International Journal of Future Generation Communication and Networking
Vol. 13, No. 4, (2020), pp. 4007–4014

Design and Performance evaluation of Cache based PBFP Trie – Proposing a state of art approach for efficient IP Lookup

[*]J. Sarada Lakshmi[1], Prof. KudaNageswara Rao[2]
[1]Research Scholar, Dept. of Computer Science and Systems Engineering
[2]Professor, Dept. of Computer Science and Systems Engineering
Andhra University College of Engg. (A), Andhra University, Visakhapatnam, Andhra Pradesh, India.
jammi_sarada@yahoo.com[1], knraoauce@gmail.com[2]

Abstract:

Internet routing is a major concern nowadays since the global routing table is increasing rapidly. One major facet in the overall performance of a router is the Lookup speed. In this paper, an innovative approach is proposed which uses cache routing-table which contains recently used IP addresses and their forwarding information to speed up the IP address lookup operation in the routers. The secondary storage structure to which the cache is augmented is Partitioned Bloom Filtered Priority Trie[20]. The performance is evaluated in terms of Lookup speed and Hit ratio which gave significant results. A synthetic trace file is generated using which the proposed scheme is tested and our approach seems to be efficient.

Keywords: Prefix Caching, PBFP Trie, IP Address lookup, Cached PBFP Trie

1. Introduction:

The Internet is ubiquitous and registering an exponential growth in size. The number of domains, sub-networks and users connected to the Internet are mounting. The Internet network constituents include routers which forward packets to the respective destinations, and physical links that transmit packets from one to another. When a packet arrives at a router, itmust decide on how to forward the packet which is inherentin determining the packet's next-hop. The ability of a router in packet forwarding process lies in finding the longest prefix in the routing table that provides the best match to the destination IP address of the data packet within negligible time. The routing table is structured in terms of destination address, prefix and next hop, where the length of a prefix is limited by the length of the destination IP address (31 for the IPv4 addresses and 127 for IPv6). Through the years, the major bottleneck in the performance aspect of IP routers is the time to find a next-hop for an incoming IP packet in the routing table. PBFP Trie had exhibited a noteworthy performance and our idea is to augment the cache to still improve. In Section 2 we discussed regarding various ideas proposed by other researchers. In the later section we proposed our architecture. Section 4 details the experimentation and performance comparison.Finally we concluded our approach in the last section.

ISSN: 2233-7857 IJFGCN

2. Relatedwork:

Rapid growth of the global forwarding information base raises serious concerns. In addressing such problem, Yaoqing Liuet.al. in [3] proposed an effective FIB scheme which achieved high hit ratio. Also their scheme prevented the cache hiding problem. The cache misses, cache replacement and routing updates were also handled efficiently.

Introducing different types of schemes which are less complex is also very much required in IP Lookup design. R.C. Chang et.al. proposed such scheme in [4] by using careful memory management design. As a part of memory management, instead of storing the range which is done usually in conventional approaches, IP routes are stored only once. A skip function was also introduced due to which the memory requirement is reduced.

Designing schemes which increase the look up speed is a prime task of IP Lookup researchers. HouassiHichemet.al. proposed a best lookup operation using Cache routing table approach in [5]. They structured the IP routing table as main routing table and cache table. They compared the performance of their proposed scheme with that of the binary trie based algorithm, which offered better performance.

A two-level Bloom filter scheme was proposed in [6] in which theoriginalprefixcouldbecachedwithoutexpansion. Foreachprefix,to test whether any matchable descendant exists, a Bloom filterisused. The scheme attained a very low cache miss ratio without increasing the number of prefixes.

In [7] GirijaNarlikaret.al. presented an investigative model which predicts the performance of software based IP lookups accurately using hierarchical data structures.This model is useful in selecting the suitable data structure based on hit ratio distribution to different networks in the routing table.The size and latency of the L2 cache and the processor speed up to some degree were found to be critical in determining the lookup performance. Also they found that there was no any significant improvement in the performance with a simple increase in the L1 cache or space requirement of the underlying data structure.

Hardware solutions were also part of Lookup problem. In [8]Guangdeng Liao et.al. proposed a new IP cache architecture which was designed along two axes – cache indexing and replacement policies. They employed 2-Universal Hashing and coupled with cache indexing scheme. Substantial throughput performance was observed.

Subsequently researchers addressed various aspects which include reducing conflict misses and cache consistency during frequent route updates[9]. Speeding up access to DRAM was addressed in [10]. Huan Liu et.al. besides evaluating the effectiveness of caching on routing prefix, proposed an On-chip routing prefix cache design for the network processor. An active prefix caching scheme was proposed in [11] wherein incremental updates are supported without prefix expansion. However a prefix caching scheme based on prefixes covering relationship was proposed in [12], while prefix caching for Named Data Networking was

ISN: 2233-7857 IJFGCN

addressed in [13]. A routing table lookup algorithm used in cluster based parallel IP router was discussed in [14].

3. Proposed Design:

In this work, we proposed an approach of augmenting the PBFP Trie with Cache so as to boost up the lookup performance. In the conventional IP Lookup, upon listening to the incoming IP Address, the prefix shall be searched and if it exists in the storage, the next hop address is obtained. Subsequently the incoming packets are routed through that port. In [20] PBFP Trie was proposed which exhibited amazing performance in terms of lookup, storage and other operations. Nevertheless the lookup speed could still be enhanced by supplementing the existing structure with Cache. Cache Memory stands for its high speed performance and less memory size. In our work we are intended to propose a simple architected efficient algorithm which improved the lookup performance of PBFP Trie. The architecture of our approach is as in figure1.

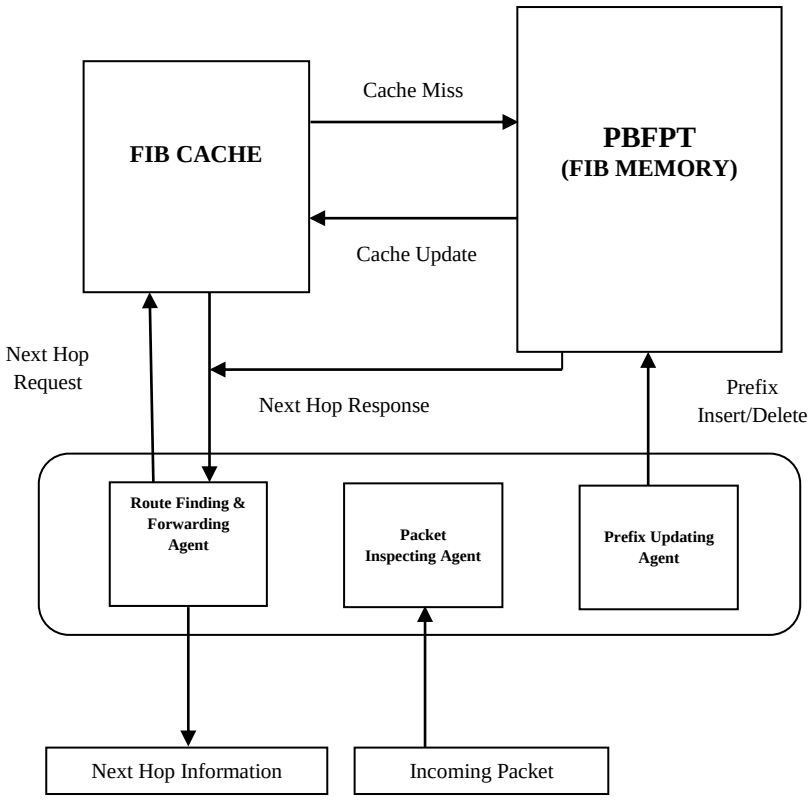


Fig 1 : Proposed Architecture

The cache is implemented as an array of locations in which the prefixes with theirrespective next hop addresses are stored. The architecture has three major components, one the memory organized as PBFPT (FIBM), the other is Cache (FIBC) and the third is the Routing Agent (RA). The incoming packet will be inspected in which the prefix will be forwarded for getting the next hop information by RA. If the prefix information is available then the next

ISN: 2233-7857 IJFGCN

hop information is obtained accordingly the packet is routed. The associated FIBC operations include Cache Initialization, Handling Cache Misses and Cache Replacement.

Cache Initialization: In handling the initial traffic Cache initialization plays a vital role. As the focus is on lookup speed and during the lookup process the cache is searched first for the prefix availability, it will always result in better performance in case the frequently accessed prefixes are stored in it. In other words if the Cache is initialized with those prefixes the performance in terms of lookup speed will be improved significantly.Mostly if the cache initialization is done with higher length prefixes most probably like root node and so on, the hit ratio and lookup speed pops up.

Cache Hits and Misses: Cache Hit ratio is a crucial metric which has an exponential effect on the lookup speed. Higher hit ratios register notable lookup speeds. Prefix availability in FIBC is termed as Cache Hit while the unavailability is said to be a Cache Miss. In case of Cache Miss, FIBM gives the Next hop information and updates the cache.

Cache Replacement: When a new prefix is to be accommodated in the cache while it is full, any of other prefixes are to be replaced according to certain policy. We used LRU (Least Recently Used) algorithm for prefix replacement since it was the algorithm whose performance is optimal.

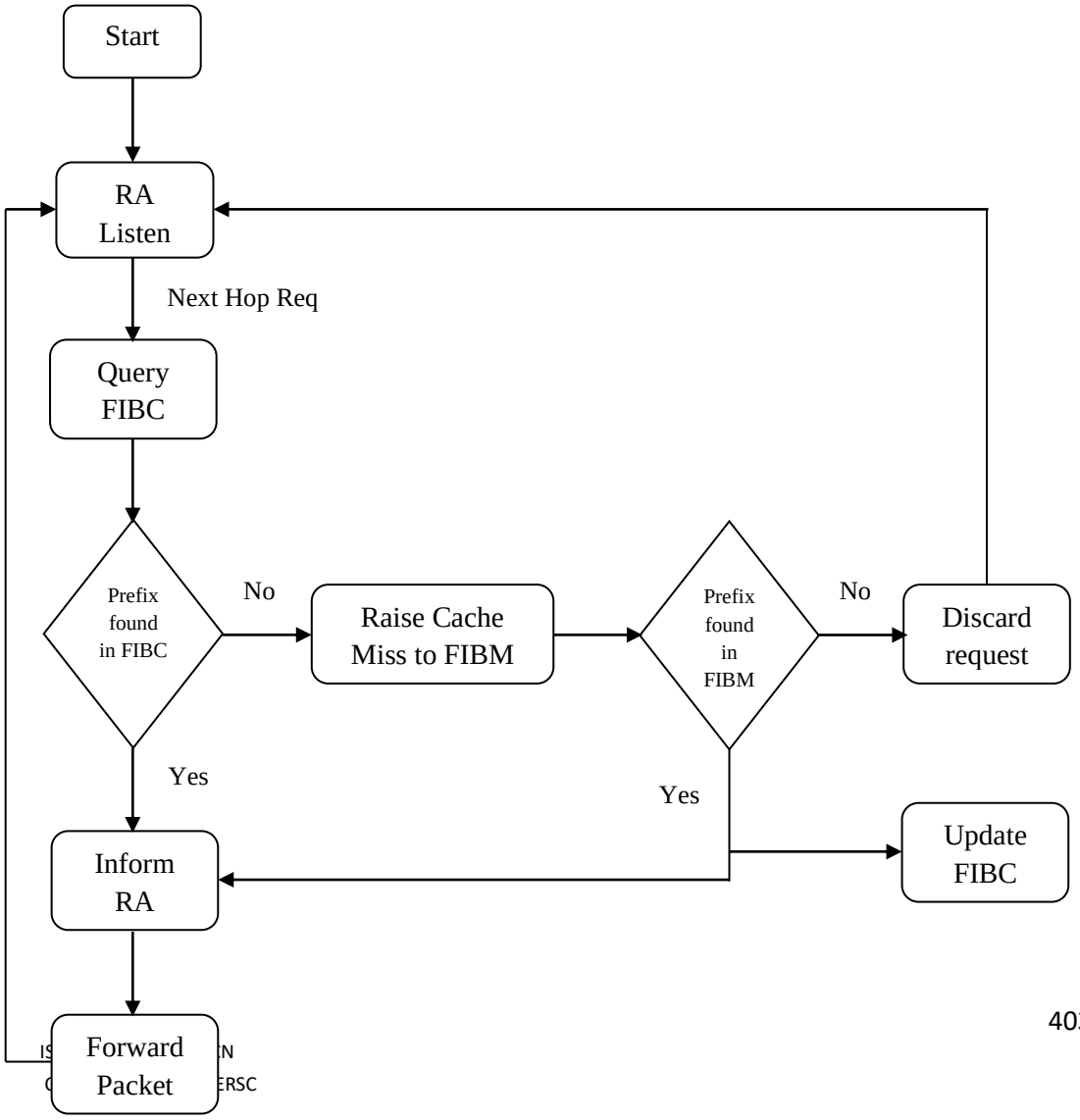


Fig 2. Flow of the lookup procedure

Lookup procedure:

The flow of prefix lookup procedure is as observed in the figure 2. Initially the Routing Agent (RA) will be listening to the incoming packets. When a packet arrives the prefix is extracted and queries the Cache for its intended next hop. If it is found in cache then RA receives the information and routes the packet. In case the prefix is not found in the cache then a cache miss is raised by informing to the FIBM. The prefix is then searched. If prefix is found, RA will be informed and also the cache gets updated with this information.

4. Experimentation and Performance evaluation:

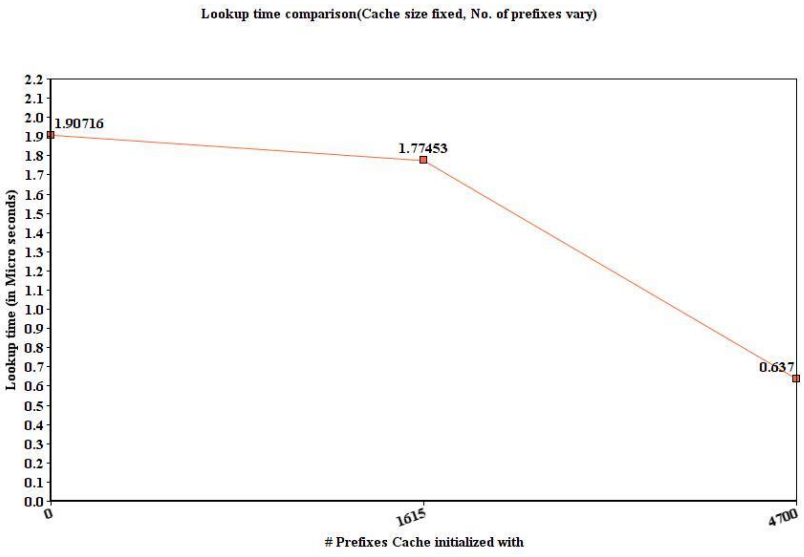
In our scheme we considered the PBFP Trie and augmented it with Cache(FIBC). A Synthetic Trace of different sizes was generated with the prefixes extracted from [19] and the performance is evaluated in terms of Lookup speed and Hit ratio.

4.1 The core operation in the total IP Lookup procedure is the time for searching for a prefix which is termed as Lookup time. Firstly we evaluated it by varying the number of prefixes besides the cache is initialized to 16384 locations. The performance results are as per the table 1 and the graphical representation could be observed in figure 3.

Table 1: Lookup times for various cache initializations

No. of prefixes Cache is initialized with	Cache Size (No. of Locations)	Lookup time (Micro seconds)
0	16384	1.90716
1615	16384	1.77453
4700	16384	0.637

As the cache is fast accessible memory, if a prefix exists in it, the lookup time for that particular prefix would be of negligible time. The same performance is observed, the more number of prefixes the cache is



ISN: 2233-7857 IJFGCN

initialized with the lookup time gets reduced. Hence Cache initialization increases the lookup speed.

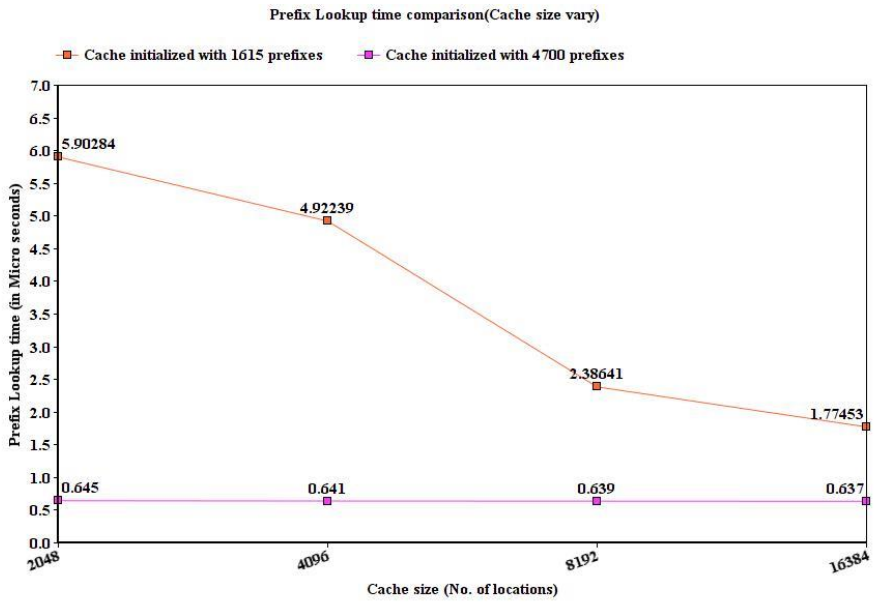
. Fig 3. Lookup time comparison for varying cache initializations

4.2 The lookup time is evaluated in another dimension by initializing the cache with certain number of prefixes and varying the number of locations in Cache. The Cache is varied by doubling the locations for which the lookup times could be found in Table 2. The graphical depiction is as shown in figure 4.

Table 2 : Lookup times with varying cache sizes and prefix initializations

Cache Size (No. of Locations)	Prefix Lookup time for 1615 initializations(in Micro seconds)	Prefix Lookup time for 4700 initializations(in Micro seconds)
2048	5.90284	0.645
4096	4.92239	0.641
8192	2.38641	0.639
16384	1.77453	0.637

Cache size plays a vital role in improving the performance. Larger the cache size, there will be provision to accommodate more number of prefixes. Besides increasing the number of prefixes in the cache initialization, if the Cache size i.e. the number of locations increases, there will be a significant improvement in the lookup speed. This imperative feature is proven here.



ISN: 2233-7857 IJFGCN

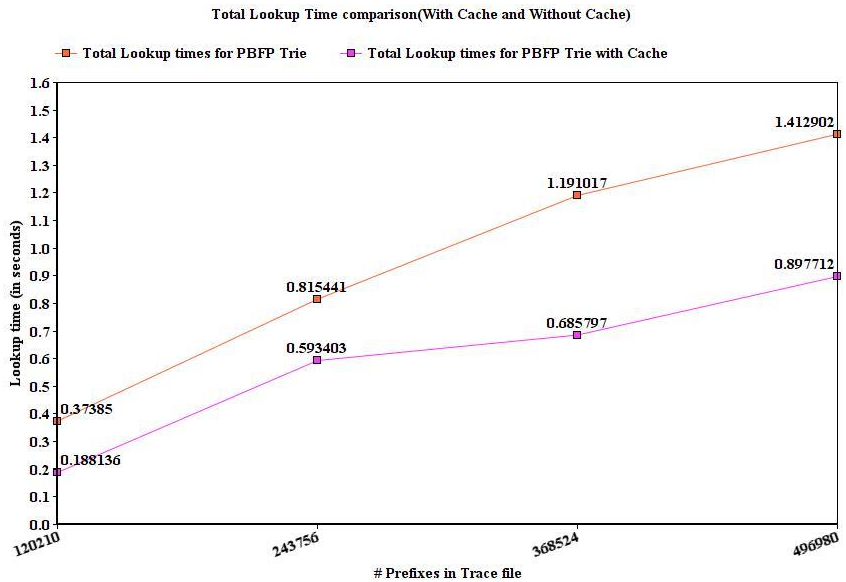
Fig 4: Lookup time comparison for varying cache sizes and cache initializations

4.3 Synthetic trace filesare generated with variable number of prefixes which could be observed from the table 3. The Cache size is fixed and total lookup time is evaluated. The graphical representation is as shown in the figure 5. The performance evaluation is done when cache is coalesced with the underlying data structure as well without cache augmentation.

Table 3 : Total lookup times with and without cache

No. of prefixes in the Trace File	Cache Size (No. of Locations)	Lookup time without cache(in seconds)	Lookup time with cache(in seconds)
120210	16384	0.373850	0.188136
243756	16384	0.815441	0.593403
368524	16384	1.191017	0.685797
496980	16384	1.412902	0.897712

Basically in [20] the underlying scheme exhibited a striking performance. In order to hike the performance the cache initialized with 4700 prefixes is combined. There is anincredibleimprovement in the lookup time resulting in around 40% rise. Of course increase in the input prefix file size proportionately increases the lookup time. It could be clearly understood that the lookup time with cache is amazing when without cache is used.



ISN: 2233-7857 IJFGCN

Figure 5 : Total Lookup time comparison with and without cache

4.4 The above same evaluation was repeated to compute the lookup time for a single prefix. The results are depict in table 4 and graphically presented in figure 6.

Table 4 : Average per prefix lookup time with cache and without cache

No. of prefixes in the Trace File	Cache Size (No. of Locations)	Prefix Lookup time without cache(in Micro seconds)	Prefix Lookup time with cache(in Micro seconds)
120210	16384	0.76428	0.6394
243756	16384	0.75952	0.6444
368524	16384	0.72121	0.6494
496980	16384	0.75722	0.6465

Similar improvement could be observed in case of average per prefix lookup time. Once the trace file is executed, depending on the number of hits and misses the time computation is performed and thus the average values are calculated.

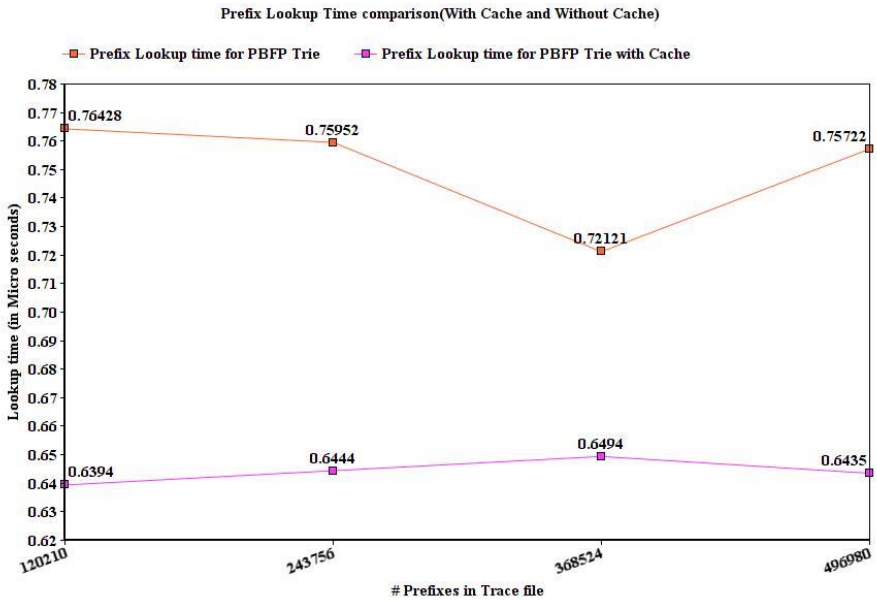


Fig 6. Per prefix lookup time comparison

ISN: 2233-7857 IJFGCN

4.5Hit ratio is another performance metric when cache is being used. The hit ratio must be high so as to get the desired performance. The performance is evaluated using a cache initialized with1615 prefixes besides varying the cache size. The computed results andthe graphical representation could be observed from Table 5 and figure 7.

Table 5 : Hit ratios for variable cache sizes

No. of prefixes with which Cache is initialized	Cache Size (No. of Locations)	Hit Ratio(%)
1615	2048	18
1615	4096	36
1615	8192	72
1615	16384	98

Hit ratio is the ratio of Number of hits to Number of hits plus misses. If the number of hits arehigh then the hit ratio increases. The hit ratio raised in proportion with the increase in cache size which means higher the cache size the hit ratios also increase accordingly. Proportionate to the cache size the number of prefixes could be accommodated. Consequently the lookup speed also increases. 98% Hit ratio was registered for a cache size of 16384 locations.

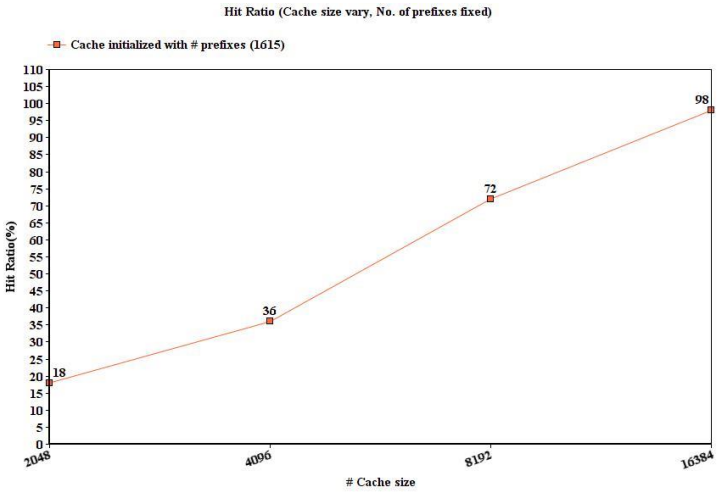


Fig 7 : Hit ratios comparison with varying cache sizes

4.6 Even when the number of prefixes initialized in the cache becomes more there will be a amazing hike in the hit ratio. Here in this evaluation the cache size is fixed and the

ISN: 2233-7857 IJFGCN

initialization prefixes are increased. The hit ratios for various cache initializations could be observed from table 6 and figure 8.

Table 6 : Hit ratios for varying cache initializations

No. of prefixes with which Cache is initialized	Cache Size (No. of Locations)	Hit Ratio(%)
0	16384	97.7
1615	16384	98
4700	16384	98.6

Despite the cache size being reasonable if the number of prefixes with the cache is initialized increases, the hit ratio also gets improved. It is due to that if the number of prefixes with which the cache is initialized increases then more number of prefixes will be available in the cache itself thereby increasing the hit ratio.

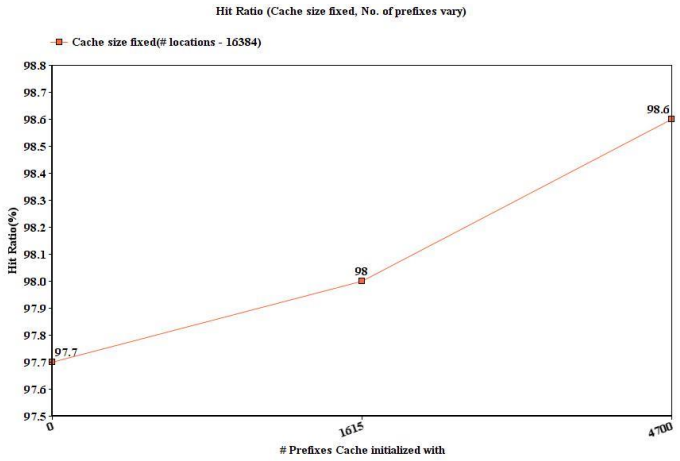


Figure 8 : Hit ratios comparison for varying prefix initializations

4.7 Lastly we also experimented for varying trace file size for which the hit ratios are registered as given in the table 7. The result is depicted in figure 9.

Table 7 : Hit ratios for variable trace file sizes

No. of prefixes in the Trace File	Cache Size (No. of Locations)	Hit Ratio
120210	16384	94.5
243756	16384	97.3

ISN: 2233-7857 IJFGCN

368524	16384	98.2
496980	16384	98.6

Although the hit ratio does not depend on the trace file size, since there may be more number of prefixes available in the cache the hit ratio seems to be high. As the trace file keeps increasing and if there are more number of misses, the hit ratio may not increase. Hence the hit ratio solely depends on Cache memory size, Cache initialization and Cache replacement policy.

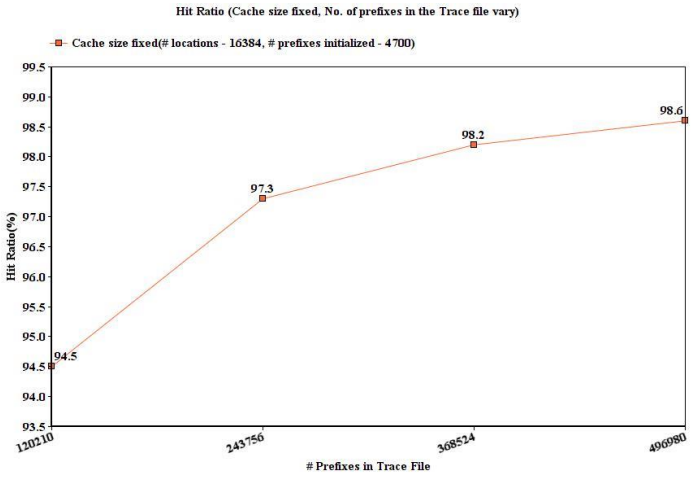


Figure 9: Hit ratios comparison for variable trace file sizes

Conclusion and Future work:

In this paper, we proposed an approach that uses cache in association to PBFP Trie to improve the lookup performance. We simulated the cache association and evaluate the results which describe the general nature of cache. The assessment shows that the lookup time when the basic data structure being used alone is more when compared to the values registered when cache is being used. Also it clearly illustrates that higher the hit ratio, lesser is the lookup time. Higher hit ratios could be accomplished using reasonable cache sizes, higher cache initializations and optimal Cache replacement strategies.

However the performance of the demonstrated design could still be made superior by using variant cache architectures viz. Heterogeneous Cache architecture [16], Set Associative Cache architecture [17], Hierarchical Cache architecture [15], Segmented Cache architecture [16], Multizone pipelined Cache [18] and implementing even the cache as PBFPT which is left as future work.

References:

ISN: 2233-7857 IJFGCN

[1] H. Lim and J. Mun, “An efficient IP address lookup algorithm using a priority-trie,” IEEE Globecom, pp. 15, Nov. 2006

[2]M. Ruiz-Sɔnchez, E. Biersack, and W. Dabbous, “Survey and Taxonomy of IP Address Lookup Algorithms”, IEEE Network Magazine, March/April 2001.

[3] Yaoqing Liu, Syed Obaid Amin, and Lan Wang, “Efficient FIB Caching usingMinimal Non-overlapping Prefixes”, ACM SIGCOMM Computer Communication Review, Volume 43, Number 1, January 2013, pp 15-21

[4] R.C. Chang and B.-H. Lim, “Efficient IP routing table lookup scheme”, IEE Pvoc. Cuiiiiiruii., Vol. 149, No. 2, April 2002, pp77-82.

[5]HouassiHichem and BilamiAzeddine, “IP address lookup for Internet routers using cache routing table”, IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 4, No 8, July 2010, pp 35-40.

[6] Junghwan Kim, Myeong-CheolKo, Moon Sun Shin and Jinsoo Kim A Novel Prefix Cache with Two-Level Bloom Filters in IP Address Lookup Appl. Sci. 2020, 10, 7198

[7] G. Narlikar and F. Zane, “Performance Modeling for Fast IP Lookups,” Proc. ACM SIGMETRICS, 2001.

[8] Guangdeng Liao, Heeyeol Yu, A New IP Lookup Cache for High Performance IP Routers, IEEE, 2010.

[9] T. Chieueh and K. Gopalan, “Improving Route Lookup Performance Using Network Processor Cache,” IEEE/ACM Supercomputing Conf., 2002.

[10] B. Talbot, T. Sherwood, and B. Lin, “IP Caching for Terabit Speed Routers,” Proc. IEEE Globcom, 1999.

[11] H. Liu, “Routing Prefix Caching in Network Processor Design,” Proc. Int’l Conf. Computer Comm. and Networks, 2001.

[12] G. Zhu, S. Yu and J. Dai, "An active routing prefix caching algorithm for IP address lookup," *2009 Fourth International Conference on Communications and Networking in China*, Xian, 2009, pp. 1-6, doi: 10.1109/CHINACOM.2009.5339861.

[13] Jinsoo KIM1and Junghwan KIM, “An Efficient Prefix Caching Scheme for Fast Forwarding in Named Data Networking”, Studies in Informatics and Control, 27(2) 175-182, June 2018

[14] T. Chiueh and P. Pradhan, "High-performance IP routing table lookup using CPU caching," *IEEE INFOCOM '99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320)*, New York, NY, USA, 1999, pp. 1421-1428 vol.3, doi: 10.1109/INFCOM.1999.752162.

[15] K. Rajan and R. Govindarajan, “Two-Level Mapping Based Cache Index Selection for Packet Forwarding Engines,” Proc. Int’l Conf. Parallel Architectures and Compilation Techniques, 2006.

[16] Kaushik Rajan and RamaswamyGovindarajan, Senior Member , “A Novel Cache Architecture and Placement Framework for Packet Forwarding Engines”, IEEE Transactions on computers, VOL. 58, NO. 8, pp.100910025, AUGUST 2009

[17] S. Kaxiras and G. Keramidas, "IPStash: a set-associative memory approach for efficient IP-lookup," Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies., Miami, FL, 2005, pp. 992-1001 vol. 2, doi: 10.1109/INFCOM.2005.1498328.

[18] S. Kasnavi, P. Berube, V. Gaudet, and J.N Amaral “A cache-based Internet Protocol address lookup architecture” IEEE, Computer Networks, Vol. 52, Issue 2, pp. 303-3268, February 2008.

[19] https://bgp.potaroo.net/index-bgp.html

ISN: 2233-7857 IJFGCN

[20] J SARADA LAKSHMI AND KUDA NAGESWARA RAO, 2020. (PBFP TRIE) PARTITIONED BLOOM FILTER WITH PRIORITY TRIE A NOVEL APPROACH FOR IP ADDRESS LOOKUP.International Journal of Advanced Research in Engineering and Technology (IJARET).Volume:11,Issue:12,Pages:1035-1051.

ISN: 2233-7857 IJFGCN