Proceedings of the Sixth IASTED International Conference
Communications, Internet, and Information Technology
July 2–4, 2007, Banff, Alberta, Canada
ISBN Hardcopy: 978-0-88986-673-7 / CD: 978-0-88986-674-4

A NOVEL COMPACTION SCHEME FOR ROUTING TABLES IN TCAM TO ENHANCE CACHE HIT RATE

Ruirui Guo and José G. Delgado-Frias
School of Electrical Engineering and Computer Science
Washington State University
Pullman, WA 99164 USA
{rguo, jdelgado}@eecs.wsu.edu

ABSTRACT
Routing table lookup is an important operation in packet forwarding. The speed at which this operation is performed has a great influence on the overall performance of the network processors. Routing tables are usually stored in main memory with a large access time. Consequently, small fast cache memories have been used to improve access time. In this paper, we propose a novel routing table compaction scheme to reduce the size of routing table stored in ternary content addressable memory (TCAM) and improve cache hit rate. Two or more routing entries are compacted into one using *don't care* element in TCAM. A small compacted routing table helps to increase cache hit rate; this in turn provides fast address lookups. We have evaluated the compaction scheme through extensive simulations involving IPv4 routing tables and traces. The original routing tables have been compacted over 60%. The cache hit rate has improved by up to 20% over the original tables depending on both cache size (number of cache entries) and traces. We have also analyzed the impact of port errors caused by caching technique, and chose sampling technique to alleviate this problem. The proposed scheme has been extended to IPv6 with similar results.

KEY WORDS
High speed internet, compaction, route lookup.

1. INTRODUCTION

IP routing plays an important role in the forwarding of packets through the Internet. It decides how and where to deliver incoming packets to the appropriate output interface of a router using the packet's destination address. The process is performed by looking up entries in a routing table, which contains information relating to other networks and hosts in the Internet. Each entry in the routing table comprises an address prefix, a forwarding address, and the interface to which the packets should be delivered when their address prefix matches.

With the continuous growth of the Internet, higher demands are placed on the IP routing in terms of speed; in particular, the growth of link bandwidth requires increasingly fast IP routing table lookups. A router needs to handle roughly 1,000 packets per second for every 10^6 bits per second of line rate [1]. Therefore, 10M routing lookups per second are needed for a current route with the line rate of 10 Gb/s (OC-192). Moreover, in the near future, the line speed will grow towards 40 Gb/s (OC-768) with the continued technological advances in optical and electronic devices [2]. Such a high line speed requires fast lookups to match. In addition, routing tables are becoming larger, thus a large memory is usually required to store such tables in a local router. This in turn may restrict the lookup speed since the complete routing table is stored in main memory with slow access time. In order to speed up the address lookup process, a cache architecture [3] is generally used in network processors. A cache is a small memory with a short memory access time. Only recently used forwarding information is held in cache. Having a large cache hit rate yields a short average access time. There have been several proposed schemes to enhance caching effectiveness [4,5,6,7]. Routing table compaction is a scheme that has the potential of improving the cache hit rate. In this scheme the table is reduced in size by combining some entries of this table. Thus, an entry in the compacted routing table may contain more than one entry of the original table. Some researchers have proposed to use Espresso algorithm to minimize IPv4 routing tables [8,9]. Espresso is a program for two levels Boolean function minimization. This technique reduces the overall number of product terms by up to 48%. In this paper, we propose one new compaction scheme for the routing tables. The scheme is to be implemented in a ternary content addressable memory (TCAM) taking advantage of TCAM feature of having a *don't care* element. Using the TCAM caching architecture and the compaction scheme, the cache achieves a higher hit rate without much cost overhead. This combination of TCAM and the compaction scheme translates into fast address lookups.

And this scheme is easy to be scaled to IPv6 with 128 bit addresses.

This paper is organized as follows. In Section II, a cache architecture for address lookups is described. The longest prefix matching technique and the routing information are also briefly explained. Our novel compaction scheme is introduced in Section III. We present the simulation results of IPv4 in Section IV. In the same section we introduce and evaluate the sampling technique to alleviate port error problems and also include the analysis of our scheme implemented on IPv6 routing. In Section V, some concluding remarks are presented.

2. CACHE AND ROUTING INFORMATION

In this section we provide a brief description of the cache architecture that can be used for table lookup routing. We also describe the TCAM features that are of particular importance in the description of our compaction scheme. Finally, we describe the routing information used for our simulations.

2.1 Cache Architecture for IP Routing

Cache memories are widely used in computer systems. In a program (with many memory accesses) it is very likely that the same data in memory is accessed multiple times in a short time period; this is called temporal locality. Thus, a small and fast memory can be used to reduce lengthy memory access times. Studies have shown that the network packet streams indeed have temporal routing locality. That is, a routing entry accessed before it is possibly referenced again in a short period of time. This feature allows caches to be used in IP address lookups. Consequently, more active forwarding entries are saved in cache; this in turn has the potential of making lookup faster.

The process of lookups is associated with a destination address. When a router receives a packet from one of its interfaces, the destination address in this packet is extracted, and compared with current entries in the cache using a *Longest Prefix Matching* (LPM) mechanism [7]. The basic organization of the cache is depicted in Figure 1. If there is a matching entry in cache, this packet will be delivered to the interface specified by this entry. If a miss occurs, an address search is made in the large routing table stored in memory. Subsequently, the matching entry is written back into cache. Here, we use *Least Recently Used* (LRU) replacement policy, in which the route entry that has not been accessed for the longest time will be evicted from the cache.

The average memory access time (AMAT) of the cache organization is determined as follows [10].

$$AMAT=(H_{cache}\times AT_{cache})+((1-H_{cache})\times AT_{memory}) \quad (1)$$

where H_{cache} is the cache hit rate, AT_{cache} is the cache access time, and AT_{memory} is the routing table access time. Current technology and memory implementations indicate that cache (using SRAM) has an access time that is 8 to 16 times faster than main memory (using DRAM) [10]. It is clear that a cache technique is needed to obtain small average access time. A small size cache usually yields low access time (AT_{cache}) while temporal locality helps to increase hit rate (H_{cache}). This paper deals with the compaction schemes to increase hit rate while keeping a small size cache memory.

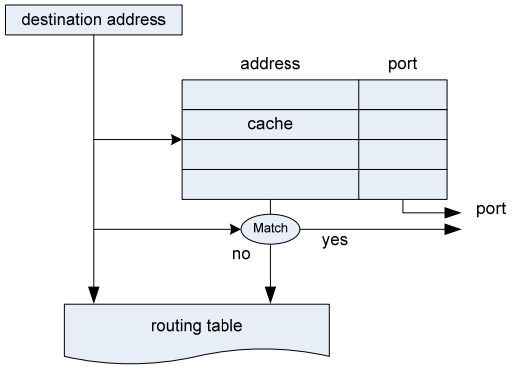

Fig. 1. A Cache Organization for IP Address Lookup.

2.2 Ternary Content Addressable Memory (TCAM)

When we implement LPM lookup mechanism in a conventional memory, we need to store both a network address and its address mask in each route entry. Given a destination address, perform bit-wise *AND* operation between this destination address and its address mask in each entry, and then check if the result is equivalent to the network address in the same entry. A ternary content addressable memory makes this process more direct by its special features. Different from the usual memories, each cell of TCAM stores one of three states: *0*, *1* and *don't care* (*) [11]. The TCAM makes the implementation of LPM much easier. All bits with the low order in an address below the prefix boundary are replaced with *don't care*. These bits are ignored when there is a comparison with the TCAM entry. TCAMs are implemented to perform a parallel search of all entries. The port associated with the longest prefix length that matches the search is chosen using a priory encoder [12]. An organization for the proposed associative cache is shown in Figure 2.

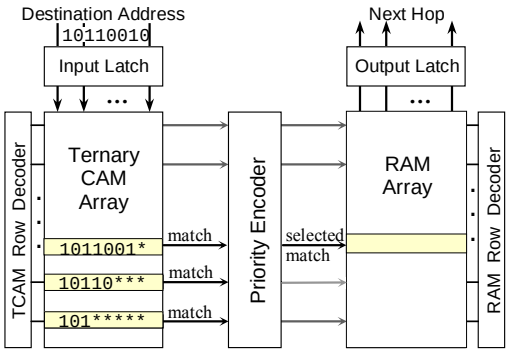

Fig. 2. A TCAM Cache Organization.

In Figure 2 an example of an address matching in this TCAM cache is included. Given a packet with a destination address *10110010,* three matching entries *1011001*, 10110**** and *101****** are found in the routing table. This packet will be delivered to the port associated with the first entry by the LPM mechanism which is implemented by means of a priority decoder.

2.3 Routing Information

Evaluations of our proposed scheme are based on four IPv4 trace files which are downloaded from the Measurement and Analysis on the WIDE Internet (MAWI) Working Group http://tracer.csl.sony.co.jp/mawi/ [13]. Each trace file has 2 million destination addresses. Analyzing these destination addresses in each trace file, we found that there are only several thousands unique addresses. The information about these trace files is shown in Table 1. The trace file number is the one given at the website; we have labeled these files from W to Z for IPv4 traces to ease file references. We also obtain IPv4 routing tables from the website of University of Oregon Route Views Archive Project http://archive.routeviews.org/[14]. In order to keep these tables neat, we delete those redundant entries in tables.

Table 1. IPv4 Traces.

Trace files	Number of unique addresses
(W) 060524	70627
(X) 060609	76533
(Y) 060615	63485
(Z) 060630	80853

3. ROUTING TABLE COMPACTION

The IP address lookup operation is a time-consuming task. Since the destination address of an arriving packet does not carry any prefix length information, the destination address is required to be compared with all the route entries in the routing table when implementing LPM. As we mentioned before, a TCAM can perform a parallel search to reduce the lookup time. A large TCAM may be too expensive to be a feasible solution. In addition, a large common memory usually slows down the lookup process. This in turn creates a need for investigating compaction schemes and fast cache memory solutions. If the number of entries in the routing table can be compacted, the table can be contained into fewer TCAM chips. Thus, the process of address lookup could be improved effectively and economically with a cache organization. In this section, one compaction scheme similar to Espresso algorithm is introduced. Then the new compaction scheme is presented.

3.1 Existing Compaction Scheme (Scheme C1)

This scheme is similar to those in [8, 9]. The basic steps are as follows.

- The entry addresses with the same destination port are considered for compaction.
- Two addresses that differ by only one bit including *don't care* (*) are candidates for compaction.
- If these conditions are satisfied, the entries with these two addresses can be combined into one by using *don't care* (*) to replace the bit.
- The two entries that have been combined are removed from further consideration
- The new entry is included in the potential entries for further compaction.

Assume that there are four destination addresses that share the same port number as shown in Figure 3. The addresses in entry *a* and *b* are the same except for the last bit. These two entries can be combined into a new entry 00100110*. The same compaction can be performed on the entries *c* and *d*. The updated routing table has two entries after the first compaction. In this example, these two entries can be compacted since they have only one bit that is different. The compaction process continues until there are no potential entries to be compacted. In our example, those four entries are compacted into one with 2-bit '*' using this scheme.

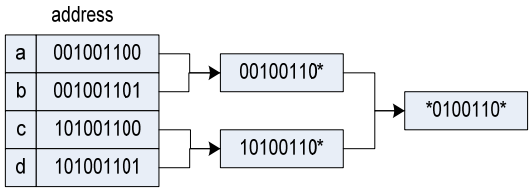

Fig. 3. An Example of Compaction Scheme C1.

3.2 Proposed New Compaction Scheme

We have developed the novel compaction scheme which is explained below. It combines many more entries than scheme C1.

3.2.1 Compaction with non-existing entries (Scheme C2)

This scheme uses the compacted entries produced by C1. The improvement is due to the way *don't care* (*) is handled. Below are the additional steps that are needed to implement scheme C2.

- Two addresses with the same destination port that have only one different bit excluding *don't care* (*) are candidates for additional compaction
- If there is no other address with a different destination port that can be compacted with one of the two addresses in the previous step, then the compaction can be performed.
- If the previous step is not possible the current compaction is abandoned.

Figure 4 shows an example where entry *d* is the compaction result of entry *a* and *b* (Figure 3) by replacing the last bit with a *don't care* (*). It shares the same port number *A* with entry *c*. When comparing the addresses in entry *d* and *c*, we take into account the *don't care* '*' in the address of entry *d* that could have a match with the 0

in the same bit position in entry *c*. If the addresses in these two entries have exactly one different bit with the exception of the *don't care* (*), they could potentially be compacted into a new entry. Then other entries with different port assignments are checked to make sure that there is no entry that falls within the new entry (in the example is labeled as entry *e)* address range. If an entry exists (in the example entry *f*) this compaction is not performed. Otherwise, entry *c* and *d* are replaced by *e*.

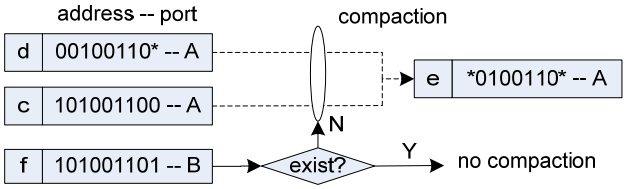

Fig. 4. An Example of Compaction Scheme C2.

4. PERFORMANCE EVALUATION

In this section, we implement our proposed compaction schemes on cache architecture, and compare their performance of two compacted IPv4 routing tables; and we use those IPv4 packet traces to evaluate the address lookup performance. We also introduce port error problems caused by caching and sampling technique to alleviate such problems. Finally we show the evaluation of our scheme scaled to IPv6 routing.

4.1 Compaction Ratio

In our evaluation, we first analyze the compaction performance of our schemes. We use compaction ratio (CR) as a measurement. CR is defined as the ratio of the number of the routing entries in the compacted and original tables. Table 2 shows the impact of these schemes on two different IPv4 routing tables RT1 and RT2. Obviously, compared with the original tables (labeled by "O") without any compaction, C1's CR is nearly 65%. The other one C2 has much lower CR, which are 29.45% and 33.64% respectively.

Table 2. The Two Compaction Schemes (CS) and Their Impact on Compaction Ratio (CR).

CS	RT1		RT2	
	No. of Entries	CR (%)	No. of Entries	CR (%)
O	90,000	100.00%	90,000	100.00%
C1	59,019	65.58%	60,185	66.87%
C2	26,505	29.45%	30,279	33.64%

As we mentioned earlier, scheme C1 is similar to the one proposed by Liu [8]. The performance reported for IPv4 is close to the one obtained here. The compaction ratio reported in [8] ranges from 52% to 57.3%.

4.2 Cache Hit Rate

The different compacted routing tables result in the variety of hit rate in the cache architecture. Hit rate is one of major measurements of performance for a cache memory. It also provides a good indication of the potential gain in performance. Because main memory access time is much longer than cache access time, many searches in memory will slow down the lookup process. However, with a higher hit rate, fewer routing table memory accesses are needed. The hit rate of cache with different compaction schemes is shown on Table 3.

Table 3. Hit Rate of Cache with Different Compaction Schemes (CS).

Trace	CS	512	1K	2K	4K
W	O	73.94%	83.98%	89.54%	92.71%
	C1	74.16%	84.22%	89.82%	93.08%
	C2	91.79%	96.99%	98.73%	99.20%
X	O	69.12%	81.44%	88.04%	91.89%
	C1	69.36%	81.73%	88.40%	92.34%
	C2	90.48%	96.68%	98.61%	99.13%
Y	O	74.76%	85.43%	91.30%	94.12%
	C1	75.02%	85.70%	91.57%	94.44%
	C2	92.62%	97.53%	98.94%	99.31%
Z	O	94.04%	94.89%	95.59%	95.82%
	C1	94.07%	94.97%	95.68%	96.00%
	C2	97.47%	98.67%	99.14%	99.31%

A minor improvement of the hit rate is obtained when scheme C1 is used, on the other hand, using the proposed scheme C2, the hit rate improvement ranges from 3.43% to 21.36%. All of the improvements depend on the trace files and cache size.

The following Figure 5 gives us another clear plot on the improvements. We observed the hit rates increase with the increasing cache size; the amount of the increment is diminished with the increasing cache size; and the amount is also depending on the different trace files.

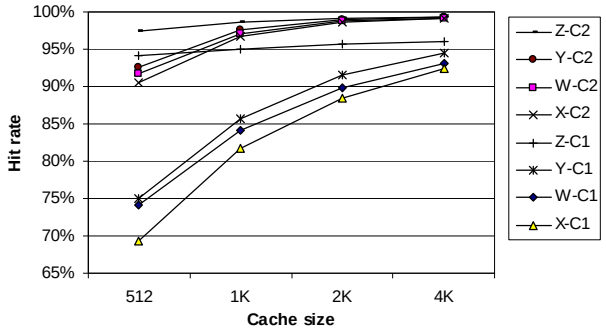

Fig. 5. Compaction Schemes' Hit Rate.

4.3 Port Error and Sampling Technique

A port error occurs when the port selected using the cache doesn't match the port that would be selected using the routing table. A port error is generated because the matching entry in the cache is not the longest prefix matching entry in the routing table. By implementing the compaction schemes, the number of port errors greatly

increases, since the time that the entry with a short prefix or with many *don't care* stays in cache could be extended. A sampling technique is one way to alleviate this problem [9]. Below, we introduce and evaluate this technique in our scheme.

It requires performing one search in the routing table every *M* lookups. *M* is the sampling rate. We called this scheme *M interval sampling*. When a sample is performed and if the port number found in cache is different from both the routing table, the matching entry in the routing table will be written into the cache to fix future port assignments.

In order to analyze the port errors, we choose port error ratio as the measurement. Port error ratio is the ratio of the number of port errors to the number of cache hits. Table 4 depicts the performance of the M interval sampling technique implemented on different compaction schemes. In the simulations M is made equal to three. We observed that the port error problem is lessened with sampling. The tables compacted by the C2 scheme benefit the most. For example trace (W), with a 512-entry cache, scheme C2 has a bit of high port error ratio 0.909% without any process, but the ratio can be reduced to a low level 0.318% by sampling.

Table 4(a). The Port Error Ratio without Sampling (SA)

Trace	CS	512	1K	2K	4K
W	O	0.000%	0.000%	0.001%	0.002%
	C1	0.001%	0.001%	0.003%	0.005%
	C2	0.909%	0.783%	0.754%	0.700%
X	O	0.002%	0.002%	0.006%	0.011%
	C1	0.003%	0.003%	0.009%	0.015%
	C2	1.126%	1.215%	0.895%	0.628%
Y	O	0.001%	0.002%	0.001%	0.004%
	C1	0.317%	0.003%	0.003%	0.006%
	C2	1.881%	1.882%	1.646%	1.506%
Z	O	0.001%	0.001%	0.002%	0.002%
	C1	0.001%	0.002%	0.002%	0.003%
	C2	0.863%	0.914%	0.916%	0.892%

Table 4(b). The Port Error Ratio with Sampling (SA)

Trace	CS	512	1K	2K	4K
W	O	0.000%	0.000%	0.001%	0.002%
	C1	0.001%	0.001%	0.002%	0.004%
	C2	0.318%	0.239%	0.172%	0.126%
X	O	0.002%	0.002%	0.004%	0.005%
	C1	0.002%	0.002%	0.005%	0.007%
	C2	0.465%	0.321%	0.206%	0.144%
Y	O	0.001%	0.001%	0.001%	0.002%
	C1	0.021%	0.002%	0.003%	0.003%
	C2	0.440%	0.255%	0.168%	0.117%
Z	O	0.001%	0.001%	0.001%	0.001%
	C1	0.001%	0.001%	0.002%	0.002%
	C2	0.092%	0.097%	0.100%	0.090%

4.4 Implementation and Evaluation of IPv6

IPv6 is the next generation protocol [15,16], which has an address length of 128 bits. It has been proposed as a long-term solution to solve the address limitations of current IPv4 by providing a much larger address space. In the future, IPv6 will gradually replace IPv4.

Because IPv6's long address format, it is required more memory to store the large routing table. The compaction is a good method to save memory usage and improve the lookup performance. In this part, we scale our scheme to IPv6 and evaluate the lookup performance.

For the simulation, we have six IPv6 traces which are downloaded from [13] and labeled from A to F. We obtain IPv6 tables, namely RT3 and RT4, both from the machine "n6tap.es.net" on the website of 6TAP router information http://www.6tap.net/ [17] and from the machine "route-server.he.net" on Hurricane Electric Internet Services http://lg.he.net/cgi-bin/index.cgi [18] via telnet. Since currently there are few users on IPv6, the sample IPv6 routing table size is not large. We have extended or combined tables to satisfy the unique destination addresses in the trace files.

Obviously, the compaction ratio of C1 is almost 50%, and that of C2 is reduced down to 31%. Given a 128-entry cache, the hit rate increment ranges from 0.62% to 1.22% by applying scheme C1; it is increased by scheme C2 from 4.54% to 8.00%. Similar to IPv4, the sampling technique is very effective to decrease port error in caching architecture.

Table 5. IPv6 Traces.

Trace files	Number of unique addresses
(A) 031214	3949
(B) 031215	3681
(C) 031216	3422
(D) 040128	3764
(E) 040129	3874
(F) 040130	3772

Table 6. Compaction Ratio (CR) of Routing Tables.

CS	IPv6 RT3		IPv6 RT4	
	No. of Entries	CR (%)	No. of Entries	CR (%)
O	3685	100.00	3974	100.00
C 1	2052	55.69	2325	58.51
C 2	1155	31.34	1444	36.34

Table 7. Hit Rate (%) of 128-entry Cache with Different Compaction Schemes (CS).

CS	Traces					
	A	B	C	D	E	F
O	81.50	84.53	85.26	88.49	88.96	89.85
C1	82.72	85.49	86.20	89.37	89.58	90.49
C2	89.49	92.14	92.56	93.19	93.77	94.39

Table 8. The Port Error Ratio (%) without (w/o) and with (w/) Sampling (SA).

CS	SA	Trace file					
		A	B	C	D	E	F
O	w/o	0.052	0.006	0.045	0.032	0.033	0.000
	w/	0.024	0.003	0.008	0.014	0.017	0.000
C1	w/o	0.479	0.034	0.076	0.037	0.102	0.005
	w/	0.047	0.025	0.028	0.018	0.038	0.001
C2	w/o	0.392	0.131	0.107	0.067	0.212	0.187
	w/	0.096	0.059	0.032	0.031	0.044	0.059

5. CONCLUSION

Routing table lookup is becoming a crucial operation as both Internet traffic table sizes increase. In order to speed up this process, we have proposed a novel compaction schemes for IP routing tables and evaluated their impact on a cache architecture. We have also presented a sampling technique to alleviate port error problems caused by the caching. The simulations using the routing information have shown that our proposed scheme improve cache hit rate and reduce port error control. The scheme can also be scaled to IPv6 easily in future. The features of our compaction schemes are presented as follows.

- *Compaction schemes based on TCAM*. The scheme C1 is similar to the IPv4 compaction scheme via Espresso algorithm. The C2 is an improvement over C1 with higher compaction.
- *High compaction performance*. The C1 reduced over 30% of the entries in the original tables, while the other C2 reduces the number of the entries almost 70% for IPv4 routing.
- *High cache hit rate.* The cache hit rate is improved by performing searches in compacted routing tables, especially for scheme C2. The hit rate improvements go from 3.43% to 21.36% by using C2 scheme.
- *Less port error ratio.* The sampling technique is an effective way to alleviate the port error problems. Given a sample trace (W) 060524, the port error ratio is decreased from 0.909% down to 0.318% using compaction scheme C2 on a 512-entry cache by sampling.
- *Scale to IPv6 routing.* Our compaction scheme can be implemented on IPv6 with 128 bit IP addresses. It also helps to achieve good lookup performance for future use.

The routing table compaction has been shown to improve the caching effectiveness. A good compaction scheme increases cache hit rate with a small number of routing entries. A scheme to tighten port errors needs be included as well. The high cache hit rate makes average memory access time shorter; this in turn speeds up the address lookup process. The small port error ratio is extremely beneficial to reduce the possibility of incorrect routing.

REFERENCES

[1] P. Gupta, S. Lin, and N. McKeown, Routing lookups in hardware at memory access speeds. *Proceedings of InfoCom'98*, San Francisco, April 1998, 1240-1247.

[2] H. Chao, Next generation routers. *Proceedings of The IEEE, Vol. 90, No. 9*, September 2002, 1518-1558.

[3] D. Feldmeier, Improving gateway performance with a routing-table cache. *Proceedings of InfoCom'88*, March 1988, 298-307.

[4] B. Talbot, T. Sherwood, and B. Lin, IP caching for terabit speed routers. *Proceedings of Globecom.1999*, 1565-1569.

[5] T. C. Chiueh and P. Pradhan, Cache memory design for network processors. *Proceedings of High-Performance Computer Architecture*, January 2000, 409-418.

[6] R. Guo and J. G. Delgado-Frias, A pipelined cache architecture for IPv6 lookups. *International Conference on Communication and Computer Networks (CCN 2004),* Cambridge, Mass., November, 2004.

[7] J. J. Ronney, J. G. Delgado-Frias, and D. H. Summerville, An associative ternary cache for IP routing. *IEE Proceedings Computers and Digital Techniques, vol.151, Issue 6,* November 2004, 409-416.

[8] H. Liu, Routing table compaction in ternary CAM, *IEEE Micro, vol. 22, no. 1,* January-February, 2002, 58-64.

[9] J. J. Rooney, An Associative Ternary Cache for IP Routing. PhD Dissertation, State University of New York at Binghamton, NY 2002.

[10] J. L. Hennessy and D. A. Patterson, *Computer architecture: a quantitative approach,* 3rd edition, (San Francisco: Morgan Kaufmann Publishers Inc. 2003).

[11] J. G. Delgado-Frias, J. Nyathi and S. Tatapudi, Decoupled Dynamic Ternary Content Addressable Memories, *IEEE Transactions on Circuits and Systems, I: Fundamental Theory and Applications,* vol. 52, no. 10, October 2005, 2139-2147.

[12] J. G. Delgado-Frias and J. Nyathi, A High-Performance Encoder with Priority Lookahead, *IEEE Transactions on Circuits and Systems, I: Fundamental Theory and Applications,* vol. 47, no. 9, September 2000, 1390-1393.

[13] The Measurement and Analysis on the WIDE Internet (MAWI) Working Group. http://tracer.csl.sony.co.jp/mawi/

[14] University of Oregon Route Views Archive Project. http://archive.routeviews.org/

[15] C. Huitema, *IPv6: the New Internet Protocol,* 2nd edition, Prentice Hall, 1998.

[16] RFC 2373, "IP Version 6 Addressing Architecture," July 1998. (http://rfc.net/rfc2373.html).

[17] 6TAP router information. http://www.6tap.net/

[18] Hurricane Electric Internet Services. http://lg.he.net/cgi-bin/index.cgi.