 선택한다. 아래의 예시적인 알고리즘 3(Algorithm 3)은 더블유팻트리가 어떻게 각각의 루트 계산에 대해 가장 적게 로딩된 포트를 선택하는지를 보여준다.

[0108] ———————————————————————————————

[0109] **Algorithm 3** Get least-loaded port of all port groups

[0110] ———————————————————————————————

[0111]
```
1: procedure GETLEASLOADEDPORT(PortGroups[])
```

[0112]
```
2:   min ← null
```

[0113]
```
3:   for all g ∈ PortGroups[] do
```

[0114]
```
4:          for all port ∈ g.Ports[] do
```

[0115]
```
5:                  if port.Dwn < min.Dwn then
```

[0116]
```
6:                          min ← port
```

[0117]
```
7:                  else if port.Dwn  = min.Dwn then
```

[0118]
```
8:                          if port.Up < min.Up then
```

[0119]
```
9:                                  min ← port
```

[0120]
```
10:                         else if port.Up = min.Up
```

[0121]
```
11:                                 if port.GUID < min.GUID then
```

[0122]
```
12:                                         min ← port
```

[0123]
```
13:                                 end if
```

[0124]
```
14:                         end if
```

[0125]
```
15:                 end if
```

[0126]
```
16:         end for
```

[0127]
```
17:  end for
```

[0128]
```
18: end procedure
```

[0129] 일 실시예에서, 더블유팻트리는 수 가지 방식들로 레거시 에프트리 라우팅 알고리즘에 관해 향상을 제공한다. 첫째, 앞에서 언급된 바와 같이 리프 스위치에서의 각각의 노드가 인덱싱되어 있고 네트워크 내의 노드들의 인덱싱에 기반을 두고 있는 에프트리와는 달리, 더블유팻트리는 수신 가중치들의 감소 순으로 노드들을 라우팅한다. 이것은 노드들, 예를 들어, 수신기 노드들(예를 들어, 시스템 내에서의 트래픽 중 많은 비율을 갖는 노드들)이 먼저 라우팅될 수 있게 한다. 추가적으로, 임의의 스위치에서의 하향 포트가 임의의 말단 노드에 할당되는 상황에서, 더블유팻트리는 그 노드와 관련된 다른 로컬 링크들 상의 상향 가중치를 업데이트한다. 이것은, 상향 링크들이 잠재적으로 그 노드를 향해 트래픽을 운반하기 때문에, 링크들의 선택시 상향 가중치가 고려될 수 있게 한다. 마지막으로, 가장 적게 사용된 하향 포트가 선택되는 상황에서, 하향 가중치를 점검한 이후, 더블유팻트리는 또한 가장 적게 다투어진 포트를 선택하기 위해 할당된 상향 가중치를 점검한다. 이것은 단지 하향 방향으로 라우팅된 링크들의 개수를 점검하는 레거시 에프트리보다 향상된 로드-발란싱의 혜택을 제공한다. 하향 링크들의 개수가 동일한 것으로 발견한 경우, 레거시 에프트리는 라우팅을 결정하기 위해 인덱싱의 순서로 복귀(revert)한다.

[0130] 이제 도 3을 참조하면, 도 3은 본 개시내용의 일 실시예가 실시될 수 있는 네트워크 환경에서의 팻-트리 라우팅을 도시한다. 네트워크 환경(300)은 k-진-n-트리로서 도시되어 있고, k-진-n-트리는 n-레벨의 팻-트리이며 k^n개의 말단 노드들을 갖고 그리고 $n \times k^{n-1}$개의 스위치들을 가지며 스위치들 각각은 2k개의 포트들을 갖는다. 도 2와 유사하게, 두드러지게 표현된 노드들(301, 306, 310 및 313)은 수신기 노드들로서 지정되어 있는데, 왜냐하면 예를 들어, 이러한 노드들은 전체 네트워크 트래픽 중 많은 부분을 운반하기 때문이다. 도 3에 도시된 실시예에서는, 더블유팻트리 라우팅 알고리즘이 사용된다. 더블유팻트리 라우팅 알고리즘의 결과로서, 네트워크 환경은 이제 루트들을 계산할 때 각각의 노드의 *receive_weight*를 고려한다.

[0131] 일 실시예의 경우, 도 3에서 제시되는 바와 같이, 수신기 노드(301)로 흐르는 상향 트래픽(이러한 상향 트래픽은 방향을 표시하는 안이 채워진 화살표를 갖는 점선으로 나타나 있음)은 전부 루트 스위치(325)를 통과한다. 그 다음에, 수신기 노드(301)에 대한 하향 루트(이것은 안이 채워진 화살표를 갖는 실선으로 나타나 있음)는 루트 스위치(325)로부터 리프 스위치(320)를 통해 흐른다. 유사하게, 수신기 노드(313)를 목적지로 하는 상향 트

래픽(이러한 상향 트래픽은 방향을 표시하는 안이 비어있는 화살표들을 갖는 점선으로 나타나 있음)은 리프 스위치(323)로 하향 라우팅되기 전에 전부 루트 스위치(328)를 통과한다. 일 실시예에서, 수신기 노드들(306 및 310)로 흐르는 트래픽에 대해 유사한 트래픽 패턴들이 또한 존재한다.

[0132] 도 3에서 이용된 더블유팻트리 라우팅 알고리즘은 네트워크 환경(300) 내에서의 이용가능한 링크들에 대한 향상된 분포를 보여준다. 이것은 레거시 에프트리를 사용하는 네트워크보다 네트워크 내에서의 성능을 향상시킬 수 있다.

[0133] 일 실시예에서, 말단 노드를 향하는 루트에 대해 임의의 스위치에서의 하향 포트가 선택되는 경우, 말단 노드를 지향하는 그 스위치로의 들어오는 모든 트래픽은 그 선택된 포트를 통해 라우팅된다. 특히, 만약 모든 링크들이 풀-듀플렉스라면, 그 스위치에 연결된 다른 모든 상향 링크들은 잠재적으로, 해당하는 그 말단 노드를 지향하는 트래픽을 상향 방향으로 운반하고 있다. 선택된 포트의 하향 가중치를 설정한 이후, 더블유팻트리는 라우팅되는 노드의 *receive_weight*와 함께 모든 이용가능한 상향 링크들을 표시(mark)한다. 동일한 하향 로드를 갖는 다수의 하향 포트들이 이용가능한 상황에서, 임의의 루트에 대한 다음 하향 부분을 선택할 때, 가장 적은 상향 가중치를 갖는 포트가 선택된다. 하향 가중치와 상향 가중치 모두에 근거하여 이러한 선택을 하는 것은 말단 노드들의 *receive_weights*에 따라 네트워크 내의 링크들이 발란싱되는 것을 보장한다.

[0134] 도 4는 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 도시한다. 도 4에 도시된 네트워크 환경은 리프 스위치들(420 및 421), 루트 스위치들(425 및 426), 그리고 말단 노드들(401 및 402)을 포함한다. 도 4 내지 도 9에 관한 논의 전체에 걸쳐, 네트워크 환경은 더블유팻트리 알고리즘을 이용한다는 것, 그리고 말단 노드(401)와 말단 노드(402)는 모두 이들의 각각의 리프 스위치들에서 동일한 인덱싱 위치를 갖고, 뿐만 아니라 그 값이 100인 동일한 수신 가중치를 갖는다는 것(즉, 말단 노드(401)와 말단 노드(402) 모두에 대해 *receive_weight* = 100)이 가정될 것이다.

[0135] 도 5는 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 도시한다. 도 5에 도시된 네트워크 환경은 리프 스위치들(420 및 421), 루트 스위치들(425 및 426), 그리고 말단 노드들(401 및 402)을 포함한다. 도 5는 말단 노드(401)를 향하는 루트들을 계산할 때, 동일한 하향 가중치들을 갖는 두 개의 업스트림 포트(upstream port)들, 링크들(450 및 451)이 두 개의 상이한 루트 스위치들(425 및 426) 상에서 이용가능함을 보여준다. 도시된 실시예에서, 도시된 상향 방향에서는 아직 어떠한 가중치도 없고, 또한 하향 방향에서도 아직까지 가중치가 없다. 따라서, 링크(450) 상의 가중치는 상향(Up) = 0, 하향(Down) = 0이다. 마찬가지로, 링크(451) 상의 가중치는 상향 = 0, 하향 = 0이다.

[0136] 도 6은 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 도시한다. 도 6에 도시된 네트워크 환경은 리프 스위치들(420 및 421), 루트 스위치들(425 및 426), 그리고 말단 노드들(401 및 402)을 포함한다. 도 6에서 도시된 바와 같이, 하향 방향 혹은 상향 방향에서 아직 어떠한 가중치도 존재하지 않기 때문에, 가장 좌측에 있는 루트 스위치(425) 상의 포트가 가장 작은 GUID를 운반하는 것으로서 선택된다. 이로 인해, 따라서 링크(450)는 말단 노드(401)의 *receive_weight*를 하향 방향으로 운반한다. 따라서, 링크(450) 상의 가중치는 상향 = 0, 하향 = 100이다. 추가적으로, 링크(452)는 말단 노드(401)의 *receive_weight*를 상향 방향으로 운반한다. 따라서, 링크(452) 상의 가중치는 상향 = 100, 하향 = 0이다.

[0137] 도 7은 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 도시한다. 도 7에 도시된 네트워크 환경은 리프 스위치들(420 및 421), 루트 스위치들(425 및 426), 그리고 말단 노드들(401 및 402)을 포함한다. 도 5와 유사한 도 7에서 도시되는 바와 같이, 말단 노드(402)에 대한 루트가 계산되고 있다. 도 5에 도시된 바와 같이, 동일한 하향 가중치를 갖는 두 개의 업스트림 포트들이 존재한다. 양쪽 링크들(452 및 453)은 모두 동일한 하향 가중치(즉, 0)를 갖는다. 그러나, 링크(452)는 그 값이 100인 상향 가중치를 갖고, 반면 링크(453)는 그 값이 0인 상향 가중치를 갖는다. 이것은 링크(453)의 상향 가중치가 링크(452)의 상향 가중치보다 더 적음을 의미한다. 앞에서 논의된 바와 같이, 두 개의 링크들이 동일한 하향 가중치를 갖는 경우, 만약 하나의 링크가 다른 링크보다 더 큰 상향 가중치를 갖는다면, 알고리즘은 더 작은 상향 가중치를 갖는 포트를 하향 라우팅을 위해 선택할 것이다.

[0138] 도 8은 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 도시한다. 도 8에 도시된 네트워크 환경은 리프 스위치들(420 및 421), 루트 스위치들(425 및 426), 그리고 말단 노드들(401 및 402)을 포함한다. 도 8에 도시된 바와 같이, 링크(450) 상의 상향 가중치(도 7 참조)가 링크(453) 상의 상향 가중치보다 더 크기 때문에, 링크(453)가 노드(402)에 대한 하향 운반을 위해 선택된다. 결과적으로, 따라서 링크(453)는 말단 노드(402)의 *receive_weight*를 하향 방향으로 운반한다. 따라서, 링크(453) 상의 가중치는 상향 = 0, 하향

= 100이다. 추가적으로, 앞에서 논의된 바와 같이, 링크(451)는 말단 노드(402)의 *receive_weight*를 상향 방향으로 운반한다. 따라서, 링크(451) 상의 가중치는 상향 = 100, 하향 = 0이다.

[0139] 도 9는 본 발명의 일 실시예에 따른, 네트워크 환경 내에서 예시적인 포트 선택을 도시한다. 도 9에 도시된 네트워크 환경은 리프 스위치들(420 및 421), 루트 스위치들(425 및 426), 그리고 말단 노드들(401 및 402)을 포함한다. 도 9는 링크 가중치들이 모두 업데이트된 이후 최종 라우팅을 도시한다. 특히, 두 개의 수신기 노드들이 그들의 각각의 리프 스위치들에서 동일한 인덱싱 위치를 공유함에도 불구하고, 토폴로지 내의 이용가능한 링크들을 이용하여, 말단 노드들(401 및 402)에 대한 루트들이 잘 발란싱되어 있음에 유의해야 할 것이다.

[0140] 도 10은 네트워크 환경에서 트리 토폴로지로 정렬된 복수의 말단 노드들 및 복수의 스위치들 간에 효율적인 로드 발란싱을 지원하기 위한 예시적인 방법(1000)을 흐름도를 통해 도시한다. 단계(1001)에서, 예시적인 방법(1000)은 복수의 말단 노드들을 정렬시키는 것으로 시작하고, 여기서, 복수의 말단 노드들은 복수의 스위치들 중 하나 이상의 스위치 상에 있고, 복수의 말단 노드들은 수신 가중치가 감소하는 내림 차순으로 정렬된다. 일 실시예에서, 말단 노드의 수신 가중치는 1과 100 사이의 값일 수 있거나 또 하나의 다른 적절한 범위 내의 값일 수 있으며, 여기서 수신 가중치의 값이 더 큰 것은, 각각의 노드가, 더 작은 수신 가중치들을 갖는 노드들보다, 네트워크 내에서 이에 비례하여 트래픽의 더 많은 배당몫을 가짐을 나타낸다.

[0141] 단계(1002)에서, 예시적인 방법(1000)은 복수의 말단 노드들을 수신 가중치들이 감소하는 내림 차순으로 라우팅하면서 계속되고, 여기서, 라우팅하는 것은 적어도 하나의 하향-진행 포트 및 적어도 하나의 상향-진행 포트를 선택하는 것을 포함한다. 내림 차순으로 라우팅함으로써, 이것은 네트워크로 하여금 트래픽의 더 많은 분량을 수신하는 말단 노드들에 대한 트래픽을 우선화할 수 있게 하고, 그리고 포트 충돌들의 가능성을 감소시킬 수 있게 한다. 일부 실시예들에서, 이러한 선택하는 것은 각각의 수신 가중치들에 근거하고 있다.

[0142] 단계(1003)에서, 예시적인 방법(1000)은 각각의 선택된 하향-진행 포트 상의 누적 하향 가중치를 라우팅되는 말단 노드의 수신 가중치만큼 증가시키면서 진행될 수 있다.

[0143] 단계(1004)에서, 예시적인 방법(1000)은 각각의 선택된 상향-진행 포트 상의 누적 상향 가중치를 라우팅되는 말단 노드의 수신 가중치만큼 증가시키면서 계속될 수 있다.

[0144] **수신_가중치들의 계산(Calculating receive_weights)**

[0145] 일 실시예에서, 노드들에 대한 관리 정보(administrative information)가 이용가능하지 않은 경우, 더 구체적으로 말하면, 노드들의 *receive_weights*가 제공되지 않거나 혹은 이용가능하지 않은 경우, *receive_weights*는 계산될 수 있다. OFED(OpenFabrics Enterprise Distribution; 오픈패브릭스 엔터프라이즈 디스트리뷰션)를 이용하는 실시예들에서, *ibdatacounts*로 지칭되는 유틸리티(utility)가, 데이터 카운터(data counter)들을 판독하기 위해 제공된다. 네트워크를 확립하고 각각의 노드에 동등한 *receive_weights*를 제공한 이후, 새로운 가중치들이 계산될 수 있거나, 혹은 특정된 기간 이후 학습될 수 있다.

[0146] 일 실시예에서, 만약 B가 임의의 기간에 걸쳐 측정된 모든 노드들에 대한 수신 대역폭들의 세트라면, 각각의 노드에 대한 가중치가, 아래에 제시되는 예시적인 방정식에서 주어지는 바와 같이, 선형 변환(linear transformation)을 사용함으로써 범위 [a, b] 내에서 할당될 수 있다.

[0147] $$W(x) = (x - a)\frac{b - a}{\max(B) - \min(B)} + a, \forall x \in B$$

[0148] 일 실시예에서, 가중치들의 새로운 세트가 데이터 카운터들로부터 획득되면, 네트워크는 최적화된 라우팅 테이블들로 재구성될 수 있다. 하지만, 일 실시예에서, 라우팅 테이블들을 최적화되도록 재구성하는 혜택들과 이러한 재구성이 요구할 비가동 시간(downtime) 간의 발란싱을 행하는 발란싱 테스트가 수행될 수 있음에 유의해야 한다. 라우팅 테이블들의 재구성은, 일 실시예에서, 토폴로지 변경과 같은 외부 요인에 의해 재구성이 유발되는 때와 같은 그러한 시간까지 연기될 수 있다.

[0149] 일 실시예의 경우, 미들웨어 머신 환경(middleware machine environment)에서 트리 토폴로지로 정렬된 복수의 말단 노드들 및 복수의 스위치들 간에 효율적인 로드 발란싱을 지원하기 위한 시스템이 제공될 수 있고, 이러한 시스템은, 서브넷 관리 인터페이스와, 그리고 서브넷 매니저를 포함하고, 여기서 서브넷 관리 인터페이스는 복수의 말단 노드들 및 복수의 스위치들과 통신하고, 서브넷 매니저는 서브넷 관리 인터페이스에 결합되어 복수의 말단 노드들을 정렬시키는 것(여기서, 복수의 말단 노드들은 복수의 스위치들 중 하나 이상의 스위치 상에 있고, 복수의 말단 노드들은 수신 가중치가 감소하는 내림 차순으로 정렬됨); 복수의 말단 노드들을 수신 가중

치들이 감소하는 내림 차순으로 라우팅하는 것(여기서 라우팅하는 것은 적어도 하나의 하향-진행 포트 및 적어도 하나의 상향-진행 포트를 선택하는 것을 포함함); 각각의 선택된 하향-진행 포트 상의 누적 하향 가중치를 라우팅되는 말단 노드의 수신 가중치만큼 증가시키는 것; 그리고 각각의 선택된 상향-진행 포트 상의 누적 상향 가중치를 라우팅되는 말단 노드의 수신 가중치만큼 증가시키는 것을 수행하도록 구성된다.

[0150] 서브넷 매니저가 서브넷 관리 인터페이스와 결합되어 앞에서 기재된 바와 같은 임의의 팻-트리 라우팅 알고리즘을 구현할 수 있는 것 혹은 팻-트리 라우팅 알고리즘의 단계들 중 하나 이상의 단계들을 구현할 수 있는 것은 본 발명의 기술분야에서 숙련된 자들에게 명백하다. 앞에서 설명된 서브넷 관리 인터페이스 및 서브넷 매니저와 같은 유닛들/모듈들의 특정 동작 프로세스들에 대해서, 동일한 개념을 공유하는 관련된 방법 실시예에서, 대응하는 단계들이 언급될 수 있고, 이러한 언급 또한 그 관련된 유닛들/모듈들을 개시하는 것으로서 고려된다는 것은, 본 발명의 기술분야에서 숙련된 자들에 대해 명백하다. 따라서, 특정 동작 프로세스들 중 일부는 설명의 편의 및 간결함을 위해 상세히 혹은 되풀이하여 기술되지 않을 것이다. 서브넷 관리 인터페이스, 서브넷 매니저 및/또는 시스템과 같은 유닛, 장치, 및 디바이스는 소프트웨어, (알려져 있거나 혹은 장래에 개발되는) 하드웨어, 그리고/또는 이러한 소프트웨어와 하드웨어의 조합의 형태로 구현될 수 있음을 이해해야 한다.

[0151] 일 실시예에서, 복수의 스위치들 및 복수의 말단 노드들은 고성능 컴퓨팅(HPC) 클러스터에서의 사용을 위한 팻-트리 토폴로지로 정렬된다.

[0152] 일 실시예에서, 서브넷 매니저는 또한 네트워크 내에서 하나 이상의 리프 스위치들 상에 있는 복수의 말단 노드들 각각에 대한 수신 가중치를 수신하도록 구성될 수 있다.

[0153] 일 실시예에서, 적어도 하나의 하향-진행 포트를 선택하는 것은, 복수의 하향-진행 포트들을 비교하는 것과, 그리고 가장 적은 누적 하향 가중치를 갖는 하향-진행 포트를 선택하는 것을 포함한다.

[0154] 일 실시예에서, 적어도 하나의 하향-진행 포트를 선택하는 것은, 복수의 하향-진행 포트들을 비교하는 것과, 그리고 가장 적은 누적 상향 가중치를 갖는 하향-진행 포트를 선택하는 것을 포함한다.

[0155] 일 실시예에서, 적어도 하나의 하향-진행 포트를 선택하는 것은, 복수의 하향-진행 포트들을 비교하는 것과, 그리고 가장 작은 글로벌 고유 식별자를 갖는 하향-진행 포트를 선택하는 것을 포함한다.

[0156] 이러한 단계들 중 적어도 일부는 또한 DSP, FPGA, ASIC 등을 포함하는(하지만, 이러한 것으로만 한정되는 것은 아닌) 다양한 하드웨어에 의해 구현될 수 있다는 것은 본 발명의 기술분야에 있는 사람들에 대해 명백하다. 예를 들어, 실시예들 중 일부 실시예에서 "동작들"은 CPU에서 실행되는 명령들에 의해 구현될 수 있거나, 혹은 "동작들"의 기능을 구현하는 DSP, FPGA, ASIC와 같은 특수 프로세서에 의해 구현될 수 있다.

[0157] 본 발명의 기술분야에서 통상의 기술을 가진 자들이 이해하는 바와 같이, 블록도에 의해 표현된 기능들은 소프트웨어 및/또는 하드웨어에 의해 수행될 수 있다. 이벤트에 의해 구동되는 것(event-driven), 인터럽트에 의해 구동되는 것(interrupt-driven) 등과 같은 특정 프로세싱 전략에 따라, 다양한 기능들이 도면에서 예시된 바와 다른 순서 혹은 시퀀스로 수행될 수 있다. 유사하게, 하나 이상의 단계들 혹은 기능들은, 명시적으로 예시되지는 않았지만, 반복적으로 수행될 수 있다. 마찬가지로, 다양한 기능들은 특정 구현에 따라 생략될 수 있다. 본 발명의 기술분야에서 숙련된 자들에게 알려진 다양한 기능들이 명시적으로 예시 혹은 기술되지 않을 수 있지만, 예시되는 블록들 혹은 모듈들에 의해 시사될 수 있다. 일 실시예의 경우, 예시되는 기능들은 주로 (컴퓨터 판독가능 저장 매체에 저장되어 시스템의 동작을 제어하기 위해 마이크로프로세서-기반의 제어기에 의해 실행되는) 소프트웨어, 명령들 혹은 코드로 구현되는 제어 로직(control logic)에 의해 수행된다. 자기 테이프 드라이브(magnetic tape drive)에 관해 일반적으로 예시되고 기술되어도, 본 발명의 기술분야에서 통상의 기술을 가진 자들은 다양한 기능들이 다양한 다른 타입의 주변 저장 디바이스들에 적용가능함을 인식할 것이다.

[0158] 본 발명은, 본 개시내용의 가르침들에 따라 프로그래밍되는 하나 이상의 프로세서들, 메모리, 및/또는 컴퓨터 판독가능 저장 매체들을 포함하는 하나 이상의 종래의 범용 혹은 특화된 디지털 컴퓨터, 컴퓨팅 디바이스, 머신, 혹은 마이크로프로세서를 사용하여 편리하게 구현될 수 있다. 소프트웨어 기술분야에서 숙련된 자들에게 명백하게 될 것으로서, 본 개시내용의 가르침들에 근거하여 적절한 소프트웨어 코딩이 숙련된 프로그래머들에 의해 용이하게 준비될 수 있다.

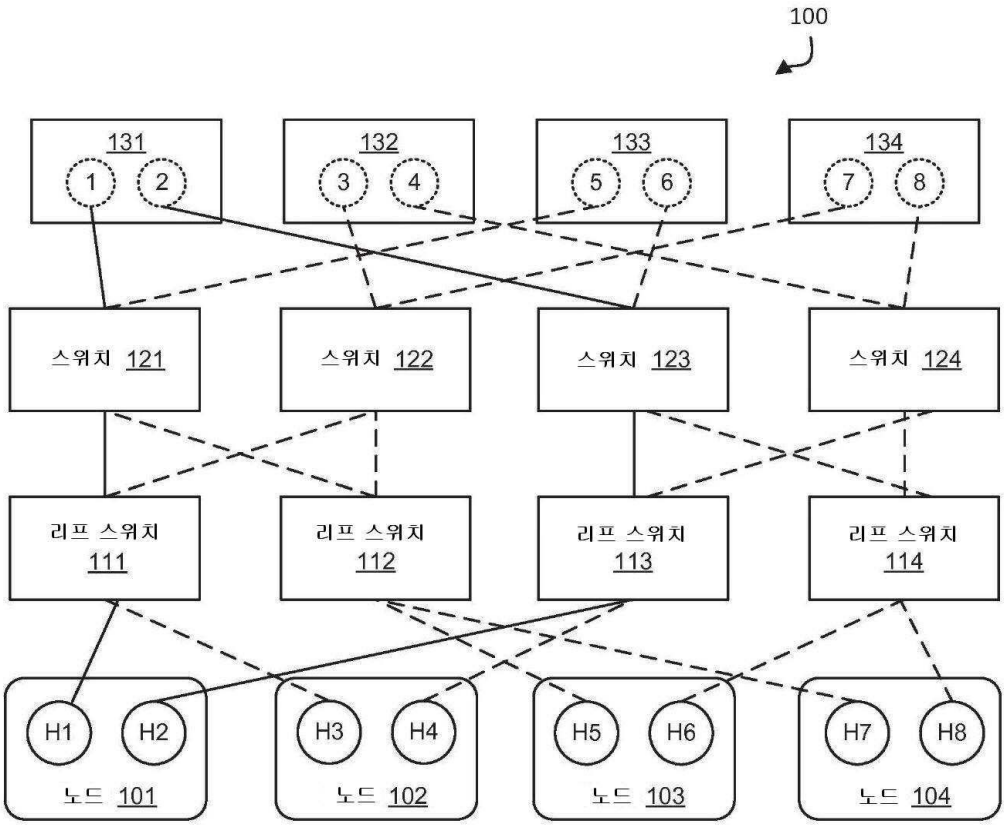
[0159] 일부 실시예들에서, 본 발명은 본 발명의 프로세스들 중 임의의 프로세스를 수행하도록 컴퓨터를 프로그래밍하는데 사용될 수 있는 명령들이 저장되어 있는 비-일시적 저장 매체 혹은 컴퓨터 판독가능 매체(매체들)인 컴퓨터 프로그램 제품을 포함한다. 저장 매체는, 임의 타입의 디스크(여기에는 플로피 디스크들, 광학 디스크들, DVD, CD-ROM들, 마이크로드라이브, 및 광자기 디스크들이 포함됨), ROM들, RAM들, EPROM들, EEPROM들, DRAM들,

VRAM들, 플래시 메모리 디바이스들, 자기 혹은 광학 카드들, 나노시스템들(여기에는 분자 메모리 IC들이 포함됨), 또는 명령들 및/또는 데이터를 저장하는데 적합한 임의 타입의 매체들 혹은 디바이스를 포함할 수 있지만, 이러한 것으로만 한정되는 것은 아니다.

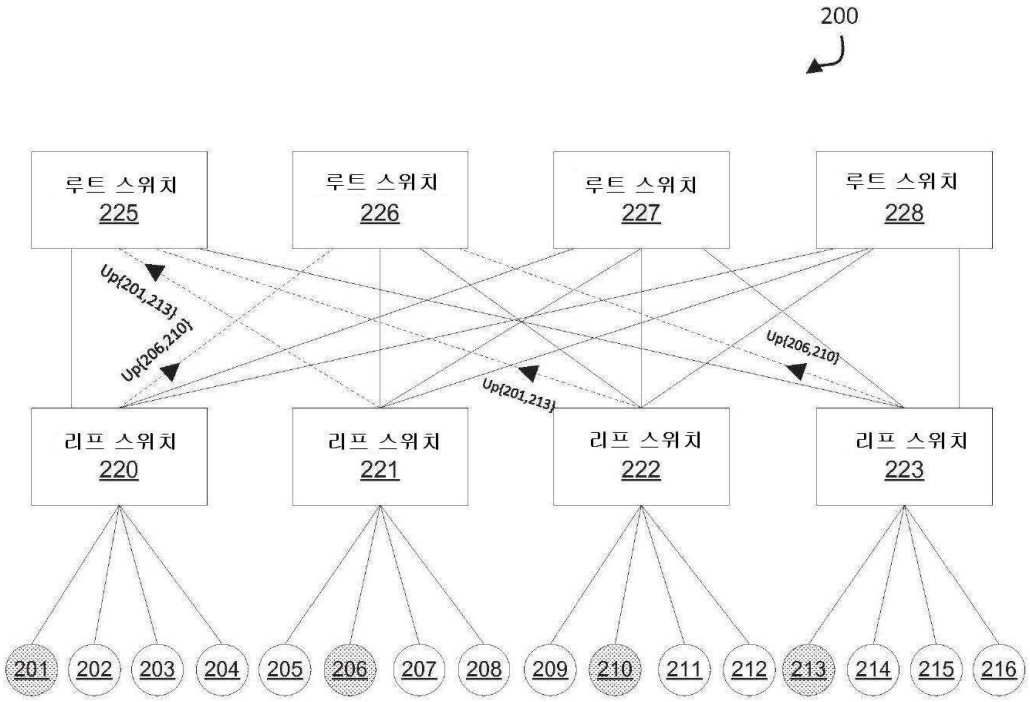
[0160] 본 발명의 앞서의 설명은 예시 및 설명 목적으로 제공된 것이다. 이것은 정확히 그 개시되는 형태들로만 본 발명을 한정하려고 의도된 것이 아니며 또한 본 발명의 모든 실시예들을 나타내도록 의도된 것이 아니다. 많은 수정물들 및 변형물들이 본 발명의 기술분야에서 숙련된 실무자들에게는 명백하게 될 것이다. 본 발명의 실시예들은, 본 발명의 원리들 및 그 실제 응용을 가장 잘 설명하기 위해서 선택 및 기술된 것으로, 그럼으로써 본 발명의 기술분야에서 숙련된 다른 사람들로 하여금 다양한 실시예들에 대해 본 발명을 이해할 수 있게 하고 아울러 그 고려되는 특정 용도에 맞는 다양한 수정물들을 갖는 본 발명을 이해할 수 있게 하기 위해 선택 및 기술된 것이다. 본 발명의 범위는 다음의 청구항들 및 이들의 등가물들에 의해 정의될 수 있게 의도되었다.

도면

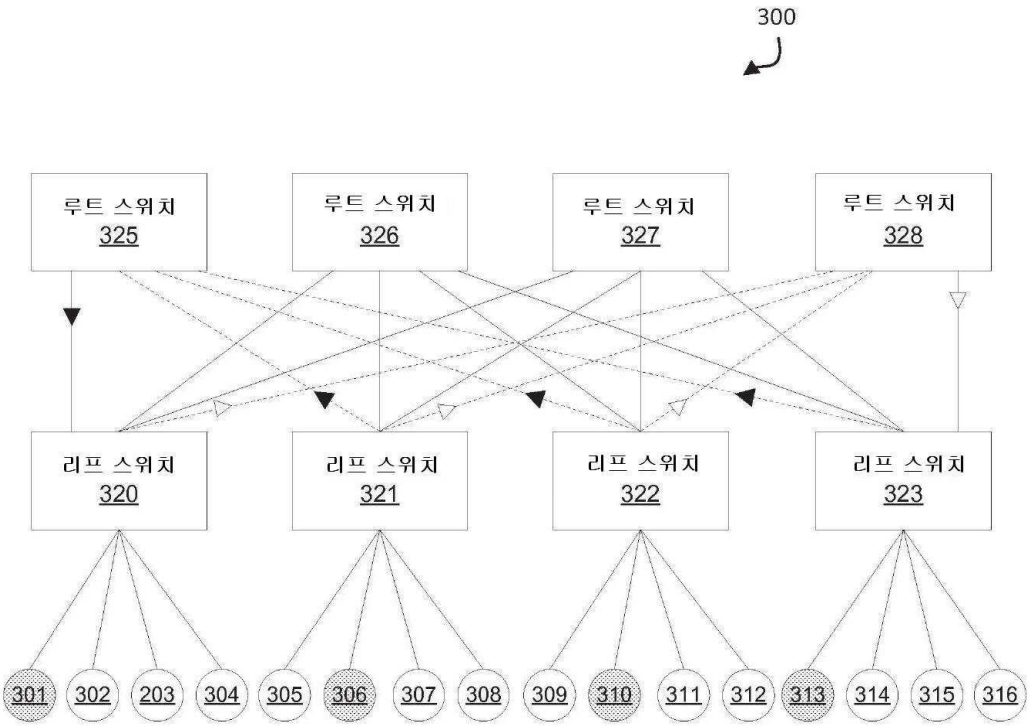
도면1



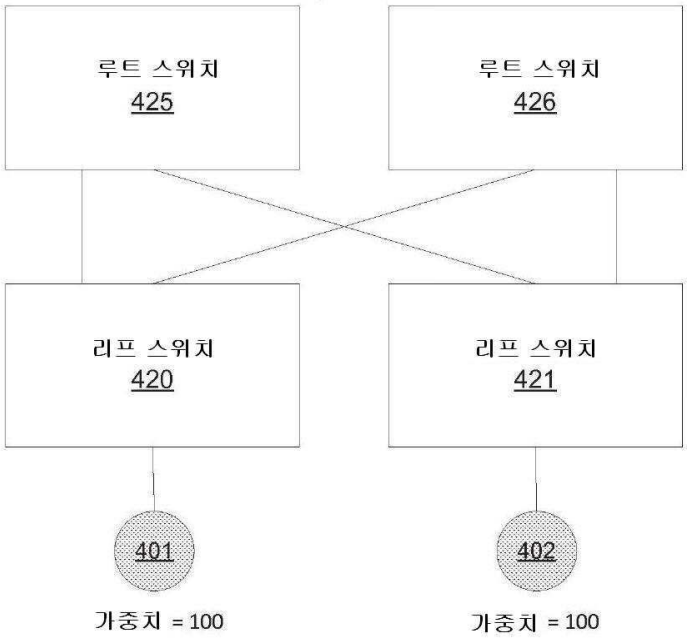
도면2


200
루트 스위치 225
루트 스위치 226
루트 스위치 227
루트 스위치 228
Up{201,213}
Up{206,210}
Up{206,210}
Up{201,213}
리프 스위치 220
리프 스위치 221
리프 스위치 222
리프 스위치 223
201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216

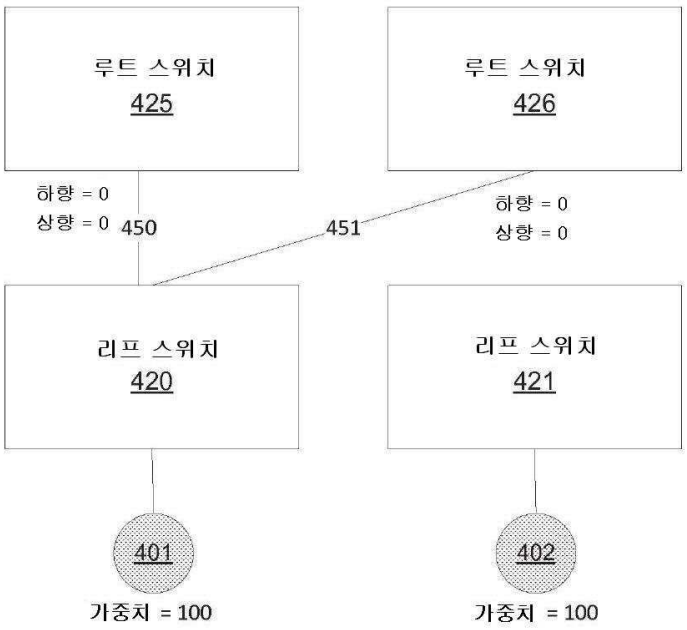
도면3


300
루트 스위치 325
루트 스위치 326
루트 스위치 327
루트 스위치 328
리프 스위치 320
리프 스위치 321
리프 스위치 322
리프 스위치 323
301 302 203 304 305 306 307 308 309 310 311 312 313 314 315 316

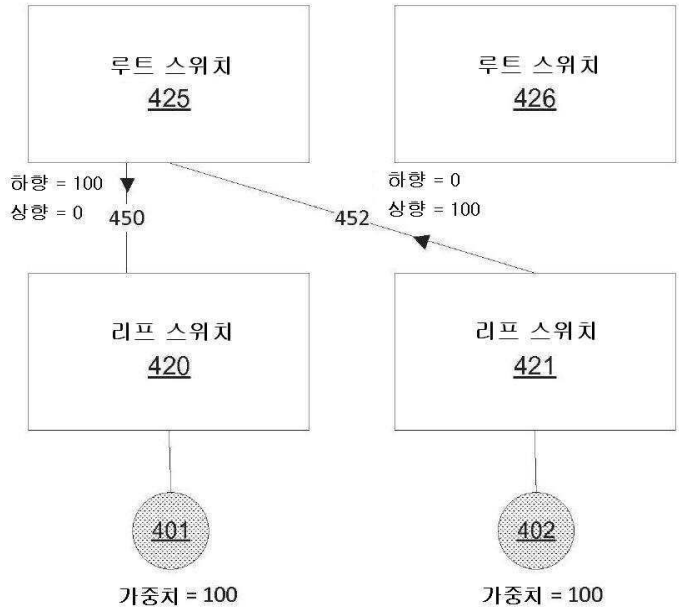
도면4


루트 스위치
425
루트 스위치
426
리프 스위치
420
리프 스위치
421
401
402
가중치 = 100
가중치 = 100

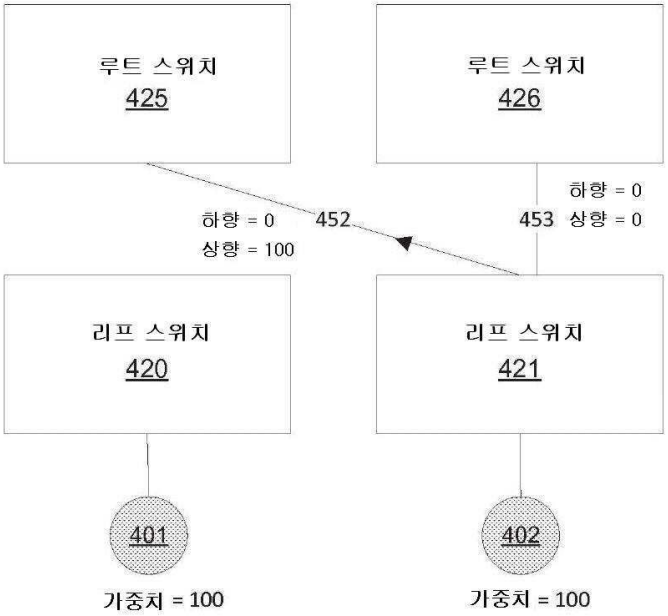
도면5


루트 스위치
425
루트 스위치
426
하향 = 0
상향 = 0
450
451
하향 = 0
상향 = 0
리프 스위치
420
리프 스위치
421
401
402
가중치 = 100
가중치 = 100

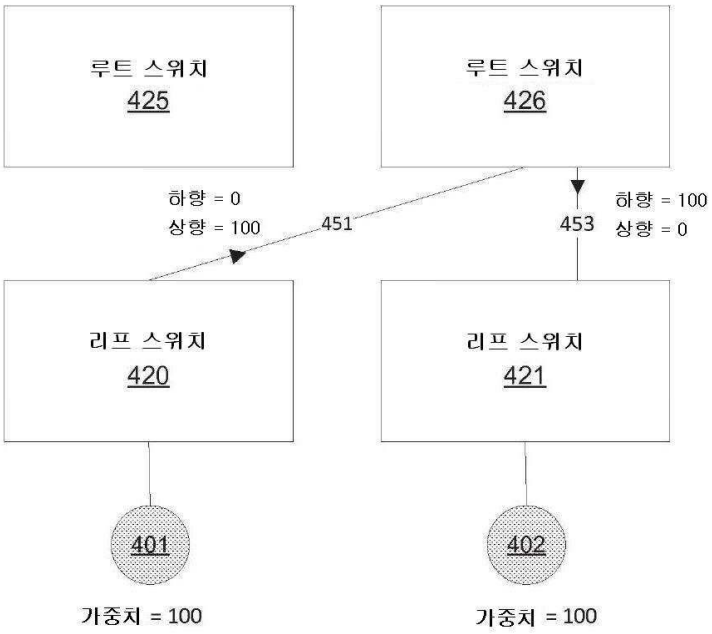
도면6


루트 스위치
425
루트 스위치
426
하향 = 100
상향 = 0
450
452
하향 = 0
상향 = 100
리프 스위치
420
리프 스위치
421
401
402
가중치 = 100
가중치 = 100

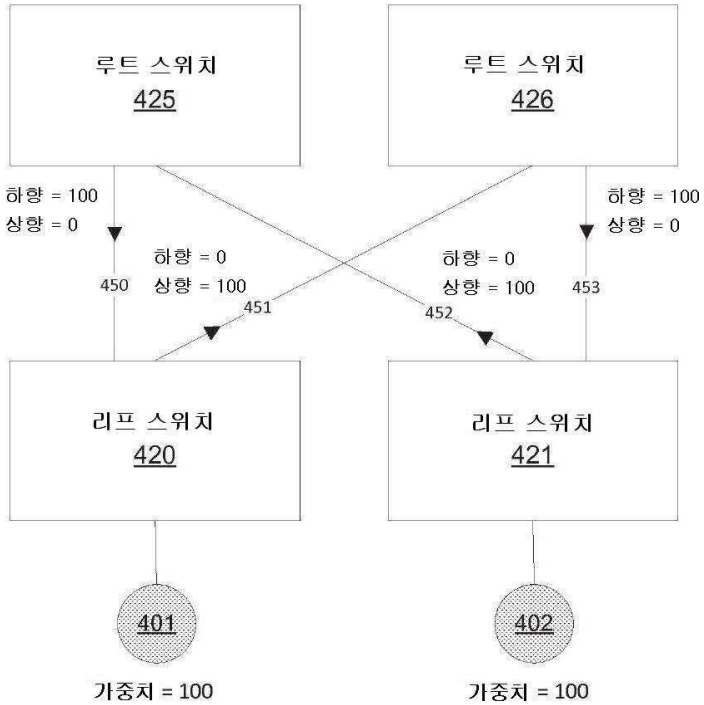
도면7


루트 스위치
425
루트 스위치
426
하향 = 0
상향 = 100
452
하향 = 0
453
상향 = 0
리프 스위치
420
리프 스위치
421
401
402
가중치 = 100
가중치 = 100

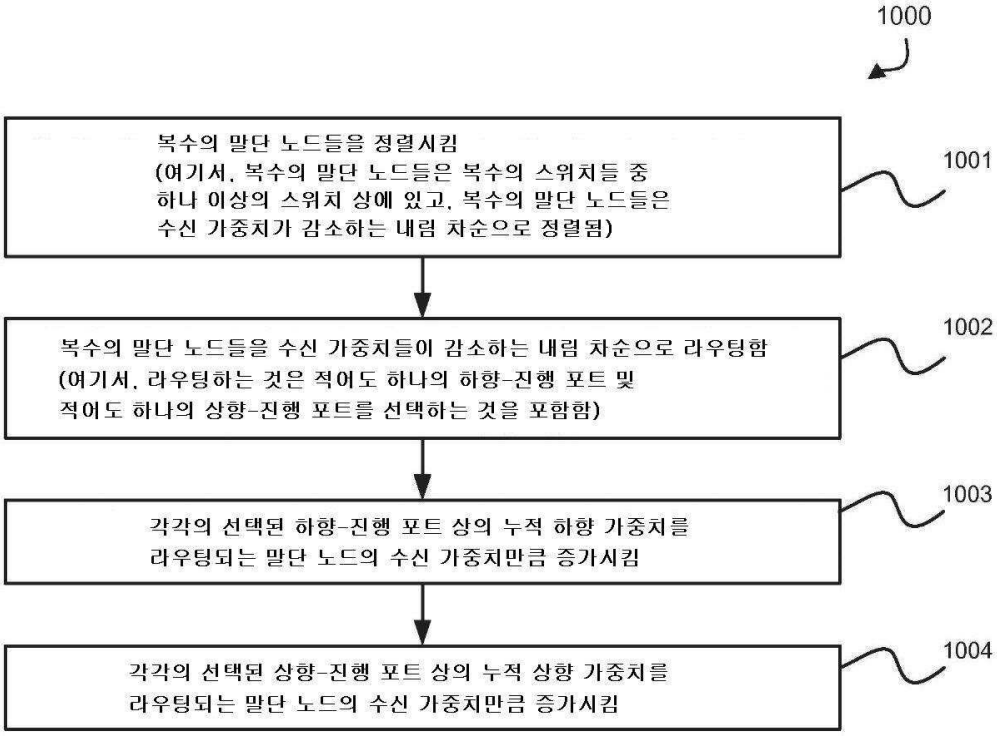
도면8


루트 스위치
425
루트 스위치
426
하향 = 0
상향 = 100
451
453
하향 = 100
상향 = 0
리프 스위치
420
리프 스위치
421
401
402
가중치 = 100
가중치 = 100

도면9


루트 스위치
425
루트 스위치
426
하향 = 100
상향 = 0
하향 = 0
상향 = 100
450
451
452
하향 = 0
상향 = 100
453
하향 = 100
상향 = 0
리프 스위치
420
리프 스위치
421
401
402
가중치 = 100
가중치 = 100

도면10


1000
복수의 말단 노드들을 정렬시킴
(여기서, 복수의 말단 노드들은 복수의 스위치들 중
하나 이상의 스위치 상에 있고, 복수의 말단 노드들은
수신 가중치가 감소하는 내림 차순으로 정렬됨)
1001
복수의 말단 노드들을 수신 가중치들이 감소하는 내림 차순으로 라우팅함
(여기서, 라우팅하는 것은 적어도 하나의 하향-진행 포트 및
적어도 하나의 상향-진행 포트를 선택하는 것을 포함함)
1002
각각의 선택된 하향-진행 포트 상의 누적 하향 가중치를
라우팅되는 말단 노드의 수신 가중치만큼 증가시킴
1003
각각의 선택된 상향-진행 포트 상의 누적 상향 가중치를
라우팅되는 말단 노드의 수신 가중치만큼 증가시킴
1004