논문 2007-44TC-10-19

IPv6 Lookup을 위한 효율적인 Priority TCAM Table 운영 알고리즘

(An Efficient Updating Algorithm for IPv6 Lookup based on Priority-TCAM)

홍 승 우*, 노 성 기*, 홍 성 백*, 김 상 하**

(Seung-Woo Hong, Sung-Kee Noh, Sung-Back Hong, and Sang-Ha Kim)

요 약

Internet의 빠른 성장으로 인한 IP Address의 고갈과 다양한 Application의 등장으로 Network은 IPv6로의 전환을 필요로 하고 있다. IPv6는 128-bits로 늘어날 주소 체계로 IPv4에 비해 LPM을 위한Lookup Table의 사이즈도 커질 것이기 때문에 대용량의 Lookup Table를 고속으로 처리하기 위한 방안이 필요하다. 이에 대한 해결책으로 하드웨어 기반의 고속의 TCAM을 사용하여 Lookup 성능을 향상시키려는 연구가 많이 되고 있긴 하지만, 최근 등장하고 있는 Priority-TCAM을 활용한 Lookup Table 구성방법에 대한 연구는 찾아 볼 수 없다. 본 논문에서는 Priority-TCAM과 기존 TCAM의 차이점을 기술하고 Priority-TCAM을사용하여 효율적으로 Lookup Table을 구성하고 운용할 수 있는 알고리즘을 기술한다.

Abstract

TCAM(Ternary content-addressable memory) has been widely used to perform fast routing lookup. It is able to accomplish the LPM(longest prefix matching) search in O(1) time without considering the number of prefixes and their lengths. As compared to software-based solutions, especially for IPv6, TCAM can offer sustained throughput and simple system architecture. However, There is no research for Priority-TCAM which can assign priority to each memory block. This paper addresses the difference of Priority-TCAM compared to the existing TCAM and proposes CAO-PA algorithm to manage the lookup table efficiently.

Keywords: TCAM, Priority-TCAM, CAO Algorithm

I. 서 론

일반적으로 라우터에서는 IP(Internet Protocol) 패킷(Packet)을 포워딩하기 위하여 입력되는 패킷의 목적지 주소를 추출하고, 추출된 목적지 주소를 키(Key)로 하여 포워딩 테이블을 Lookup하여 패킷이 포워딩 되어야 할 다음 홉(Next Hop)을 결정한다. 이러한 라우팅 룩업은 패킷이 지나쳐가는 모든 라우터 홉에서 수행되어 패킷은 최종적으로 출발지 노드(Node)에서부터 목적지 노드로 도달되게 되는 것이다. 1993년 채택된 CIDR (Classless Inter-Domain Routing) 방식의 주소 체계로 인해 라우터에서는 수많은 라우팅 Prefix 주소들이 라우팅 테이블에 저장되게 되었고, IP Lookup을 위해서는 입력 패킷의 목적지 주소와 가장 길게 일치(Longest Prefix Match)되는 Prefix 주소가 그 결과 값이 되도록 유지해야 했다. 전통적인 대용량의 라우팅 테이블에서 고속의 Lookup 성능을 위하여 해싱 (Hashing)이나 트라이(Trie)를 이용한 소프트웨어 기반의 방법이 많이 사용되었지만, IPv6에 적용하기에는 어렵고 또한 그 성

* 정회원, 한국전자통신연구원
(Electronics and Telecommunications Research Institute, ETRI)
** 정회원-교신저자, 충남대학교 공과대학 전기정보통신공학부 컴퓨터전공
(Department of Computer Science & Engineering, Chungnam National University)
접수일자: 2007년4월20일, 수정완료일: 2007년10월16일

능의 한계로 인해, 최근에는 고속의 TCAM(Ternary Content-Address Memory)을 이용한 하드웨어 기반의 IP Lookup 기술이 많이 연구되고 있다.

본 논문에서는 LPM을 위한 IP Prefix의 특성을 이용하여, 빠르고 쉽게 TCAM의 Route Table을 갱신하고 TCAM 메모리를 효율적으로 사용할 수 있는 Route Update 알고리즘을 제시하고자 한다. 이를 위하여 TCAM 메모리 영역을 우선순위(Priority)를 가진 일정한 크기의 Block으로 분할하고, 입력되는 각 라우팅 Prefix들을 Prefix 길이별로 Trie를 구성하여 Trie의 Depth 별로 우선순위를 할당 및 해당되는 우선순위 TCAM 메모리 영역에 저장하는 방법을 사용한다.

II. TCAM 기반의 IP Lookup 관련 기술

IPv6는 Address 체계가 128bits로 확장되어 IPv4에 비해 주소 체계가 복잡해졌고, 늘어난 주소 길이로 인해 라우팅 테이블이 사이즈가 더욱 커질 가능성이 있어 고속의 Lookup 기술이 요구된다.

최근 고성능의 TCAM은 초당 십억 패킷을 처리할 수 있기 때문에 TCAM을 사용한 Lookup 기술 개발이 많이 사용되고 있다. 하지만 이러한 높은 Lookup 성능과 함께 다음과 같은 3가지의 단점이 있다. 첫째로 TCAM은 일반적인 SDRAM에 비해 상당히 고가이기 때문에 낭비없이 효율적으로 사용해야 한다. 둘째는 Key 비교를 내부 모든 메모리에 병렬적으로 수행하기 때문에 많은 파워를 소모한다. 따라서 TCAM 메모리 용량을 무한정 높이기 힘들다. 셋째는 Longest Prefix Match을 위해서는 모든 Prefix 들이 Prefix 길이의 오름차순으로 정렬되어 있어야 하기 때문에 매번 Prefix가 저장될 때마다 전체 Prefix Table을 정렬해 주어야 하는 오버헤드가 생긴다.

이와 같은 TCAM의 단점들을 극복하고 효율적으로 TCAM을 사용하고자 하는 연구는 크게 두 가지 부류로 나눌 수 있다. 한 가지 부류는 Routing Table Compaction 방법을 이용하여 Routing Table 사이즈를 줄여 TCAM 메모리 공간에 대한 요구를 줄이고, 그에 따라 전력소모 요구도 줄이는 것이다. 그 예로서 [9]에서는 Pruning과 Mask Extension을 이용한 compaction 알고리즘을 제시하였고, [10]에서는 Prefix overlapping과 Prefix minimization 을 이용하는 TCAM 구조를 제시하고 있다. 하지만 Routing Compaction은 Routing Table 사이즈를 줄일 수 있다고 하지만, Route Update가 복잡하고 느리다는 문제가 있다. 다른 한 가지 부류는 TCAM의 높은 Update Complexity을 극복하고자 하는 것이다. 대표적인 Update 알고리즘은 Prefix-length ordering constraint을 활용한 PLO-OPT 알고리즘과 Prefix-ancestor-chain을 활용한CAO-OPT 알고리즘이다. PLO-OPT 알고리즘은 Prefix update시 Prefix 정렬을 위해 최악의 경우 L/2(L=Maximum Prefix Length, eg. IPv4=32, IPv6=128)의 메모리 이동을 필요로 한다. 트라이(Trie) 구조를 활용하는 CAO-OPT 알고리즘은 Prefix 정렬을 위해 O(AD/2)의 Complexity(AD는 Prefix 트라이에 존재하는 모든 Prefix Chain에 대한 평균 길이) 비용이 필요한데, 일반적으로 Prefix chain의 길이는 Prefix length보다 훨씬 짧을 가능성이 높으므로 좋은 성능을 보여준다고 할 수 있다.

한편, 최근 TCAM의 높은 Update Complexity를 근원적으로 제거하기 위해 Priority-TCAM이라는 새로운 형태의 TCAM이 개발되고 있다. Priority-TCAM은 TCAM의 메모리 영역이 일정한 크기의 블록(Block)으로 구분되어 있으며, 각 블록에 우선에 Priority를 할당할 수 있다. [그림 1]은 Priority-TCAM의 구조와 동작의 간단한 예를 보여주고 있다. TCAM의 메모리 영역이 3개의 메모리 슬롯(Slot)을 가진 Block 0, Block 1, Block 2로 구성되어 있고, Block 0에는 Priority 1, Block 1에는 Priority 2, Block 2에는 Priority 2가 할당되어 있다. [그림 1]과 같이 각 블록에 Routing Prefix들이 저장되어 있다면, 입력되는 패킷에 대하여 Lookup Match가 되는 Routing Prefix는 0, 6이다. 이 경우 일반적인 TCAM에서는 상위의 주소에 해당되는 0이 선택되겠지만, Priority-TCAM에서는 우선순위 인코더에 의해 Lookup Match된 Routing Prefix가 저장된 Block의 Priority를 비교하여 Priority가 높은 6을 그 결과로 한다.

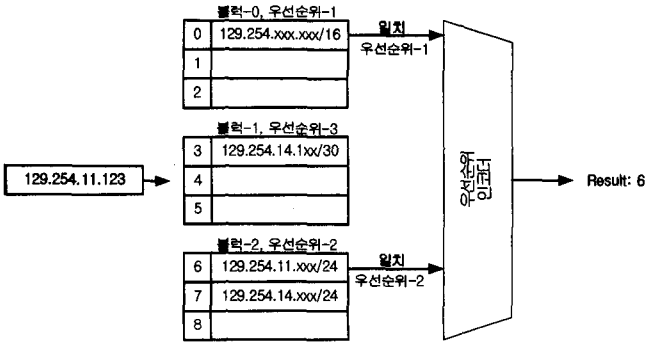


그림 1. Priority-TCAM의 동작방식
Fig. 1. Logical action of Priority-TCAM.

이와 같이 Priority-TCAM은 기존의 TCAM과는 달리 Routing Table을 구축할 때 Prefix length가 긴 순서대로 Priority를 할당하기만 하면 전혀 정렬을 위한 Update 비용이 들지 않는다. 그러나 Priority-TCAM은 메모리를 일정한 크기의 블록으로 나누기 때문에 필연적으로 Memory Fragmentation이 발생한다. 즉, 빈번히 발생하지 않는 Prefix length의 몇 개의 Routing Prefix가 하나의 Block을 점유해 버리는 경우가 생기는데, IPv6 Routing Table에서는 최악의 경우 128개의 Block에 Fragmentation이 발생할 수도 있다. 따라서 Priority-TCAM에서 Memory Fragmentation을 줄여 메모리 효용성을 높이고, Update Performance를 낮추지 않는 방법에 대한 연구가 필요하다.

Ⅲ. 제안하는 Priority-TCAM 운영 알고리즘

1. CAO Priority 할당 알고리즘

Routing Prefix에 Priority를 할당하는 가장 쉬운 방법은 PLO(Prefix-Length-Ordering)를 이용하는 것인데 본 논문에서는 PLO-PA(Priority Assignment) 알고리즘이라고 부르도록 한다. PLO-PA 방식은 Prefix의 길이와 Priority를 단순히 1:1로 대응시키는 것인데, 예를 들어 PL이 64이면 P=64, PL이 24이면 P=24 식으로 할당하는 것이다. PLO-PA는 앞서 언급한 것처럼 Priority-TCAM의 Fragmentation 문제를 발생시킨다. 하나의 Block에 1K의 Routing Entry를 저장할 수 있는 Priority-TCAM에 IPv6 Lookup Table을 구성한다면, 최악의 경우 (128x1024)-128 슬롯의 메모리가 낭비되는 것이다. 본 논문에서는 PLO-PA의 문제점을 해결하기 위해 CAO(Chain-Ancestor-Ordering)을 활용하는 CAO-PA 알고리즘을 제시한다. 우선 CAO-PA 알고리즘의 기술을 위해 다음과 같은 용어를 정의한다.

CAO(Pi, Pj): Pi와 Pj가 서로 중첩되어 있고, Prefix Length가 다르다.
Anc(P): P와 CAO(Pi, P)관계가 성립되고, P보다 Prefix Length가 짧은 어떤 Prefix Pi
Des(P): P 와 CAO(Pi, P) 관계가 성립되고 P 보다 Prefix Length가 긴 어떤 Prefix
PC(Pi): CAO관계에 있는 Prefix Pi들의 Prefix 집합
PC(Pi > Pj): PC(Pi)에 속하는 Pi들을 Anc(P) 순으로 정렬한 Prefix Chain.
CAO-Tree: PC(Pi) 이용하여 구성한 Tree, Anc(P)가 Des(P)보다 상위에 위치한다.
Level(P): CAO-Tree에서 P가 위치하는 곳의 Tree Depth

CAO-PL 알고리즘은 서로 다른 두 Prefix Pi, Pj가 서로 CAO(Pi, Pj) 관계를 가지지 않으면 서로 어떠한 우선순위 관계도 성립하지 않는다는 Prefix의 특성을 기반으로 한다. [그림 2]의 예제에 나타나 있는 것처럼, P1과 P2는 Prefix 길이 16만큼 중첩되어 있기 때문에 CAO(P1, P2)관계가 성립된다. 또한 P1은 P3, P5 및 P4와도 CAO 관계를 가진다. 하지만 P3과 P4는 32 bits를 비교해보면 3ffe:aabb, 3ffe:aacc로 중첩되지 않아 CAO(Pi, Pj)는 성립되지 않으며, 이것은 P3과 P4는 Lookup시에 동시에 Match 될 수는 없기 때문에 두 Prefix가 동일한 Priority를 가져도 상관없다는 것을 의미한다. 따라서 [그림 1]과 같이 CAO(Pi, Pj) 관계를 이용하여 PC(P1 > P2 > P3 > P5), PC(P1 > P2 > P4) 두 개의 Prefix Chain을 가진 CAO-Tree로 표현 할 수가 있다.

[그림 3]에 CAO-Tree에 의해 Routing Prefix가 저장되는 TCAM 우선순위 블록이 나타나있다. CAO-Tree의 깊이에 따라 Level(P)가 정해질 수 있는데, 각

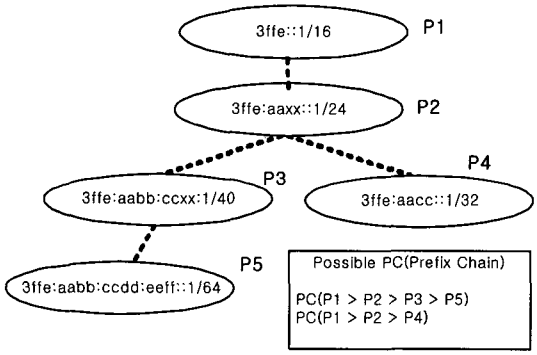


그림 2. CAO-Tree 예제
Fig. 2. CAO-Tree example.

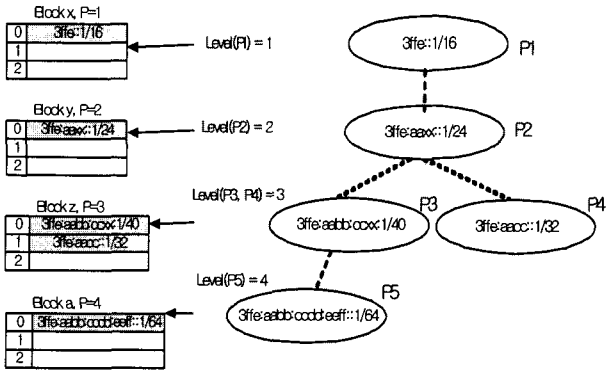


그림 3. CAO-Tree를 이용한 Priority 할당 방법
Fig. 3. Priority assignment using CAO-Tree level.

Level(P)에 속한 Routing Prefix P는 Level 번호와 동일한 우선순위를 할당 받고, 해당되는 우선순위의 TCAM 블록에 저장된다. [그림 3]의 예에 나타나 있는 것과 같이 P3과 P4는 서로 다른 Prefix Length를 가지더라도 LPM을 위한 순위 관계에 영향을 받지 않기 때문에 같은 우선순위 블록에 저장이 가능한 것이다. 이와 같은 방법으로 서로 다른 Prefix Length를 가지더라도 동일한 우선순위 블록에 저장 할 수 있기 때문에 TCAM 메모리의 단편화 현상을 줄이고 메모리의 효율성을 높일 수 있는 것이다.

2. 룩업테이블 변경 알고리즘

[그림 4]에 TCAM 룩업 테이블 운용 중에 새로운 라우팅 프리픽스가 추가 될 경우의 알고리즘을 기술하였다. 새로운 라우팅 프리픽스 P가 입력이 되면, CAO-Tree의 루트의 첫 번째 아들 노드부터 순차적으로 P와의 CAO 관계를 검사해 나간다. [그림 4]의 알고리즘의 마지막 조건문과 같이, CAO-Tree의 현재 검사중인 레벨의 모든 형제노드들 중에서 CAO(현재노드, P)의 관계가 성립하는 노드가 존재하지 않는다면, 입력 프리픽스 P는 그 레벨의 새로운 형제노드가 되며, 그 레벨에 해당되는 우선순위를 할당 받고, 할당 받은 우선순위의 TCAM 블록에 저장된다. 만약, CAO(현재노드, P) 관계가 성립하고, 현재노드가 Anc(P)이면, 현재노드의 다음 후손(아들 노드)와 재귀적으로 CAO(다음 후손노드, P) 관계를 검사한다. 반대로, CAO(현재노드, P) 관계가 성립하고 현재노드가 Des(P)이면, 입력 프리픽스 P의 위치는 P가 포함되는 PC의 중간부분이 되기 때문에 그 하위에 위치한 기존의 후손 노드들은 CAO-Tree에서 한 레벨씩 내려가게 된다. 이에 따라 한 레벨씩 증가된 후손 노드들은 기존에 저장되어 있던 TCAM 우선순위 블록에서 한 레벨씩 내려간 TCAM 우선순위 블록으로 이동시켜야 한다.

[그림 5]에서 프리픽스 P6가 새롭게 입력되면, P6는 P4의 부모 노드가 되기 때문에 기존의 P4 자리에 위치하고 P4는 P6의 아들 노드로서 한 레벨 아래로 내려가게 된다. 따라서 P4는 블록-z에서 블록-a로 이동 저장되고, P6는 블록 z의 P4가 이동한 빈자리에 저장되는 것이다. 본 논문에서 제안하는 CAO-PA 알고리즘은 하나의 우선순위 TCAM 블록에 서로 다른 PL을 가진

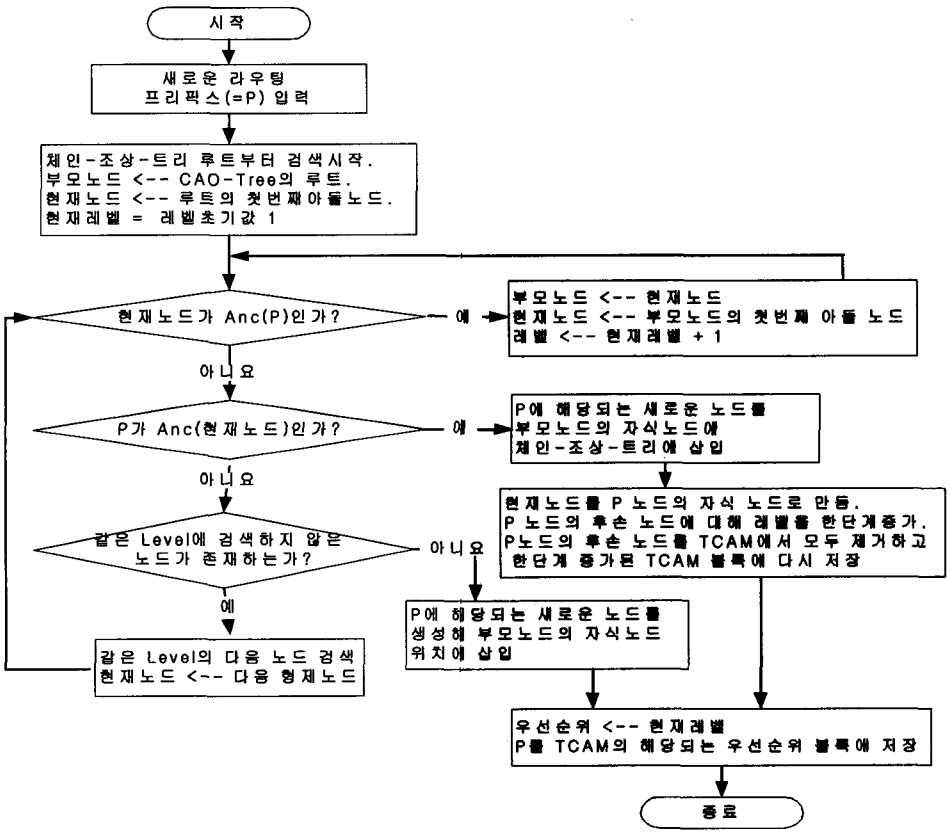


그림 4. 룩업 테이블 변경 알고리즘

Fig. 4. Flow chart of the proposed algorithm.

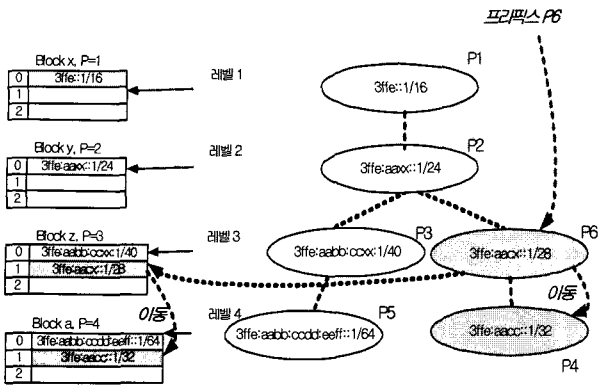


그림 5. 우선순위 블록의 이동 방법

Fig. 5. Algorithm for updating the priority block.

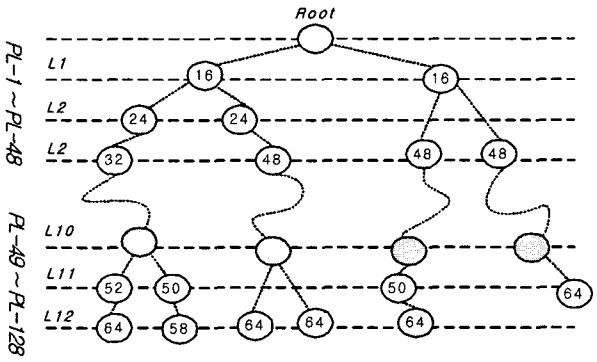


그림 6. IPv6를 위한 다단계 CAO-Tree

Fig. 6. Multi-level CAO-Tree for IPv6.

프리픽스들을 중첩 저장하기 때문에 TCAM 메모리의 효율성을 높일 수 있지만, CAO-Tree에서 낮은 레벨에 위치하는 노드의 입력 또는 삭제시 하부의 모든 Des(P)의 노드들의 위치 이동이 일어나기 때문에 룩업 테이블 변경 알고리즘의 성능 저하를 가져올 가능성이 있다.

룩업테이블 변경의 성능 향상을 위하여 다단계 (Mutli-Tier) CAO-Tree를 이용한 CAO-PA 알고리즘을 사용할 수 있다. [그림 6]는 IETF RFC 2374에 기술된 IPv6 주소 할당 방식에 대한 권고를 참고로 하여 두 단계의 CAO-Tree를 형성하였다.

MT-CAO-PA 방법을 사용하게 되면, 높은 레벨에 위치하는 노드의 어떠한 입력 또는 삭제도 낮은 레벨에 속하는 노드에 영향을 주지 않기 때문에 룩업테이블 변경에 따른 성능 저하를 탄력적으로 줄일 수 있다.

Ⅳ. 시뮬레이션 및 결과

본 장에서는 CAO-PA 알고리즘과 PLO-PA 알고리즘을 구현하여 두 알고리즘의 TCAM 메모리의 효율성 및 성능을 비교한 결과를 기술한다.

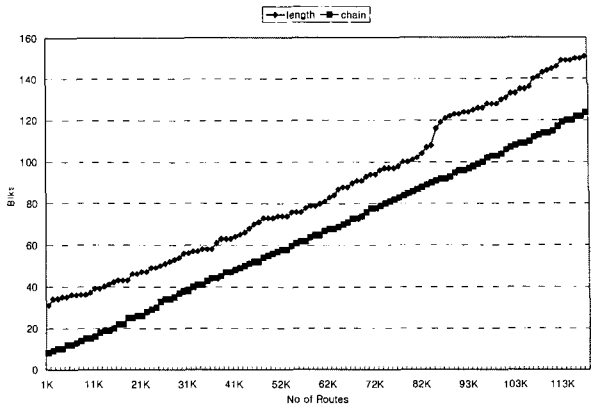


그림 7. CAO-PA vs PAO-PA 알고리즘의 TCAM 블록리소스 사용량 비교

Fig. 7. Comparison of blocks resource usage.

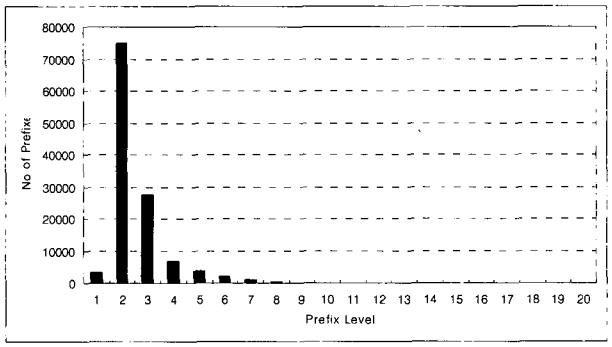


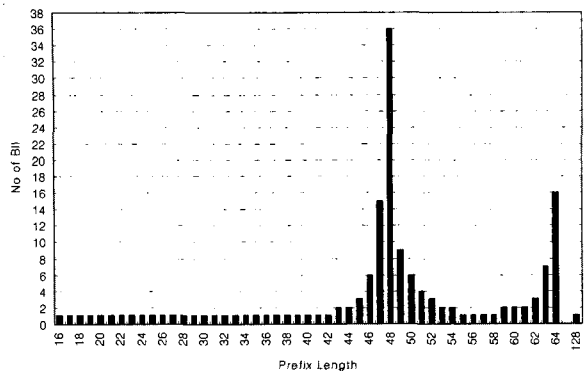
그림 8. 입력 Prefix Data의 Prefix Chain 분포도

Fig. 8. Prefix chain distribution of input data.

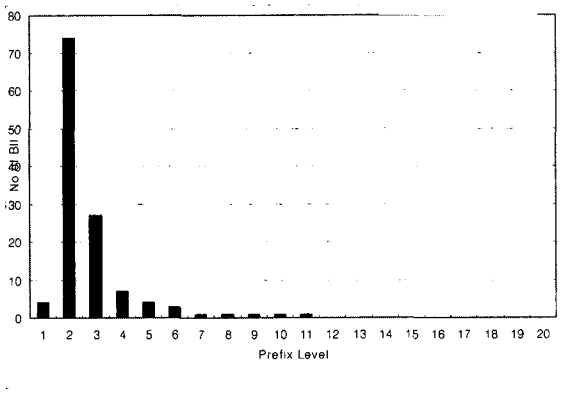
기존의 다른 연구에서는 실제 IPv4의 BGP Route 데이터베이스를 사용하여 Simulation하는 경우가 많았지만, IPv6의 경우에는 IPv6 Network이 실제 Network에 널리 사용되고 있지 않고 있기 때문에 Route Database를 얻기가 힘들다. 따라서 본 논문에서는 [11]에서 제공하는 V6Gene을 사용하여 약 10만개의 IPv6 프리픽스를 생성하여 Simulation의 입력으로 사용하였다. V6Gene은 IPv6 관련 RFC 및 초기 IPv6 프리픽스의 패턴을 분석하여 가상의 IPv6 Prefix Database를 생성해 주어 IPv6 관련 시험에 사용할 back data를 제공한다.

[그림 7]은 입력 데이터에 대하여 PLA-PA와 CAO-PA의 TCAM 블록 사용량을 나타내는 그래프이다. CAO-PA 알고리즘이 PLO-PA 알고리즘에 비하여 적은 수의 TCAM 블록을 사용하고 있음을 볼 수 있다.

[그림 8]은 CAO-PA 알고리즘에 의해 나타난 입력 Prefix Data의 Prefix Chain의 분포도를 나타낸다. 가장 긴 Prefix Chain의 길이는 11이지만, 99%가 6이하의 길이를 가지고 있으며, 62%가 Chain의 길이가 2이다. 이는 IPv6로 Prefix 길이가 길어졌지만, CAO-Tree의



(a)



(b)

그림 9. (a) PLO-PA에서 사용된 PL별 블록사용 분포
(b) CAO-PA에서 사용된 Level별 블록사용 분포

Fig. 9. (a) Block usage distribution of the PLO-PA algorithm.
(b) Block usage distribution of the proposed CAO-PA algorithm.

표 1. CAO-PA vs PLO-PA 알고리즘의 TCAM 블록 사용 효율성 비교

Table 1. Comparison of block usage efficiency.

	Total Blocks	Fragmented Blocks	Utilization
PLO-PA	154	50	67%
CAO-PA	124	12	91%

표 2. CAO-PA 알고리즘의 Update로 인한 블록이동 횟수 및 블록이동 비율

Table 2. Movement rate of the proposed CAO-PA algorithm.

Total Movement	Total Route	Move Rate
6784	119614	0.056

Depth가 그다지 길어지지 않음을 나타낸다.

[그림 9. (a), (b)]는 PLO-PA 알고리즘에서 PL별로 사용된 TCAM 블록과 CAO-PA에서 사용된 Level별로 사용된 TCAM 블록의 분포를 나타낸다. PLO-PA가 CAO-PA에 비하여 훨씬 넓은 사용 분포를 나타내며 Fragment 블록이 많음을 확인할 수 있다.

[표 1]은 PLO-PA와 CAO-PA의 TCAM 블록 Fragmentation을 보여준다. [그림 9 (a), (b)]에 나타난 바와 같이 훨씬 넓은 사용 분포를 가진 PLO-PA의 사용 효율성이 낮음이 나타나 있다.

[표 2]는 CAO-PA 알고리즘을 이용했을 경우 TCAM 블록 이동이 일어난 회수를 나타낸다. 전체 Route에 대해 이동 비율이 0.056으로 TCAM 블록 이동으로 인한 성능 저하가 그리 크지 않음을 나타낸다.

V. 결 론

TCAM 메모리 영역이 일정한 크기의 우선순위를 가지는 Priority-TCAM은 기존의 TCAM과는 달리 Routing Prefix의 정렬(Sorting)보다는 우선순위를 할당하는 방식이 더욱 중요하다고 할 수 있다. 본 논문에서는 제시하는 CAO-PA 알고리즘은 Routing Prefix간의 CAO 관계를 이용하여 CAO-Tree를 구성하고, CAO-Tree의 Level에 따라 우선순위를 할당하는 방식을 사용한다. CAO-PA 알고리즘은 서로 다른 길이의 Prefix Length를 가진 Prefix들이 같은 우선순위 블록에 저장될 수 있도록 함으로서 Priority-TCAM의 문제점인 Block Fragmentation을 줄여준다. 또한 MT-CAO-Tree의 변경 알고리즘을 제시하여 Route Table 변경으로 인한 성능 악화가 최소화 되도록 하였다. 실제 Simulation을 통해 CAO-PA 알고리즘이 TCAM 메모리 이용률을 두드러지게 높이는데 반해, Lookup Table 변경에 따른 성능 저하는 그리 크지 않음을 볼 수 있다.

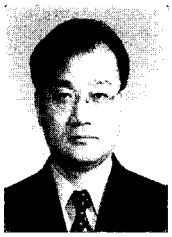
참 고 문 헌

[1] Vince F, et al. Classless Inter-Domain Routing (CIDR): an address assignment and aggregation strategy (RFC 15119)
[2] R. Hinden and S. Deering. IP Version 6 Addressing Architecture, RFC 2373
[3] R. Hinden, M.O'Dell, S. Deering, An IPv6 Aggregatable Global Unicast Address Format, July 1998.
[4] Devavat Shah, Gupta, P. Fast Updating Algorithms for TCAM. IEEE Micro January/February 2002.

[5] Ravikumar V.C, Rabi N. Mahapatra Texas A&M University, TCAM Architecture for IPLookup Using Prefix Properties, IEEE Micro March-April 2004.
[6] R. Panigraht and S. Sharma, Reducing TCAM Power Consumption and Increasing Throughput, Proc. 10th Symp, IEEE CS Press, 2002, p.107-112
[7] H.Liu, Routing Table Compaction in Ternary CAM, IEEE Micro, Jan-Feb.2002, vol.22, no.1, pp.58-64
[8] Zhiyong Liang, Jianping Wu, Ke Xu, A TCAM-based IP Lookup Scheme for Multi-nexthop Routing, ICCNMC'03
[9] Kai Zheng, Bin Liu, V6Gene: A Scalable IPv6 Prefix Generator for Route Lookup Algorithm Benchmark, AINA'06
[10] P. Gupta, S.Lin, and N. McKeown, Routing Lookups in Hardware at Memory Access Speed, Proc. Of IEEE INFOCOM'98, San Francisco, April 1998, pp.1240-1247
[11] Miguel A.Ruiz-Sanchez, Ernst W.Biersack, Walid Dabbous, ,Survey and Taxonomy of IP Address Lookup Algorithms, IEEE Network, March/April 2001.
[12] V.Srinivasan and G.Varghese, "Fast Address Lookups using Controlled Prefix Expansion", Proc,.ACM Sigmetrics'98, Une 1998, pp.1-11

저 자 소 개

홍 승 우(정회원)
1999년 경성대학교 전자계산학과 학사 졸업.
2001년 부산대학교 전자계산학과 석사 졸업.
2007년 현재 한국전자통신연구원 선임연구원.
<주관심분야 : QoS, MIP, 무선통신>



홍 성 백(정회원)
1982년 광운대학교 전자통신 공학과 학사 졸업.
1990년 연세대학교 전자공학과 석사 졸업.
1982년 현재 한국전자통신연구원 FMC기술팀 팀장.
<주관심분야 : 통신공학, BcN, USN>

노 성 기(정회원)
1990년 한양대학교 산업공학과 학사 졸업.
1992년 포항공대 산업공학과 석사 졸업.
2006년 충남대학교 컴퓨터과학과 박사 졸업.
2007년 현재 한국전자통신연구원 책임연구원
<주관심분야 : QoS, Performance Analysis>

김 상 하(정회원)
1980년 서울대학교 화학과 학사 졸업.
1989년 Uniiv. of Houston (컴퓨터) 석사 졸업.
1989년 Uniiv. of Houston (컴퓨터) 박사 졸업.
1992년 현재 충남대학교 전기정보통신공학부 교수
<주관심분야 : 이동통신, 인터넷통신>