BOUNDARY CUTTING: AN EFFICIENT APPROACH FOR PACKET CLASSIFICATION ALGORITHM WITH ENHANCED SEARCH PERFORMANCE

#1MEGHANA KATUKOJWALA,
#2KATTA PRIYANKA,
#3B.RAMESH, Assistant Professor,
Department of Computer Science and Engineering,
SREE CHAITANYA INSTITUTE OF TECHNOLOGICAL SCIENCES, KARIMNAGAR, TS.

ABSTRACT: Several solutions were attempted in order to efficiently group packets based on their availability.Our study examined various different applications of decision trees. Using these trees, messages were classified into hundreds of categories.The seed set and branching depth of the decision tree were optimized for proximity to the ideal solution. The preceding explanation increases the need for higher storage capabilities while also increasing the time needed to execute a search.There are ways that employ decision trees to identify the best pair and multi-pair for each packet.

The importance of multi-match packet classification has grown in tandem with the popularity of more sophisticated network applications. It is critical to evaluate not only the primary factor itself to ensure that all of the results that match the main factor can be retrieved. Effective classification strategies must be discovered to avoid the difficulties of this procedure.To improve the efficiency of box sorting, a unique approach known as "boundary cutting" has been devised.Typical packet categorization applications necessitate significant matching.

The demonstrated method has two key advantages.By substituting rule boundaries for predefined gaps, the suggested method improves on earlier approaches to border-cutting.The aforementioned parameters have a significant impact on the quantity of RAM required.Removing boundaries makes internal nodes' data processing more challenging, yet binary search excels in this area.

Keywords: Decision tree algorithms, Packet classification, Boundary cutting, Priority matching, Binary search**.**

1. INTRODUCTION

Because of their ability to sort data packets, servers connected to the internet have a significant edge when it comes to offering specific services. Researchers have been focusing substantially on multimatch classification due to the increased demand for worm-finding and network-breaking technologies. A large bandwidth yet low on-chip memory integrated circuit is excellent for storing the packet categorization rule library. When off-chip memory is increased, device performance suffers.

It is critical to understand how much RAM is required to build a packet sorting table. The most important metric for evaluating packet classification systems is data processing efficiency. This is due to the fact that each incoming packet must be separately sorted at wire speed. To determine the packet type, the chip must perform a series of memory queries that go beyond its own internal memory.

The main goal of this study is to examine and analyze various decision tree-based classification algorithms for use in packet analysis. The goal was to find the best starting point and number of cuts for Hi Cuts and Hyper Cuts-style decision trees. The preceding explanation increases the need for higher storage capabilities while also increasing the time needed to execute a search. To be effective, this technique must first be processed using sophisticated formulas that are intricately linked to each set of regulations. Because their rule sets are retained off-chip buttheir inner nodes are kept on-chip, decision trees are fast to search.

A decision tree facilitates the identification of packets that require prioritized for sorting by quickly identifying the most critical matches. The capacity to sort messages into various categories while simultaneously displaying all possible outcomes and the most important matching criteria is a critical feature of today's network applications. Data classification problems can be solved using sophisticated and clever computer systems. We devised a novel way to packet clustering using the boundary-cutting technique during our inquiry. Each rule can eliminate the required volume by properly placing things. In this post, we will look at an innovative technique for quickly separating packets into discrete groups. By inspecting the region's borders, the machine can determine the precise region covered by each rule. This means that complex algorithms cannot match the precision of a human trimming method. It's also faster and uses less memory than before.

When partitioning techniques such as Hi Cuts or Hyper Cuts are used, there is no reduction in the total number of rules that apply to a subspace. This is because these strategies are only effective under certain situations. This study demonstrates the utility of the boundary cutting (BC) approach. Repeated use has proved the usefulness of this method for achieving clean, accurate cuts following the lines of each guide. When dividing a prefix plane into pieces based on rule boundaries, it is critical to consider more than just the beginning and ending lines of each rule. Decision tree approaches, regardless of strategy, only consider the subspace in which an input packet is located. In the first stage, we compare the raw data header-identified subspace to the rule fields. At various stages, each internal node of the border-crossing decision tree suffers unique reductions. Each node in the tree must execute a binary search to determine which path to take.

2. OVERVIEW OF EARLIER DECISION TREE AL-GORITHMS

Assuming k = 1, P must follow the previously described rule Rk. If d = 1, the value Fd must be assigned to every header field, resulting in a False result. In Rk, how many variables can you use and access at once? The fact that there are N rules and D fields leads to the formula (d) N = D * N. When multiple rules contribute equally to the classification of a packet, the rule with the highest priority is chosen and communicated back. Multiple criteria can be used to group packets.

Most collections of ideas can be neatly classified into five categories. In the first two fields, it is critical that the source and target prefixes match. The compatibility of sending and receiving terminals will be covered in more detail further down. To figure out how many cuts are needed, we must first compare the field to a constant field that fluctuates based on the methodology.

A collection of rules is used to decide the order of the leaf nodes in a decision tree. This is referred to as binth. By following these instructions, you will learn how to do a linear search. How does a rule come into effect? A rule is a two-dimensional (2D) region that contains the first two prefix domains. The goal is to determine which complex plane regions are covered by each rule in a given two-dimensional rule set. The maximum field length (W) of IPv4 and the amount of space required are indicated.

3. EXISTING SYSTEM

The primary goal of our research was to investigate the efficacy of decision trees for categorizing data into meaningful groups. The decision tree technique can be improved by dividing it into two independent parts. While many rules are kept off-chip, internal tree nodes are stored in on-chip memory. Decision tree algorithms can find the best way to cluster a large number of packet matches in addition to prioritizing the most important match. Hi cuts and hyper cuts are two popular versions of the standard decision tree approach. They decide the appropriate field and amount of cuts by examining which option results in the best performance. Both storage requirements and search times have been reduced. Careful planning is required before attempting this strategy. You must create comprehensive regulations for each and every unique set of rules.

4. DISADVANTAGES

Preprocessing processes necessitate a large investment of time and resources in order to design and execute.

Creating decision trees is a time-consuming and resource-intensive computer task. The solution is to expand algorithms with fresh sets of rules.

Regardless of the sector it controls, each regulation has a set time limit. This is why this method is seen as archaic.

5. PROPOSED SYSTEM:

- By outlining their borders, this essay illustrates an innovative and effective approach of arranging these bundles. The total number of rules is lowered because each proposed regulation addresses a distinct topic.
- The proposed method enables an estimate of the packet classification table. Unlike other decision tree systems, this one does not demand the use of complex techniques.

6. ADVANTAGES OF PROPOSED SYSTEM

- The investigated method surpasses prior methods due to its more sophisticated use of rule limitations rather than arbitrary thresholds. However, this results in less RAM being required.
- It is well known that the BC indexing approach is incapable of conducting the type of in-node searching that the binary search methodology enables.

7. IMPLEMENTATION

Developing a BC decision tree is a multi-step procedure:

The first and end locations of a rule serve as natural prefix plane bounds. However, because decision trees normally look for a subspace where an input packet is situated, the choice you select will have little effect on the ultimate result. The decision tree's nodes at the ends are used to compare data from an input file with domain-specific rules.

The Cutting Edge Knowledge Base:

The BC decision tree does not have predetermined breakpoints at any particular time. To find the associated edge for each key, a binary search must be done on each tree node. The reference to the child node is kept if the entered value is greater than or equal to the intended value. The preceding information can be derived from a message with the following headers: (000110, 111100, 19, 23, TCP). A binary search is done on the root node header of the data.

The main node header 000110 is compared to the intermediate item 010000. This search will only consider the input and the 000100 item due to the limited amount of information. Its scope is also restricted. The kid indicator on the second edge can be accessed as the input data grows. As a result, the search takes into account a larger pool of probable candidates. The provided data looks to be less than 001000, yet there are no records. The investigation has switched to this second edge since a signal was identified. The binary search method finds the header with the value 111100 from the next-to-last edge. Linear search, like the HiCuts and HyperCuts algorithms, is used to discover the rules within the leaf node.

Taking a predetermined route:

In this context, the BC technique is simplified. Binth is used in decision trees, such as the BC algorithm, to assess if a given region should be treated as a leaf node. It has been dubbed a "internal node" since it meets more conditions than a binary node. In this scenario, the node is referred to as a leaf node. To identify which region will act as the central hub, the BC method employs a selection procedure. Once partitions inside this subspace have been defined, the rules can begin at those partitions. The goal of this research is to look into a novel use of the binth technique for changing or eliminating the constraint imposed by a rule on the internal structure of a node. When the number of rules in a partition hits a certain threshold, the advanced framework ensures that no rules in that partition are allowed to progress to the next partition.

A well-structured database is highly valued in the data management industry:

There are two methods for incorporating rule sets into decision trees. The primary idea is to contrast rule tables and decision trees. Each rule is only kept in a single copy by the rule structure. Another important feature is that each leaf node in a decision tree is linked to the rule table that governs its behavior. Each rule point requirement for a binth leaf is detailed below. When looking for the best rule for a packet or the complete collection of matching rules, memory accesses rise. This is because determining how many rules are included within a node's leaves is straightforward. The second technique entails adding nodes just for the purpose of storing rules. When fewer rule tables are consulted, the time necessary to perform the search is lowered. Consistent execution of the guidelines, on the other hand, results in a significant improvement in recollection. If you want to keep rules on leaf nodes, search efficiency is more important than memory restrictions.

8. RELATED WORK

Many Internet services rely on the ability to categorize packets, such as network traffic monitoring and allowing routers to apply filters on packets. Tennant Content Addressable Memories (TCAMs) are often used in the workplace to separate rapid data. TCAMs operate by simultaneously comparing each message to all three criteria. Messages are sorted in the order they arrive. When traditional packet categorization rules that rely on field ranges are converted into TCAM-compatible rules, a phenomena known as "range expansion" occurs. As a result, there will be many more rules. However, if TCAMs have enough memory, this should not be a problem. Unfortunately, energy storage capacity in Ternary Content Addressable Memories (TCAMs) is extremely limited. Furthermore, when rules tighten, more energy and heat are generated.

New Internet services emerge on a regular basis, while packet analysts are subjected to an increasing number of laws. The goal of this study is to see how few TCAM (Ternary Content Addressable Memory) records are required to attain the same degree of accuracy as a cutting-edge packet classifier. In this article, we will look at the TCAM Razor. This is a fantastic time management tool, and you should make full use of it. TCAMs, or thermal control and tracking devices, are relatively expensive. TCAMs are 30 times more expensive per bit held than double-data-rate SRAMs. Instead, for the sake of this discussion,

If the port range field's length is limited, 2(L-1) TCAM entries may be required instead. This necessitates the investigation of other computational methodologies. The user specifies that they want to reformat the existing text in a scholarly manner without adding any new content. If k = 1, the supplied instance of rule Rk matches packet P correctly. The clustering of packets demonstrates the process. If d = 1, the result is presumed to be False, and all header fields are coded as Fd. If there are N rules and D fields, then d will have the value Rk, which represents the number of fields in the kth rule. When a packet only obeys one rule, that rule is chosen. There are several frameworks available for labeling packages. The applicable rules and regulations are outlined here.

Rule sets are commonly classified into five types. The prefixes in the first two fields must match between the source and the target. The interoperability of sending and receiving terminals will be discussed further in the sections that follow. A perfect match with the protocol-dependent, fixed variable is required to find the optimal number of cuts for the target variable. The order of the leaf nodes in a decision tree is determined by a set of rules. Binth is the term for this. Here are some guidelines for conducting a linear search.

HiCuts *:*

Each rule produces what is theoretically a five-dimensional hypercube as long as they stay within the bounds. Each data file's location is defined in the file's header. Using the method, the space is initially iteratively partitioned along each axis. This is because the number of intersecting rule hypercubes in each subspace has decreased. HiCuts allows you to add more nuanced information to the decision tree by increasing the number of cuts. However, when critical incisions are missing, the study process is substantially hindered. It may be difficult to establish a reasonable balance between the needs of long-term storage and frequent access. The HiCuts technique improves heuristic performance by modifying two parameters: the space factor (spfac) and the threshold (binth). These factors determine the complexity and memory requirements of the decision tree.

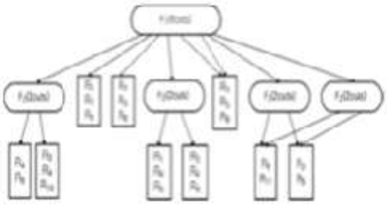


Fig:-HiCuts algorithm.

Having stated that, what exactly is a rule? A rule is defined as a two-dimensional (2D) region containing the first two prefix domains. The prefix "plane" is always used for all of the rules in the 2D set. Assume W is the maximum IPv4 field length supported. The most common answer is 32. Let F2 denote this specific field, and (i, j) denote the distance between two fields. If you know (W - i) and (W - j), you can compute the area using the "two times the difference" rule.

Hyper Cuts:

The Hyper Cuts method evaluates numerous fields at the same time, whereas the HiCuts technique considers only one field to determine the cut measures. The algorithmic decision tree was built using the Hyper Cut approach using the same data. The space is allocated a value of 3, whereas the binth is awarded a value of 1.5. In this situation, the combination of "fields" and "and" is particularly effective. The edges of the node are made up of binary string combinations of 00, 10, 01, and 11. Each conceivable combination is made up of two binary numbers, one from the first field and the other from the second.

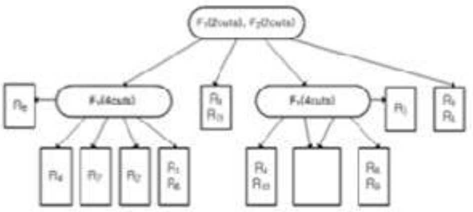
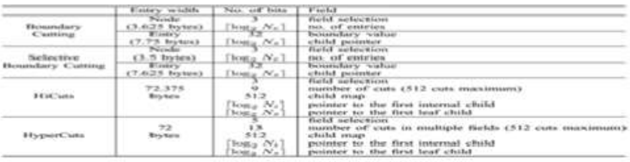


Fig:-hyper cuts

9. DECISION TREE CHARACTERISTICS

Part two entails building decision trees with a lower bound of binth using the BC, SBC, HiCuts, and Hyper Cuts algorithms. The pruning procedure begins when the number of rules in a subspace falls below a certain threshold (called binth). The efficiency of algorithms such as HiCuts and Hyper Cuts improves or degrades as available space increases. A non-optimized decision tree's terminal nodes are solely made up of binary rules. Working your way up the tree from its leaves, you can find the universally applicable rules to a subtree's progeny. The rules are then conveyed to the parent of the subtree. This step is often repeated multiple times, or until the decision tree's starting point is reached. In this instance, the binth number ensures that all of the rules observed at each node along the path from the root to the leaf are kept. While optimization does not improve search speed considerably, it does yield many more reusable rules.



	Entry width	No. of bits	Field
Boundary Cutting	Node (3.625 bytes)	3 $\lceil \log_2 N_e \rceil$	field selection no. of entries
	Entry (7.75 bytes)	32 $\lceil \log_2 N_n \rceil$	boundary value child pointer
Selective Boundary Cutting	Node (3.5 bytes)	3 $\lceil \log_2 N_e \rceil$	field selection no. of entries
	Entry (7.625 bytes)	32 $\lceil \log_2 N_n \rceil$	boundary value child pointer
HiCuts	72.375 bytes	3 9 512 $\lceil \log_2 N_i \rceil$ $\lceil \log_2 N_e \rceil$	field selection number of cuts (512 cuts maximum) child map pointer to the first internal child pointer to the first leaf child
HyperCuts	72 bytes	3 13 512 $\lceil \log_2 N_i \rceil$ $\lceil \log_2 N_e \rceil$	field selection number of cuts in multiple fields (512 cuts maximum) child map pointer to the first internal child pointer to the first leaf child

Fig:- What data structures are implemented at the decision tree's leaf nodes

Fig:- Rule sets may include fewer or more wild cards than others depending on their distinctive traits.

10. CONCLUSION

This study looks at the boundary cutting algorithm as well as many clustering strategies. During testing, the data is divided into subsets based on their degree of resemblance.

These categories are then used to determine whether a particular package represents a security concern. Many people believe that the boundary cutting algorithm is more precise than the grouping algorithm because of its capacity to precisely determine the edge of the space.

REFERENCES

1. H. J. Chao, “Next generation routers,” Proc. IEEE, vol. 90, no. 9, pp.1518–1588, Sep. 2002.
2. A. X. Liu, C.R.Meiners, andE.Torng, “TCAMrazor: A systematic approach towards minimizing packet classi- fiers in TCAMs,” IEEE/ACM Trans. Netw., vol. 18, no. 2, pp. 490–500, Apr. 2010.
3. C. R. Meiners, A. X. Liu, and E. Torng, “Topological transformation approaches to TCAM-based packet classification,” IEEE/ACM Trans. Netw., vol. 19, no. 1, pp. 237–250, Feb. 2011.
4. F. Yu and T. V. Lakshnam, “Efficient multimatch packet classification and lookup with TCAM,” IEEE Mi- cro, vol. 25, no. 1, pp. 50–59, Jan. –Feb. 2005.
5. F. Yu, T. V. Lakshman, M. A. Motoyama, and R. H. Katz, “Efficient multimatch packet classification for network security applications,” IEEE J. Sel. Areas Com-mun., vol. 24, no. 10, pp. 1805–1816, Oct. 2006.
6. H. Yu and R. Mahapatra, “A memory-efficient hash-ing by multi-predicate bloom filters for packet classifi- cation,” in Proc. IEEE INFOCOM, 2008, pp. 2467–2475.
7. H. Song and J. W. Lockwood, “Efficient packet clas- sification for network intrusion detection using FPGA,” in Proc. ACM SIGDA FPGA, 2005, pp. 238–245.
8. P. Gupta and N. Mckeown, “Classification using hi- erarchical intelligent cuttings,” IEEE Micro, vol. 20, no.1, pp. 34–41, Jan.–Feb. 2000.
9. S. Singh, F. Baboescu, G. Varghese, and J. Wang, “Packet classification using multidimensional cutting,” in Proc. SIGCOMM, 2003, pp. 213–224.
10. P. Gupta and N. Mckeown, “Algorithms for pack- et classification,” IEEE Netw., vol. 15, no. 2, pp. 24–32, Mar.–Apr. 2001