(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2014-0147050
(43) 공개일자 2014년12월29일

(51) 국제특허분류(Int. Cl.)
H04L 12/741 (2013.01) *H04L 12/721* (2013.01)
(21) 출원번호 10-2014-0074391
(22) 출원일자 2014년06월18일
심사청구일자 없음
(30) 우선권주장
13/921,090 2013년06월18일 미국(US)

(71) 출원인
엑스플라이언트 인코포레이션
미국, 캘리포니아 95131, 산 호세, 엔. 퍼스트 스트리트 2315
(72) 발명자
허치슨 가이
미국, 캘리포니아 95050, 산타 클라라, 벤톤 스트리트 1261
다니엘 자히
미국, 캘리포니아 94303, 팔로 알토, 그리어 로드 3517
(뒷면에 계속)
(74) 대리인
오세일

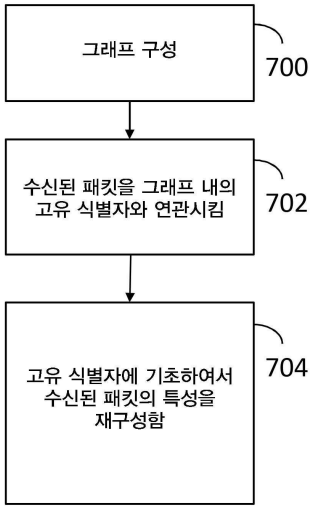
전체 청구항 수 : 총 20 항

(54) 발명의 명칭 **파스 트리 내에서 경로들을 고유하게 이뉴머레이션하기 위한 장치 및 방법**

(57) 요 약

방법은 네트워크 트래픽과 연관된 패킷 헤더들의 세트를 특성화하는 그래프를 구성하는 단계를 포함한다. 상기 그래프는 상기 그래프 내에서 경로를 형성하는 패킷 헤더들의 각 가능한 조합에 대한 고유한 식별자를 갖는다. 수신된 패킷이 상기 그래프 내의 고유 식별자와 연관된다. 상기 고유 식별자에 기초하여서 상기 수신된 패킷의 특성들이 재구성된다.

대 표 도 - 도7



(72) 발명자

슈미트 제랄드

미국, 캘리포니아 95127, 산 호세, 오브저버토리 드라이브 10431

간디 사친

미국, 캘리포니아 95118, 산 호세, 스코사 에비뉴 1415

특허청구의 범위

청구항 1

네트워크 트래픽과 연관된 패킷 헤더들의 세트를 특성화하는 그래프를 구성하는 단계로서, 상기 그래프는 상기 그래프 내에서 경로를 형성하는 패킷 헤더들의 각 가능한 조합에 대한 고유한 식별자를 갖는, 상기 그래프를 구성하는 단계;

수신된 패킷을 상기 그래프 내의 고유 식별자와 연관시키는 단계; 및

상기 고유 식별자에 기초하여서 상기 수신된 패킷의 특성들을 재구성하는 단계를 포함하는,

방법.

청구항 2

제 1 항에 있어서,

상기 고유 식별자는 비-가환성 함수 (non-commutative function) 에 기초하는,

방법.

청구항 3

제 2 항에 있어서,

상기 비-가환성 함수는 CRC (Cyclic Redundancy Check) 함수인,

방법.

청구항 4

제 1 항에 있어서,

상기 특성들은 거쳐진 경로 (traversed path) 내에 존재하는 헤더들을 특정하는,

방법.

청구항 5

제 1 항에 있어서,

상기 특성들은 플래그들의 연관된 세트를 갖는,

방법.

청구항 6

제 1 항에 있어서,

상기 특성들은 동작들의 연관된 세트를 갖는,

방법.

청구항 7

제 1 항에 있어서,

상기 그래프를 경로 테이블로서 연관 메모리 내에 로딩하는 (loading) 단계를 더 포함하는,

방법.

청구항 8

제 1 항에 있어서,

단일 룩업 (single lookup) 으로 다수의 경로들을 매칭시킬 수 있는 다중 동시적 매칭 파서 (multiple simultaneous match parser) 로서 연관 메모리를 동작시키는 단계를 더 포함하는,

방법.

청구항 9

네트워크 트래픽과 연관된 패킷 헤더들의 세트를 특성화하는 그래프를 저장하는 연관 메모리 (associative memory) 로서, 상기 그래프는 상기 그래프 내에서 경로를 형성하는 패킷 헤더들의 각 가능한 조합에 대한 고유한 식별자를 가지며, 상기 연관 메모리는 수신된 패킷의 속성들을 고유 식별자와 매칭시키는, 상기 연관 메모리; 및

상기 고유 식별자에 기초하여서 상기 수신된 패킷의 특성들을 재구성하는 인덱스 메모리를 포함하는,

프로세서.

청구항 10

제 9 항에 있어서,

상기 연관 메모리는 TCAM (Ternary Content Addressable Memory) 인,

프로세서.

청구항 11

제 9 항에 있어서,

상기 연관 메모리는 단일 룩업 (single lookup) 으로 다수의 경로들을 매칭시킬 수 있는 다중 동시적 매칭 파서 (multiple simultaneous match parser) 로서 동작하는,

프로세서.

청구항 12

제 9 항에 있어서,

상기 고유 식별자는 비-가환성 함수 (non-commutative function) 에 기초하는,

프로세서.

청구항 13

제 12 항에 있어서,

상기 비-가환성 함수는 CRC (Cyclic Redundancy Check) 함수인,

프로세서.

청구항 14

제 9 항에 있어서,

상기 특성들은 거쳐진 경로 (traversed path) 내에 존재하는 헤더들을 특정하는,

프로세서.

청구항 15

제 9 항에 있어서,

상기 특성들은 플래그들의 연관된 세트를 갖는,

프로세서.

청구항 16

제 9 항에 있어서,

상기 특성들은 동작들의 연관된 세트를 갖는,

프로세서.

청구항 17

그래프 내에 호들 (arcs) 에 할당된 고유 값들을 형성하는 단계;

상기 그래프 내에서 경로들을 한정하는 단계;

상기 할당된 값들에 기초하여서 상기 그래프를 통한 계산된 경로들을 형성하는 단계;

상기 계산된 경로들을 사용하여서 경로 테이블을 구성하는 단계; 및

상기 계산된 경로들 중 임의의 경로들이 동일한 값을 갖는지의 여부를 결정하고 그러하다면 상기 계산된 경로들을 형성하는 단계 및 상기 경로 테이블을 구성하는 단계를 반복하는 단계를 포함하는,

방법.

청구항 18

제 17 항에 있어서,

상기 경로들을 한정하는 단계는 상기 그래프 내에서의 순환적 경로들 (cyclic paths) 을 통한 천이들의 개수를 한정하는 단계를 포함하는,

방법.

청구항 19

제 17 항에 있어서,

상기 경로들을 한정하는 단계는 상기 그래프 내의 경로들을 선택적으로 제거하는 단계를 포함하는,

방법.

청구항 20

제 17 항에 있어서,

상기 할당된 고유 값들은 비-가환성 함수에 기초하는,

방법.

명 세 서

기 술 분 야

[0001] 관련 출원에 대한 교차 참조

[0002] 본원은 2013년 6월 18일자에 출원된 미국 특허 출원 번호 13/921,090에 대한 우선권을 주장하며, 이 문헌의 내용들은 본 명세서에서 참조로서 인용된다.

[0003] 본 발명은 네트워크 데이터 패킷들을 스위칭하는 것에 관한 것이다. 보다 구체적으로, 본 발명은 파스 트리 (parse tree) 에서 고유 경로들의 세트를 이뉴머레이션 (enumeration) 함으로써 네트워크 데이터 패킷들을 스위칭하는 것에 관한 것이다.

배 경 기 술

[0004] 데이터 패킷들을 입력 포트에서 출력 포트로 스위칭하기 위한 다양한 기법들이 존재한다. 스위칭 디바이스에서 요구되는 제 1 동작은 자신이 수신한 패킷의 타입을 식별하고 이 패킷으로부터 데이터 필드들의 위치를 파악하고 추출하는 것이다. 이러한 프로세스는 패킷을 파싱 (parsing) 하는 것으로서 알려진다.

[0005] 파싱은 바이트들의 스트링을 그의 구성요소들 (constituents) 로 컴퓨터에 의해서 형식적으로 분석하는 것을 수반하며, 이로써 서로에 대한 그들의 신태틱 관계 (syntactic) 를 나타내는 파스 트리를 낳으며, 이 트리는 시멘틱 정보 (semantic information) 및 다른 정보를 또한 포함할 수 있다. 이로써, 파스 트리는 몇몇 형식 문법 (formal grammar) 에 따라서 스트링의 신태틱 구조를 표현하는 오더링된 루트형 트리 (ordered rooted tree) 이다. 파스 트리들은 통상적으로 의존 문법들 (dependency grammars) 의 의존 관계의 차원에서 구성된다. 파스 트리들은 그들의 구조 및 요소들이 입력 언어의 신택스를 보다 명확하게 반영한다는 점에서, 추상 신택스 트리들 (abstract syntax trees) (또한 간단하게 신택스 트리로도 알려짐) 과는 구별된다.

[0006] 지난 십년에 걸쳐서, 컴퓨터 네트워킹은 ATM (Asynchronous Transfer Mode), 토큰 링 (Token ring), 파이버 채널 (Fiber channel), 및 인피니밴드 (Infiniband) 와 같은 매우 다양한 특정 목적용 네트워크 프로토콜들 및 명세들로부터 보편적 물리 층 및 프로토콜 (universal physical layer and protocol) 로서 이더넷을 사용하는 것으로 수렴하였다. 호 (call) 핸들링, 디스크 액세스 및 프로세서 상호접속과 같은 특정 특징들에서 지향되는 이러한 레거시 네트워크 프로토콜들 (legacy network protocols) 이외에, 가상 머신들, 분산형 컴퓨팅 및 클라우드 컴퓨팅과 같은 새로운 기술들은 이더넷 (Ethernet) 및 TCP/IP (Transmission Control Protocol/Internet Protocol) 에 새롭고 변화된 요구사항들을 부여하였다.

[0007] 이러한 수렴은 특정 목적용 네트워크 프로토콜들이 제공했던 것과 동일한 특징들을 제공하지만 전송 층으로서 이더넷을 사용하는 새로운 프로토콜들, 알고리즘 및 태그들의 갑작스러운 증가로 이어졌다. 새로운 프로토콜들, 특징들 및 기술들의 변화 레이트가 증가함에 따라서, 이에 대응하여서 이러한 기술들을 현장에 보다 신속하게 배치할 필요가 생겼다.

[0008] 통상적으로, 신규 프로토콜들은 하드웨어에 대한 변화, 구체적으로 스위칭 기능을 구현하는 ASIC (Application Specific Integrated Circuit) 에 대한 물리적 변화에 의해서 제공될 수 있다. 이는 신규 프로토콜들의 전개 레이트를 이러한 변화들이 구현 및 가공될 수 있는 속도로 제약시킨다.

[0009] 최근 수년에 걸쳐서, 본 산업 분야는 이러한 변화의 영향을 완화시키기 위한 일환으로서 프로그램가능한 파서들(parsers)을 제공하는 기법으로 이동하였다. 프로그램가능한 파서들은 파싱 동안에 찾게 된 헤더들에 기초하여서 가능한 패킷 타입들의 트리를 구성함으로써 동작한다. 현 파서들은 플래그(flag)로서 알려진, 찾게 된 헤더 각각에 대한 비트를 설정함으로써 파싱 동안에 이 파서들이 만나게 된 헤더들의 타입들을 추적한다.

[0010] 찾게 된 헤더들의 타입을 추적하기 위해서 플래그들을 사용하는 것은 몇몇 경우들에서는 불충분한데, 그 이유는 이 방식이 헤더들이 찾아진 순서를 기록하지 않기 때문이다. 따라서, 기존의 파서들은 중요하다면 순서를 추적하기 위해서 추가 플래그 비트를 사용해야 하며 순서 정보가 보존되어야 하는 경우들을 알아야 한다.

[0011] 도 1은 7 개의 상이한 타입의 패킷 헤더들을 검출하는 샘플 파스 트리를 나타낸다. 모든 패킷들은 선태적(optional) 헤더들 VLAN (102) 또는 VLAN (103) 를 포함할 수 있는 이더넷 패킷 (101) 으로서 도착하고 이어서 IPv4 (104), IPv6 (105), 또는 기타 (108) 로서 분류된다. IP 패킷들은 TCP (106), UDP (107), 또는 Other IP (109) 로서 더 분류된다. 이 파스 트리에서, 8 개의 플래그들의 세트가 취해진 경로를 고유하게 결정할 수 있다.

[0012] 도 2는 도 1의 파스 트리를 나타내지만, 도 2에서는 GRE 헤더 (210) 가 부가되었다. GRE 헤더는 이더넷 헤더를 가리킬 수 있기 때문에, 루프가 이제는 가능하며 플래그들의 세트가 헤더가 찾아지게 된 위치를 더 이상 결정할 수 없다. 예를 들어서,

[0013] Eth -> VLAN -> IPv4 -> GRE -> Eth

[0014] Eth -> IPv4 -> GRE -> Eth -> VLAN

[0015] 위의 스퀀스들이 이제 동일한 플래그들을 정확하게 설정할 것이지만, 파싱된 패킷 데이터를 수신하는 포워딩 엔진에 의해서 상이하게 다루어질 필요가 있을 수 있다.

[0016] 전술한 바를 고려하여, 파스 트리를 통한 경로들을 찾기 위한 개선된 기법을 제공하는 것이 바람직할 것이다.

발명의 내용

[0017] 방법은 네트워크 트래픽과 연관된 패킷 헤더들의 세트를 특성화하는 그래프를 구성하는 단계를 포함한다. 상기 그래프는 상기 그래프 내에서 경로를 형성하는 패킷 헤더들의 각 가능한 조합에 대한 고유한 식별자를 갖는다. 수신된 패킷이 상기 그래프 내의 고유 식별자와 연관된다. 상기 고유 식별자에 기초하여서 상기 수신된 패킷의 특성들이 재구성된다.

[0018] 프로세서는 네트워크 트래픽과 연관된 패킷 헤더들의 세트를 특성화하는 그래프를 저장하는 연관 메모리(associative memory)를 포함한다. 상기 그래프는 상기 그래프 내에서 경로를 형성하는 패킷 헤더들의 각 가능한 조합에 대한 고유한 식별자를 갖는다. 상기 연관 메모리는 수신된 패킷의 속성들을 고유 식별자와 매칭시킨다. 인덱스 메모리가 상기 고유 식별자에 기초하여서 상기 수신된 패킷의 특성들을 재구성한다.

[0019] 방법은 그래프 내에 호들(arcs)에 할당된 고유 값들을 형성하는 단계; 상기 그래프 내에서 경로들을 한정하는 단계; 상기 할당된 값들에 기초하여서 상기 그래프를 통한 계산된 경로들을 형성하는 단계; 상기 계산된 경로들을 사용하여서 경로 테이블을 구성하는 단계; 및 상기 계산된 경로들 중 임의의 경로들이 동일한 값을 갖는지의 여부를 결정하고 그러하다면 상기 계산된 경로들을 형성하는 단계 및 상기 경로 테이블을 구성하는 단계를 반복하는 단계를 포함한다.

도면의 간단한 설명

[0020] 본 발명은 첨부 도면을 참조하여서 취해지는 다음의 상세한 설명 부분과 연관되어서 보다 완벽하게 이해된다.

도 1은 무-루프 토폴로지(loop-free topology)를 갖는 파스 트리를 예시한다.

도 2는 루프들을 포함하는 파스 트리를 예시한다.

도 3은 본 발명의 실시예에 따른 할당된 노드 및 경로 값들을 갖는 도 2로부터의 파스 트리를 예시한다.

도 4는 본 발명의 실시예와 연관된 프로세싱 동작들을 예시한다.

도 5a는 본 발명의 실시예에 따른 일 예시적인 경로에 대해 컴퓨팅된 헤더 값들 및 경로 값들을 나타낸다.

도 5b는 본 발명의 실시예에 따른 제 2 샘플 경로에 대해 컴퓨팅된 헤더 값들 및 경로 값들을 나타낸다.

도 6은 본 발명의 실시예에 따른 경로 값 계산을 사용하는 파서의 하드웨어 구현을 예시한다.

도 7은 본 발명의 실시예와 연관된 프로세싱 동작들을 예시한다.

몇몇 도면들에 걸쳐서 유사한 참조 부호들은 대응하는 요소들을 나타낸다.

발명을 실시하기 위한 구체적인 내용

[0021] 본 발명은 파스 트리에 걸쳐 있는 헤더들의 각 가능한 조합에 고유 식별자를 할당한다. 유리하게는, 이러한 고유한 조합은 하드웨어에서 효율적으로 컴퓨팅될 수 있다. 도 3은 본 발명의 실시예에 따른 할당된 노드 및 경로 값들을 갖는 도 2로부터의 파스 트리를 예시한다.

[0022] 도 4는 본 발명의 실시예와 연관된 프로세싱 동작들을 예시한다. 먼저, 그래프 내의 호들 (arcs) 에 값들이 할당된다 (410). 예를 들어서, 방향성 순환적 또는 비순환적 그래프 (directed cyclic or acyclic graph) 는 이 그래프 내의 각 호에 할당된 값들을 가질 수 있다. 이 값은 임의의 값이지만, 각 호에 대해서는 고유해야 한다. 값들이 순차적으로 할당될 수 있지만, 랜덤 할당 또는 반-램덤 (semi-random) 할당이 보다 양호한 결과를 산출할 가능성이 크다. 이러한 문맥에서, 호는 그래프 내에서 2 개의 노드들 간의 링크이다. 경로 (path) 는 그래프를 통한 호들의 시퀀스이다.

[0023] 이어서, 그래프 내의 경로들에 대해서 제약 사항들이 부여된다 (412). 예를 들어서, 도 2에 도시된 바와 같은 방향성 순환적 그래프는 의도된 애플리케이션의 구현자의 지식에 기초하여서, 천이들의 개수를 순환적 경로들보다 작게 제약함으로써 방향성 비순환적 그래프로 변환될 수 있다. 예를 들어서, GRE (210) 로부터 이더넷 (201) 으로의 호가 오직 한번만 거치게 되도록 한정하는 것은 이 그래프를 비순환적 그래프로 변환한다. 비순환적 그래프가 일단 생성되면, 구현자의 지식에 기초하여서 구현자의 애플리케이션에서 발생하지 않는 것으로 알려지거나 관심 대상이 아닌 노드들로의 천이들을 제거함으로써 천이들의 개수가 더욱 줄어들 수 있다.

[0024] 다음으로, 경로들이 그래프 내에서 계산된다 (414). 도 5a 및 도 5b는 이하에서 기술되는 바와 같은, 2 개의 그러한 경로들에 대한 계산의 실례를 나타낸다. 증분적 계산을 수행하는 랜덤 선택에 의해서 또는 구현자에 의해서 공식이 선정되어야 한다. 이 공식은 비-가환성 (non-commutative) 이어야 하며, 이는 A+B != B+A를 의미한다. 대부분의 CRC (Cyclic Redundancy Check) 함수가 이 조건을 만족시킨다.

[0025] 다음으로, 경로 테이블 (path table) 이 구성된다 (416). 즉, 이뉴머레이션된 (enumerated) 가능한 경로들의 결과들로부터 테이블이 구성된다. 다음으로, 이 테이블이 충돌치 (collision) 로 지칭되는 공통 값들에 대해서 평가된다. 2 개의 경로들이 동일한 값을 가지지 않을 수 있으며, 가지게 되면 충돌이 존재한다. 충돌이 존재하지 않으면 (418에서 "아니오"), 프로세싱은 완료된다 (420). 그렇지 않고 충돌이 존재하면 (418에서 "예"), 프로세싱은 블록 (414) 으로 돌아간다. 충돌이 발생하였다면, 새로운 호 할당 세트 및/또는 새로운 공식이 적용되고 블록들 (414 내지 418) 에 대한 프로세싱이 반복된다. 이러한 사이클은 무충돌 테이블 (collision-free table) 이 생성되거나 알고리즘이 어느 정도의 임의적 한계에 이르러서 실패를 통보할 때까지 계속된다.

[0026] 도 3은 도 2로부터의 루프들을 가지지만 천이 호들 (transition arcs) 각각에 대해서 부여된 고유 값들의 세트이 이뉴머레이션된 예시적인 파스 트리를 나타낸다. 비-가환성 함수의 실례는 이하에서 Python 의사코드 (pseudocode) 로 나타난다. 도 5a 및 도 5b에서 도시된 값들에 대해서, 이 함수는 도 5a 및 도 5b의 IncCalc 행에서 도시된 출력 값들을 생성한다.

[0027] **간단한 비-가환성 함수의 실례**

[0028] 이 함수는 cstate의 하위 비트가 1 인지의 여부에 기초하여서 값 0×83 을 그의 현 상태로 반복적으로 XOR한다. 이어서, 이 함수는 cstate의 값을 1 비트만큼 시프트한다. 이 시프트는 함수에 그의 비-가환성 특성을 제공한다. 이는 아래와 같은 다항식 x^7 + x + 1을 사용하여서 8 비트 CRC 함수의 단순화된 버전이다:

[0029]
```
def function8 (val, cstate):
```

[0030]
```
    if (cstate & 1):
```

[0031]
```
            cstate = (cstate >> 1) ^ 0x83
```

[0032]
```
        else:
```

[0033]
```
            cstate = (cstate >> 1)
```

[0034]
```
        cstate = (cstate ^ val) & 0xFF
```

[0035]
```
        return cstate
```

[0036]
```
def twoseq():
```

[0037]
```
        path_fig5a = [1,7,20,17]  # arc sequences from Fig 5A
```

[0038]
```
        path_fig3b = [3,20,17,1]  # arc sequences from Fig 5B
```

[0039]
```
        cstate = 0
```

[0040]
```
        for arcValue in path_fig5a:
```

[0041]
```
            cstate = function8(arcValue,cstate)
```

[0042]
```
            print "path_fig3a: %4d %4d" % (arcValue,cstate)
```

[0043]
```
        cstate = 0
```

[0044]
```
        for arcValue in path_fig5b:
```

[0045]
```
            cstate = function8(arcValue,cstate)
```

[0046]
```
            print "path_fig3b: %4d %4d"  % (arcValue,cstate)
```

[0047] function8에 의해서 위에서 기술된 공식을 사용하여, 동일한 헤더 타입들을 포함하지만 상이한 순서로 배열되는 2 개의 패킷들에 대한 경로 해시 계산들 (path hash calculation) 이 2 개의 상이한 경로 해시 계산값들을 낳았음을 보인다.

[0048] 제 1 패킷에 대한 파스 경로 상에서, 도 3에 도시된 이뉴머레이션된 경로 값들을 사용하면, 경로 값들의 결과적인 시퀀스는 도 5a에 도시된 바와 같이 1,7,20 및 17이다. 위의 twoseq() 함수는 도 5a에 도시된 바와 같은, 1,132,86 및 58의 증분적 경로 해시 계산값들을 컴퓨팅하기 위해서 formula8() 함수를 사용한다. 제 1 패킷에 대한 최종 경로 해시 값으로서 58의 마지막 증분적 경로 해시 계산값을 사용한다.

[0049] 각 경로 해시 계산은 상태 변수 cstate를 상수 값 (본 경우에서는 제로) 으로 초기화함으로써 동일한 방식으로 시작된다. 파서가 지나가는 (traversed) 각 호에 대해서, 새로운 증분적 상태 값 cstate는 cstate의 값 및 각 호에 대한 호 값으로 formula8()을 호출함으로써 컴퓨팅된다. 위의 의사코드는 각 새로운 cstate 값을 컴퓨팅할 때에 cstate의 값 및 arcValue를 공급한다. 일 실시예에서, 오직 마지막 값이 후속 프로세싱을 위해서 보유되고 그 상을 지나갈 수 있다. 전술한 실례에서, formula8() 계산은 1,132,86,58의 증분적 경로 해시 계산값들을 산출한다. 오직 마지막 증분적 경로 해시 계산값 58이 제 1 패킷에 대한 마지막 경로 해시 값으로서 전달될 수 있다.

[0050] 제 2 패킷에 대한 파스 경로 상에서, 경로 값들의 결과적인 시퀀스는 3, 20, 17, 1이며, 이로써 3,150,90,44의 증분적 경고 해시 값들이 생성되며, 여기서 44는 후속 프로세싱을 위해서 사용되는 최종 경로 해시이다. 이러한 값들은 도 5b에서 도시된다.

[0051] 올바르게 선정된 함수 및 호 값들의 세트에 있어서, 파스 트리에 통한 매 유효한 경로는 고유한 식별자를 낳으며, 이는 이어서 나중에, 취해진 경로 및 어느 헤더들이 존재하였는지를 재구성하는데 사용될 수 있다. 이 단일 값을 저장하는 것은 모든 중간 값들을 저장하는 것보다 매우 보다 컴팩트 (compact) 하다.

[0052] 도 6은 경로 값 계산들을 사용하는 파서의 하드웨어 구현을 도시한다. 예시적으로, 도 6의 기능적 블록들은 ASIC로 구현될 수 있다. 데이터가 통상적으로 전체 패킷보다 작은 청크들 (chunks) 로 해서 파서에 도달한다. 파서는 패킷 내에서 어느 결정 지점 (decision point) 을 찾아봐야 하는지를 결정하는데, 이는 패킷의 시작 지점으로부터 다수의 바이트의 오프셋 (offset) 으로서 표현된다. 이 오프셋에서의 데이터가 키 생성 유닛 (501) 에 의해서 추출되고 현 상태와 함께 다음 상태 테이블 (502) 로 전송된다. 다음 상태 테이블은 통상적으로 TCAM (Ternary Content Addressable Memory) 또는 다른 연관 데이터 구조로서 구현될 수 있다. TCAM은 파스 트리 내의 각 호에 대한 엔트리를 갖는다. TCAM에서의 매칭은 다음 상태 및/또는 취해질 동작을 제공한다. 이 동작들은 패킷으로부터 추출될 필드들, 패킷으로부터의 필드들의 오프셋들 또는 특정 필드들이 존재하는지의 여부를 표시하는 플래그들와 같은, 추출된 데이터 구조 (506) 내에 기록될 데이터를 특정할 수 있다. 동작들은 또한 파서가 패킷을 계속 파싱해야하는지의 여부 또는 충분한 정보가 발견되어서 파싱이 종료될 수 있는지의 여부를 특정할 수도 있다.

[0053] 다음 상태 테이블 (502) 의 결과들은 블록 (503) 에서 현 상태를 업데이트하고 증분적 경로 값 계산을 수행하는데 사용될 수 있다. 종래 기술에서, 플래그들이 이 시점에서 설정되지만, 이 동작은 생략될 수 있는데, 그 이유는 각 경로가 고유한 식별자를 갖기 때문이다. 따라서, 이러한 식별자는 경로 컴포넌트들 (path components) 및 순서를 특정하는데 사용될 수 있다. 동작들이 파싱이 완료되었다고 표시하면, 최종 경로 값이 경로 값 테이블 (505) 로 전송된다. 그렇지 않다면, 현 상태 및 경로 값들이 키 생성 유닛 (501) 으로 다시 전송되고 여기서 파싱이 완료될 때까지 추가 탐색이 수행된다.

[0054] 입력 데이터로부터의 패킷 데이터 및 다음 상태 테이블 (502) 로부터의 동작들은 패킷으로부터 관심 데이터 필드를 추출하는데 사용되고, 이 관심 데이터 필드는 이어서 추출된 데이터 구조 (506) 로 전달된다. 파싱의 끝 부분에서, 경로 값 테이블 (505) 의 결과들이 이 구조에 부가되고, 이어서 패킷이 전송될 방식을 결정할 제어 경로에 전송된다.

[0055] 도 7은 본 발명의 실시예와 연관된 프로세싱 동작들을 예시한다. 먼저, 그래프가 구성된다 (700). 도 4의 동작들이 그래프를 구성하도록 사용될 수 있다. 다음으로, 수신된 패킷이 그래프 내의 고유 식별자와 연관된다 (702). 도 6의 프로세서는 이러한 동작을 구현하는데 사용될 수 있다. 특히, 다음 상태 테이블 (502) 이 경로의 호들을 증분적으로 지나가도록 사용될 수 있다. 이는 궁극적으로 인덱스 메모리 (505) 에 적용되는 최종 경로 값을 생성한다. 마지막으로, 수신된 패킷의 특성들이 고유 식별자에 기초하여서 재구성된다. 이 동작도 또한 도 6의 프로세서를 사용하여서 구현될 수도 있다. 특히, 추출된 데이터 구조 (506) 가 거쳐진 경로 (traversed path) 에 대해서 생성된다. 추출된 데이터 구조는 이 거쳐진 경로를 고유하게 식별하고 이로써 수신된 패킷과 연관된 패킷 헤더들을 특성화한다. 추출된 데이터 구조는 또한 연관된 플래그들 및 동작들을 가질 수 있다.

[0056] 몇몇 개선된 파서들은 다중 동시적 매칭 파서 (multiple simultaneous match parser) (때로 캥거루 파싱 (Kangaroo parsing) 으로 지칭됨) 를 사용하며, 여기서 다음 상태 테이블 (502) 은 단일 룩업 동안에 파스 트리 내의 다수의 호들을 매칭시킬 수 있다. 도 3의 경우에, 룩업마다 3 개의 매칭들을 수행할 수 있는 캥거루 파서는 이더넷 (401) 으로부터 노드들 VLAN (402) 및 VLAN (403) 을 통해서 IPv4 (404) 까지 이동할 수 있다. 동일한 단일 룩업이 잠재적으로 이더넷 (401) 으로부터 IPv4 (404) 까지 바로 이동할 수 있고, 경로 값의 목적은 취해진 각 경로에 대해서 상이한 경로 값을 가지는 것이므로, 경로 값 공식은 이러한 타입의 파서에 대해서는 이를 고려해야 한다.

[0057] 캥거루 타입 파서에서, 경로 값 공식은 간단하게 노드 식별자보다는, 취해진 각 가능한 경로에 대한 몇몇 정보를 포함한다. 도 3은 도 2로부터의 파스 트리이지만, 도 3에서는 설계 시에 각 호가 고유한 값을 갖도록 이뉴머레이션되었다. 다수의 호들이 단일 룩업으로 거쳐진다면, 경로 값 계산은 어느 호들이 거쳐지는 (traversed) 지를 결정하고 이뉴머레이션된 호 값들에 기초하여서 신규 경로 값을 컴퓨팅한다. 이 경우에, 경로 값은 룩업마다 3 개의 값들까지를 포함할 것이다.

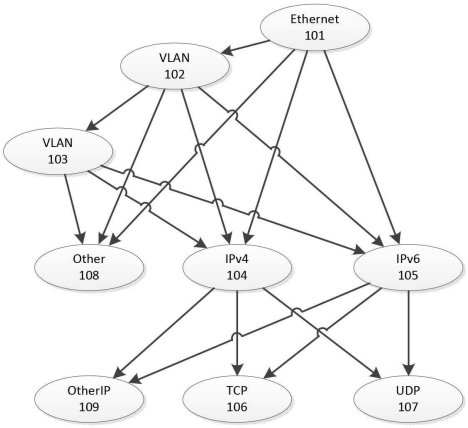
[0058] 경로 값을 컴퓨팅하는데 사용될 수 있으며 각 경로에 대한 고유한 이뉴머레이션 값들을 생성할 가능성이 높은 다른 값들이 존재한다. 예를 들어서, 다음 상태를 결정하는데 사용되는 이더넷 타입 값들 및 다음 상태 룩업 (next state lookup) 으로 전송되는 키 값들이, 노드 개수들 또는 호 개수들과 같은 내부 상태와 함께 다음 상태를 결정하는데 사용되는 패킷으로부터의 데이터의 조합이 그러한 것과 같이, 모두 구현가능한 후보들이다.

[0059] 설명을 위해서 제공된 위의 전술한 바들은 본 발명의 철저한 이해를 제공하기 위해서 특정 용어들을 사용하였다. 그러나, 특정 세부사항들은 본 발명을 실시하는데 필요하지 않음이 본 기술 분야의 당업자에게 자

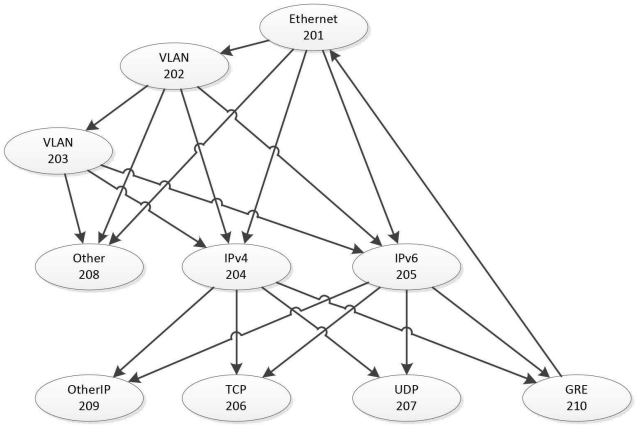
명할 것이다. 따라서, 본 발명의 특정 실시예들의 전술한 설명 부분들은 본 발명을 예시적으로 설명하기 위해서 제공되었다. 위의 설명들은 본 발명을 정확하게 개시된 형태로만 한정하는 것이 아니며; 수많은 수정 사항들 및 변경 사항들이 위의 교시사항들의 조명 하에서 가능하다. 위의 실시예들은, 본 발명의 원리들 및 그의 실제적 애플리케이션을 최상으로 설명하여서 본 기술 분야의 당업자가 본 발명 및 고려되는 특정 용도에 맞게 수정된 다양한 수정 사항들을 갖는 다양한 실시예들을 최상으로 활용할 수 있도록, 선정 및 기술되었다. 다음의 청구항들 및 이의 균등 범위가 본 발명의 범위를 규정한다.

도면

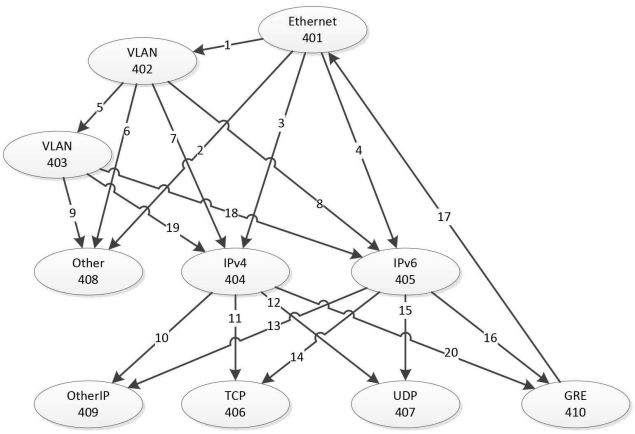
도면1



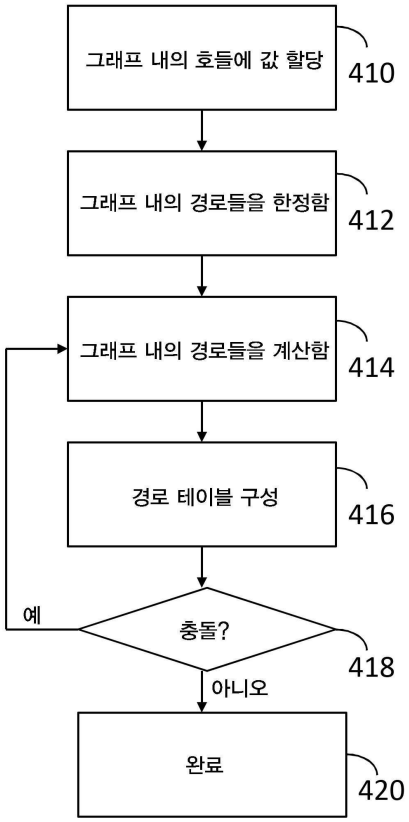
도면2



도면3



도면4



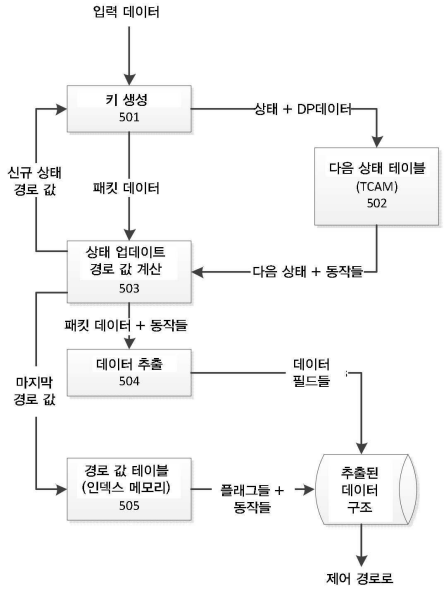
도면5a

From	To	Path Val	IncCalc
Ethernet	VLAN	1	1
VLAN	IPv4	7	132
IPv4	GRE	20	86
GRE	Ethernet	17	58

도면5b

From	To	Path Val	IncCalc
Ethernet	IPv4	3	3
IPv4	GRE	20	150
GRE	Ethernet	17	90
Ethernet	VLAN	1	44

도면6



도면7