AN ASSOCIATIVE TERNARY CACHE FOR IP ROUTING

James J. Rooney[1] José G. Delgado-Frias[2] Douglas H. Summerville[1]

[1]Dept. of Electrical and Computer Engineering.
State University of New York
Binghamton, NY 13903-6000

[2]School of Electrical Engr. and Computer Sc.
Washington State University
Pullman, WA 99164-2752

Abstract

In this paper we present a study of an associative cache for Internet IP routing. An output port assignment requires one cache memory access when the assignment is found in cache. The cache array is divided into sets that are of variable size; all entries within a set have the same prefix size. Our study shows that an associative ternary cache provides an output port at the speed of one memory access with a very high hit rate. For an 8K-entry cache the hit rate ranges from 97.62% to 99.67% on traces of 0.2 to 3.5 million addresses. A port error occurs when the port selected by the cache differs from the port that would have been selected from the routing table. The worst port error rate is 0.52%, and is reduced to 0.05% with the use of a sampling technique.

1. Introduction

The continuous growth in Internet related technologies and applications have imposed higher demands on network routers [7]. This in turn has lead to the emergence of network processors that are designed to execute network protocols at different computer network layers. These layers include: the link switches –layer 2, network routers –layer 3, and transport gateways –layer 4 [1]. Some of the specific tasks include: *Routing* that determines the output port to forward packets to, *Forwarding* that sends packet according to a flow control strategy, *Scheduling* which schedules packets according to their priorities to support Quality of Service –QoS, *Filtering* that monitors packets to support privacy –firewall, and *message manipulation* that supports application specific processing and time-to-live counter.

With the increasing speed of communication channels between computer and other devices in a network, it is important that the routing algorithm execution time be extremely short. This time has a great influence on how fast a packet advances through the network. A packet cannot be transferred until the next hop is selected. Routing algorithm execution time must be reduced to decrease packet latency.

The basic transfer unit on the Internet is the Internet datagram, which is divided into header and data areas. The datagram header contains the source and destination addresses. Each host on an Internet is assigned a unique 32-bit Internet address that is used in all communication with that host. IP Routing does not alter the original datagram. The source and destination addresses remain unaltered. The address consists of two parts, Internet part and the local part. The Internet part identifies a site, and the local part identifies a physical network and a host at that site.

Each IP routing table entry includes the following fields: a network mask, a destination network address and a next-hop address. In IPv4, the network mask and destination address fields are 32 bits long. The network mask field is composed of contiguous binary 1s to designate the network field followed by binary 0s to fill out the 32-bit field. IP addresses and masks are usually written as four decimal integers separated by decimal points, where each integer gives the value of one octet of the IP address, this is called dotted decimal notation. Given an incoming packet's destination host address, the network mask field of an IP routing table entry is used to extract the destination network address, which is then compared to the entry's destination network address field. If they match, this routing table entry is considered a potential candidate. This process is repeated until all possible addresses in the routing table have been considered. When there are multiple matches the match with the longest network address is chosen over the others; this is called longest prefix match [2]. The packet is routed via the output port specified in this entry to the corresponding next-hop router. If none of the table entries match the incoming IP packet's destination host address, the packet is forwarded to the default router. To reduce time in executing best matching prefix algorithms the search is done using a tree-like structure. Accessing this structure requires a number of memory references; this number depends on how deep the tree is. There is a need to provide new techniques for fast IP routing.

This paper has been organized as follows. In Section 2, the associative cache routing scheme is described. The scheme's performance is presented in Section 3. A comparison of our results to related work is presented in section 4. Some concluding remarks are provided in Section 5.

2. Associative Cache Scheme

Our approach is based on an associative IP routing cache using a high performance bit-pattern associative array system. The associative router implements a Classless Inter-Domain Routing (CIDR) which the most common and flexible form of Internet

routing [8]. To our knowledge, this is the first attempt to use an associative cache approach in this context. Other approaches have used other types of memories [1,6,7]. Since the data streams presented to Internet Routers exhibit different characteristics than general purpose CPUs, our cache scheme is considerably different than a conventional CPU cache. We have greatly enhanced the basic Bit Pattern Associative Array [9] approach to be used as the cache for Internet Routing Tables. The CAM based approach features a parallel search of all entries, thereby performing a cache lookup in one cache memory cycle. Our approach can also execute routing algorithms for regular topologies.

2.1 Bit-Pattern Cache for IP Routing

Figure 1 shows the proposed bit-pattern scheme for cache Internet routing lookups. The cache array is divided into sets that are of variable size. The size of each set is made proportional to the equivalent set size in the routing table, which are heavily skewed to entries in sets 8 to 16. All entries within a set have the same prefix size. Since we are dealing with a 32-bit IP address, there are 32 possible prefix values.

Each cache entry contains a 32-bit network address and a designated port. Each cache entry corresponds to and is equivalent to one routing table entry. The network address entry is generated by masking the standard route table network address with the network mask, and replacing all low order elements below the prefix-suffix boundary with don't care elements. For example:

9.20/255.255.128 IBM (port designation)

In the bit-pattern associative scheme, this entry would be converted to a single 32-bit address with the designated port; by masking, the address and replacing all elements below the prefix-suffix boundary with don't care (X) elements.

00001001 00010100 0XXXXXXX XXXXXXXX IBM (port)

The Bit-pattern Associative Cache scheme is described below.

1) *Destination Address.* A 32-bit destination serves as input to the cache array.
2) *Associative Pattern Array.* As shown in Figure 1 the addresses in the bit-pattern array are grouped by sets, which indicate the prefix value of the network address. The upper set in the cache is set [0] with a 32-bit prefix and proceeding down to bottom of the array, we find set [31] with a one bit-prefix. An implication of this is that the array is ordered, starting at the top by the longest network addresses. When searching for a match the destination address is compared to all entries of the array in parallel. The match closest to the top of the array (set [0]) has the longest network address.
3) *Priority.* For a basic IPv4 router, since the top-most match would have the longest network address, the port from that entry would be selected. Additional services can be performed by adding logic in this function. For instance, filtering which is the denial of certain networks from designated sources, and priority based routing could be accomplished in this selection function. If there were alternative ports to select from this would also be handled in this priority function.
4) *Output interface assignment.* Selecting the output port associated with the selection condition makes the output port assignment.

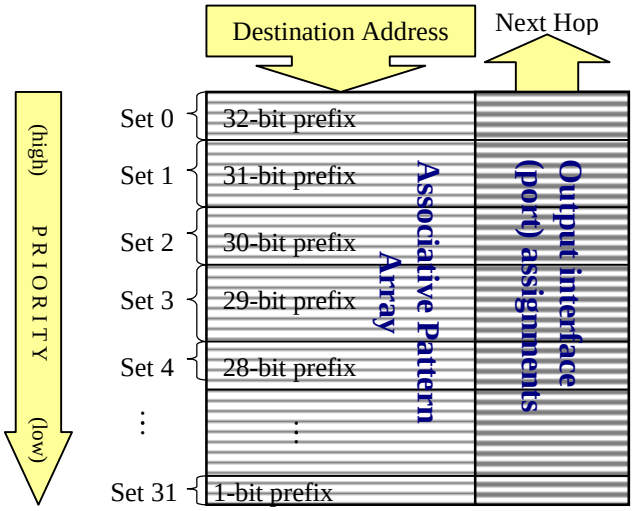

Figure 1. Bit-pattern associative scheme.

3. Associative Ternary Cache Performance

We have studied the performance of our associative router using six IP header trace files, which have been obtained from the web site http://pma.nlanr.net/PMA/ [5]. Files in these directories are public packet header traces of Internet traffic, after being sanitized in the IP address fields. The encoding of the IP addresses fields is done in a manner to retain all statistical features of the original fields. To make these files useful, a set of IP addresses were obtained from the routing table, and used as replacements for the sanitized address fields. Sanitized addresses that were repeated use repeat addresses from the routing table. This assured that the temporal locality of the IP Header files was maintained. Six IP packet traces, containing 12,450,768 header files were downloaded and used to predict the cache performance. Our

routing tables were obtained from the web site: http://www.merit.edu/ipma [4]. In our simulator we implemented a Least Recently Used (LRU) replacement policy for the proposed associative cache; however, other policies have been implemented for comparison purposes.

A summary of the IP Address Trace Files (from http://pma.nlanr.net/PMA/) and their features is shown in Table 1 below. The first column identifies the file in the pma.nlanr.net web site directory. The second and third columns indicate the number of IP Headers and unique IP destination addresses, respectively. The ratio of number of packets to number of unique addresses could be used as an indicator of temporal locality in a trace file. This ratio is shown in the last column.

Table 1. IP Address Trace Files

IP Header File ID	Number of packets/ destination addresses	Number of unique IP addresses	ratio
20011206/ADV-1007598319-1	203,352	1,465	139
20011001/BWY-1001908207-1	1,774,231	5,690	312
20011207/TAU-1007728734	1,816,318	9,601	189
20011001/BWY-1001895342-1	1,887,643	17,119	110
20011001/IND-1001895343-1	3,266,998	32,014	102
20011206/APN-1007673284	3,502,216	15,264	229

3.1 Cache hit rate

One of the prime measures of performance on caches is hit rate. This refers to the percent of cache accesses that successfully retrieve the required data. In our case the data would be the output port. The principle of locality helps to achieve a high cache rate in microprocessors. This principle (temporal locality) is present in most routers; i.e. packets with the same destinations tend to cluster in time.

We have run a number of simulations with traces that contain from 0.2 to 3.5 million routing decisions. Table 2 shows cache hit rates (CHR) for 1K, 2K, 4K, and 8K-entry caches. Cache hit rates range from 70.84% to 99.67%. The trace of 3.26 million packets exhibits less locality than any other; this in turn leads to lower hit rate. It should be noticed that the hit rate for this trace greatly improves as the cache size increases. For all the other traces, the hit rate is very high and improves with cache size.

Table 2. Cache hit rate (1K, 2K, 4K and 8K entries)

Trace size	8K	4K	2K	1K
203,352	99.20%	99.20%	99.20%	98.91%
1,774,231	99.67%	99.65%	99.44%	92.50%
1,816,318	99.49%	99.35%	98.76%	94.96%
1,887,643	99.10%	98.99%	98.34%	92.75%
3,266,998	97.62%	93.21%	82.15%	70.84%
3,502,216	99.54%	99.26%	97.22%	81.83%

Figure 2 shows cache hit rate and its relation to ratio of the number of packets and unique IP addresses. It should be noted that benchmarks with lower ratio tend to have lower cache rate, for cache sizes greater than 1K. This is due to limited temporal locality.

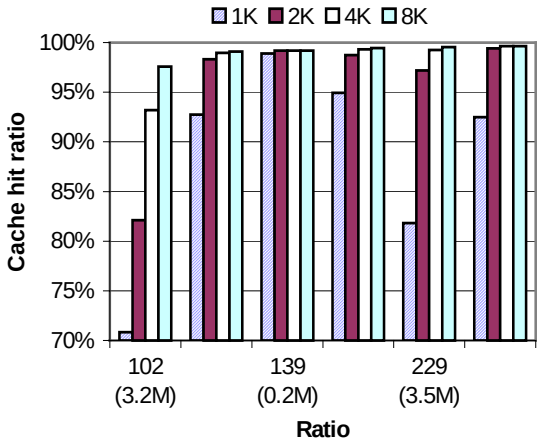

Figure 2. Plot of cache hit rate and number of packets/unique IP addresses ratio

3.2 Port error rate

A port error occurs when the port selected by the cache differs from the port that would have been selected from the routing table. This in turn leads to an inappropriate port assignment. Table 3 presents the simulation results of the traces' port error rates.

Table 3. Port error rate (1K, 2K, 4K and 8K-entry)

Trace size	8K	4K	2K	1K
203,352	0.012%	0.012%	0.012%	0.023%
1,774,231	0.026%	0.026%	0.029%	0.062%
1,816,318	0.048%	0.055%	0.031%	0.316%
1,887,643	0.084%	0.071%	0.072%	0.193%
3,266,998	0.52%	0.482%	0.327%	0.202%
3,502,216	0.097%	0.086%	0.090%	0.114%

There is not a specific pattern to this measurement other than the error rate is decreased for cache sizes above 1K except for the 3.2M traces, which has limited locality.

The major causes for port misses are:

1. The longest prefix network address match, for the destination address being searched is in the routing table but does not happen to be in the cache at this time. The cache finds another match, and in some cases the port assignment is not the same as would be found in the routing table.
2. The second cause of port misses is due to a cache coherency problem. In this case, the problem is that changes to the routing table are made frequently, making the port assignment different in the routing table and cache.

Cache coherency can be assured by updating the cache with the new information that is received by the routing table or by flushing the cache every time the routing table is updated. Another approach that would reduce port misses from either of the two causes would be to use sampling, where samples of cache assignments are sent to the routing table to confirm the proper assignments.

3.3 Port error rate improvements with sampling

Sampling requires a search of the routing table after a cache hit and a comparison of the ports selected by both searches. When a port selected by the cache is different than that selected by the router, the cache entry is removed and the routing table entry is written to the cache. This activity does not interfere with the normal operation of the cache, since the port selection on all cache hits, even if they are in error, are returned to the cache controller. The search of the routing table after a cache hit is performed at a sampling rate designed into the cache simulator software. Table 4 presents the simulation results of the port error rates at a sampling rate of 33%. At this rate the routing table is searched every third cache hit and the port selections compared. The average reduction of port misses in these simulations is close to 9 when measured against all 6 traces and the 3 cache sizes. It should be noted that the patterns remain the same as Table 3, namely that port misses decrease with cache size greater than 1K with the exception of the one trace that shows poor locality.

Table 4. Port error rate with sampling.

Trace size	8K	4K	2K	1K
203,352	0.003%	0.003%	0.003%	0.003%
1,774,231	0.002%	0.002%	0.002%	0.017%
1,816,318	0.006%	0.005%	0.005%	0.011%
1,887,643	0.011%	0.009%	0.007%	0.019%
3,266,998	0.050%	0.066%	0.081%	0.052%
3,502,216	0.008%	0.006%	0.006%	0.044%

Figure 3 shows improvements obtained by using our sampling approach. This improvement is obtained by having port error rate with no sampling divide by port error rate with sampling. It should be pointed out that all the IP traces have an improvement that ranges from 2.59 (3.5 IP trace, 1K cache size) to 28.7 (1.81 IP trace, 1K cache size).

Most simulations with sampling duplicated the cache-hit ratio without sampling within 0.01%. The largest changes occurred with the 3.27M trace file where cache-hit ratio changed from 93.21% to 93.05% for the 4K cache.

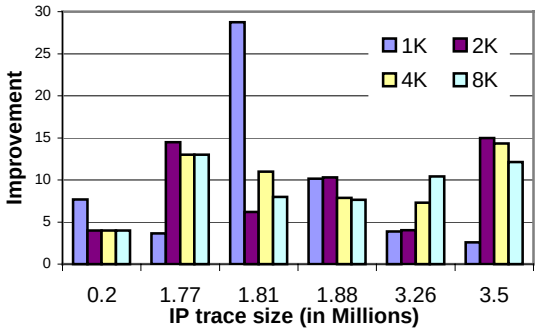

Figure 3. Port error rate improvement using sampling.

3.4 Cache writes

Another issue to consider in our scheme would the number of times the cache needs be written. Table 5 shows the number of writes that are required to handle the traces and the cache sizes. In Figure 4, we show the relative number of writes, this is relative to 1K-entry cache. For large traces the number of writes is greatly reduced with the increase of cache size (an order of magnitude for some traces).

Table 5. Number of cache writes.

	1K	2K	4K	8K
203,352	1792	1624	1624	1624
1,774,231	87001	9994	6205	5809
1,816,318	73647	22600	11866	9378
1,887,643	130346	26979	19160	17248
3,266,998	937696	585927	227005	81040
3,502,216	613931	97861	26108	16198

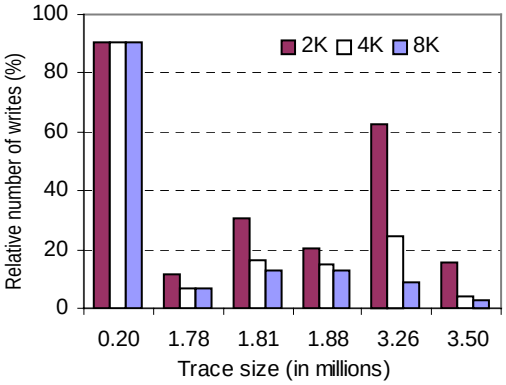

Figure 4. Relative number of cache writes.

3.5 Replacement policies

When a set is full and a new entry needs to be inserted a replacement has to be made. We have investigated the impact of the replacement policies on the performance of our approach. Three primary cache replacement strategies are employed to select the cache entry to replace. These are: Random—candidate entries are randomly selected, Least-Recently Used (LRU)—The entry that is replaced is the one that has been unused for the longest time, and

First In First Out (FIFO)—The first cache entry written to a set will be the first replaced. Table 6 shows the result of 1.78M benchmark.

Table 6. Performance of replacement policies.

Measurements	LRU	Random	FIFO
Cache Hits	1689616	1640595	1640325
Port Hits	1682882	1635470	1635693
Port Misses	6734	5125	4632
Cache Writes	84615	133636	133906
Cache Hit Ratio	95.23%	92.47%	92.45%
Port Error Rate	0.40%	0.31%	0.28%

In all the benchmarks LRU performs better than the other two replacement policies. In general, LRU gets a higher cache hit ratio of approximately 2%. Sampling to reduce port error rates was not included in these simulations. In the other benchmarks a similar performance is obtained.

4. Comparison with other schemes

To our knowledge this is the first study to perform a systematic analysis of the design of an Associative Ternary Cache for IP routers. Related work has been reported by: C Partridge of BBN [6] and T. Chiueh and P. Pradhan of SUNY-Stony Brook [1]. In this section, we compare overall cache-hit/cache-miss results for our designs with these studies.

4.1 Comparisons with BBN study

This study was done on a backbone router using a 5-minute trace from FIX WEST. The total number of Internet packets was 4.97 million, which used 31.9 thousand total destination network address from the router [6]. We have compared it to two of our traces, which have both smaller and larger address ratios. This comparison results are shown in Table 7; blank spaces ----- indicate that the data is not available.

The cache-hit ratios from our scheme are higher, even for the trace (3.27M) that has lower Ratio, than those reported by BBN. C. Partridge paper [6] does not report port errors caused by not achieving the longest prefix network address match. Therefore a comparison of port error rate cannot be made with our approach. Our scheme uses LRU cache replacement policy. In BBN experiments, the replacement policy was not reported.

The cache-hit ratios from our scheme are higher, even for the trace (3.27M) that has lower Ratio, than those reported by BBN. C. Partridge paper [6] does not report port errors caused by not achieving the longest prefix network address match. Therefore a comparison of port error rate cannot be made with our approach. Our scheme uses LRU cache replacement policy. In BBN experiments, the replacement policy was not reported.

Table 7. Comparisons of Cache-Hit Ratios

Cache Study	BBN [6]	Ternary Cache	Ternary Cache
Trace Size	4.97M	3.27M	3.50M
Address Ratio*	156	102	229
Cache Size	Cache Hit Ratio		
1K	60.9%	70.8%	81.3%
2K	77.4%	82.2%	97.2%
4K	-------	93.2%	99.3%
5K	93.3%	-------	-------
8K	-------	97.6%	99.5%
10K	97.4%	-------	-------

**Address Ratio* is the number of trace packets divided by the number of unique destination network addresses. A high number may indicate a high temporal locality.

4.2 Comparisons with Stony Brook study.

The paper by T. Chiueh and P. Pradhan [1] reports the results of a research effort on cache memory designs for Internet processors. This research included gathering of a large Internet trace file of 184.4 million packets. The two approaches are very different; ours uses a variable set ternary cache for Internet routers, while Chiueh-Pradhan's employs a more conventional associative cache with set sizes of 1, 2, and 4 for Internet processors. We have compared to what seemed to be the best results of their designs, which were cache sizes of 4K and 8K entries with block size of one, and is called the Host Address Range Cache (HARC). We have shown the cache-miss ratio, which is defined as 100% minus the cache-hit ratio in Table 8.

Table 8: Cache-Miss Ratio comparisons

		Ternary Cache	Stony Brook
	Trace Size →	*12.5M*	*184.4M*
Cache Size	Associativity	Cache-Miss Ratio	
4K	1	------	7.54%
	2	------	4.58%
	4	------	3.64%
	Variable	1.72%	-------
8K	1	------	4.48%
	2	------	2.20%
	4	------	1.56%
	Variable	0.80%	------

The data shown under the Ternary Cache Study is the consolidation (12.5 million packets) of all six IP traces that were reported in this paper. The cache-miss ratio is the average of the six traces. This consolidation captures the effects of "flushing" the cache at the start of each of the six trace simulations.

Chiueh and Pradhan paper [1] does not report port errors caused by not achieving the longest prefix network address match. Therefore a comparison of port error rate cannot be made with our approach. The paper does state that a port assignment would be made in one CPU cycle (which includes one cache

memory cycle) if the assignment were found in cache. This compares with our port assignment in just one cache memory cycle when the assignment is in cache. From Table 8, it can be observed that our scheme leads to cache ratio numbers of about half of the best performance on the HARC scheme. These comparisons suggest that our ternary cache scheme is superior to a conventional approach.

5. Concluding Remarks

In this paper we have presented a study of a novel associative cache scheme for Internet IP routing. This is the first study that deals with associative caches for this application. In our scheme an output port assignment requires only one cache memory access when the assignment is found in cache. When it is not found in cache, the routing tables must be accessed.

Our study shows that an associative cache provides an output port at the speed of one memory access with a very high hit rate. A list of the main features of the proposed associative ternary cache is provided below.

- *High cache hit rates.* A high cache hit rate is achieved with relatively small cache sizes; this is due to the high temporal locality exhibited by these IP trace files. The 4K and 8K-entry cache hit rates ranges from 93.21% to 99.65% and from 97.62% to 99.67%, respectively.
- *Reduction of port misses.* A sampling technique was introduced to reduce port misses. It has been shown that the worst-case port error rate, caused by not having the longest prefix match in cache, could be reduced from 0.52% to just 0.08%. The sampling technique also reduces port misses caused by cache incoherency due to routing table updates. Sampling is an alternative to flushing the cache to assure cache coherency every time the routing table is updated. It was also shown that the number of cache writes is significantly reduced as the cache size was increased from 1K to 8K entries; often as much as an order in magnitude.
- *LRU replacement policy.* The least recently used (LRU) replacement policy has shown to outperform the other two policies (random and FIFO) by as much as 2%.
- *Comparison with other schemes.* Our scheme when compared with two other approaches [1, 6] shows a better cache hit ratio, which is the primary measure of performance for cache schemes. There is no report on the port miss ratio for the other two approaches.

Our scheme imposes no additional delays to IP routing schemes since the same address that is sent to the routing tables is used for the cache. The cache entries are inserted as the routing table provides the port assignment to a given address. The sampling technique updates the cache in much the same way, inserting the correct routing table entry into cache after the cache-hit selection has been forwarded. Thus, the cache scheme is completely transparent to the user and improves the performance of the router node.

The proposed associative cache IP router scheme introduces a novel way to deal with sets, which are traditionally of a fixed size. In our scheme we allow some sets to be much larger than others to accommodate the statistics of routing table prefix distributions.

Acknowledgements

This work was supported in part by the National Science Foundation under contract CCR9900643.

References

[1] T. Chiueh and P. Pradhan, “Cache Memory Design for Internet Processors,” *6th Symposium on High Performance Computer Architecture (HPCA-6)*, Toulouse, France, January 2000.

[2] W. Doeringer, G. Karjoth, M.Nassehi, “Routing on Longest-Matching Prefixes,” *IEEE/ACM Transactions on Networking*, Vol. 4, No. 1, February 1996.

[3] P. Gupta, S. Lin, and N. McKeown, “Routing Lookups in Hardware at Memory Access Speeds,” *IEEE InfoCom*, San Francisco, April 1998.

[4] Merit Networks Inc., *Internet Performance Measurement and Analysis (IPMA) project*, a joint effort of the University of Michigan Department of Electrical Engineering and Computer Science and Merit Network. The web site is: http://www.merit.edu/ipma.

[5] National Laboratory for Applied Network Research, *Passive Measurement and Analysis Project of the National Laboratory for Applied Network Research at San Diego Super Computer Center*. The web site is: http://pma.nlanr.net/PMA.

[6] C. Partridge, “Locality and Route Caches”, Position Statement for the *1996 NSF Workshop on Internet Statistics Measurement and Analysis*. Available at: http://www.caida.org/outreach/isma

[7] C.Partridge et. al., “A 50-Gb/s IP Router”, IEEE/ACM Transactions on Networking, Vol.5, No.3, June, 1998.

[8] Y. Rekhter and T. Li, “An Architecture for IP Address Allocation with CIDR,” *RFC 1528*, September 1993.

[9] D. H. Summerville, J. G. Delgado-Frias, and S. Vassiliadis, “A Flexible Bit-Associative Router for Interconnection Networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 7, no. 5, pp. 477-485, May 1996.