A Single-Cycle Multi-Match Packet Classification Engine Using TCAMs

Mehrdad Nourani and **Miad Faezipour**
Center for Integrated Circuits & Systems
The University of Texas at Dallas
Richardson, Texas 75083
{nourani, mxf042000}@utdallas.edu

***Abstract*— Most conventional packet classifiers find the highest priority filter that matches the packet. However, new networking applications such as network intrusion detection systems and load balancers require all (or the first few) matching results in packet classification. An efficient TCAM-based architecture for multi-match search is introduced in this paper. We propose a novel partitioning scheme based on filters and their intersection properties. An efficient contention resolver unit is designed to enhance performance of the search by choosing only one partition. Our approach finds all matches in exactly one conventional TCAM cycle while reducing the power consumption by at least two orders of magnitude.**

Keywords: Ternary content addressable memory, multi-match, packet classification, maximum-intersection partitioning, minimum-intersection partitioning, contention resolver.

I. Introduction

A. Background

Packet classification in general refers to finding the best matching filter containing multiple fields among the *filter* set (also called *rule* set in the literature) set for a given packet. The standard five-tuple fields include the source address, destination address, protocol, source port and destination port [1]. Among these fields, source and destination address fields are prefixes and often require longest prefix match (LPM) methods. Protocol field can be wild cards or exact values. Source and destination port numbers are typically introduced as ranges. Packet classification performs searching the table of filters to assign a flow identifier for the highest priority filter which matches the packet in all fields. The returning flow ID indicates the action that is next applied to the packet.

Filter fields are combination of prefixes, wild cards and exact values. Hence, Ternary Content Addressable Memories (TCAMs) that have the ability to store *don't-care* values in addition to 1's and 0's are often utilized to store filters and perform the parallel search in packet classification. A traditional packet classifier assigns one TCAM entry for each filter and finds the index of the highest priority matching filter in the database (Figure 1). Range fields may require more than one TCAM entry to save a filter. TCAM range matching solutions have been introduced in [1], [2], [3] and [4]. Each filter f_i $(0 \leq i \leq n-1)$ contains multiple fields (e.g. in the standard five-tuple), and also the filter database often may have up to hundred thousand filters. Therefore, wide TCAM devices both in terms of bits and entries are used for packet classification applications.

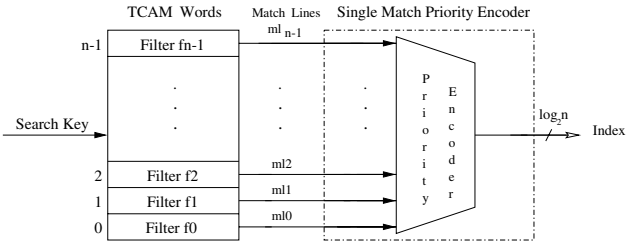

Figure 1. TCAM structure as a single-match packet classifier

B. Motivation

New emerging networking applications such as Network Intrusion Detection Systems (NIDS) and load balancers require finding all or the first few matching filters in packet classification. Malicious intrusions and denial-of-service attacks, that are expected to grow rapidly, can be monitored and detected by NIDS. Once all the matching filter headers are found, a detection system such as Snort [5] scans the packet payload for existing worms. The concept of multi-match classification for NIDS is becoming a major stream of research, since there is a great demand for network worm detections.

Packet level accounting, transparent monitoring and the Programmable Network Element (PNE) are other networking applications of multi-match classification. PNE is the general platform for packet processing in layers 2-4 in the edge. Packets entering PNEs are classified to identify the relevant functions. Multi-matching classification can be utilized to support multiple functions in PNEs [6].

Ideally, the single and multi-match problem in packet classification should be solved at wire speed to keep up with the high data rate. Pure software solutions suffer from low speed, since they often require several instructions and as a result several memory accesses to find a single or multiple matches. In the past few years researchers in industry and academia employed architectural solutions to the classification problem. This stream of research proposed the most widely used packet classification device technology, TCAM [7]. TCAMs are well suited for performing high speed parallel searches on database with ternary entries, since they provide the match results with deterministic throughput (i.e. one search per cycle) and deterministic capacity. Hence, TCAM has become quite popular for packet classification tasks [1], [3], [8], [9]. While TCAMs perform packet classification at high speed, there are some drawbacks that do not make them suitable for specific classification applications. TCAMs cannot directly report all possible

Proceedings of the 14th IEEE Symposium on High-Performance Interconnects (HOTI'06)
0-7695-2654-3/06 $20.00 © 2006 IEEE

matches in a database. This is due to the native structure of a TCAM cell design, which consists of a priority encoder, to produce the highest priority match. Another major drawback is the TCAM high power consumption due to searching all entries in parallel. Power consumptions in a TCAM grows drastically with the number of entries being searched, which leads to poor scalability.

C. Main Contribution & Paper Organization

We propose a parallel architecture for multi-matching packet classification by efficiently partitioning the entire packet filter set into disjoint subsets. Each subset is mapped to a relatively small TCAM which produce a match in one cycle. Since the TCAM entries remain unchanged in our design, our multi-matching hardware can be easily adopted for IPv6 where the bit width of the TCAM entries (filters) will be quite high [10], [11]. Performance of conventional software-based and most hardware-based classification techniques linearly degrades with number of filters and the number of matches to be found. Our system finds r matches in at most 1-cycle regardless of total number of filters and matches. Such property significantly improves the performance by achieving 1-2 order of magnitude higher speedup. Our partitioning scheme can also be employed as a low-power solution to the conventional single-match packet classification in general and multi-match packet classification in particular.

The rest of this paper is organized as follows. In Section II we take a glance at prior work related to TCAM-based multi-match packet classification. In Section III we describe our novel maximum-minimum intersection partitioning scheme. In Section IV the TCAM-based implementation for multi-match packet classification tasks is introduced. We summarize our experimental results in Section V. Finally, concluding remarks are in Section VI.

II. Prior Work

Some recent work focused on multi-match packet classification using TCAMs. Authors in [6] reorganize the TCAM entry filters in a compatible order to report all matches. The authors use a geometric intersection scheme to remove the overlaps and negation among the intersecting filters placed in the TCAM. Their multi-search mechanism is based on the TCAM search along with a SRAM to store all matching indices. This scheme has a high search throughput. However, an Access Control list (ACL) may require very large number of TCAM entries. This approach may not be suitable for such databases, since it cannot be easily scaled to large tables.

The Set Splitting Algorithm (SSA) introduced in [12] splits the filter set into two groups to remove at least half of the intersections among filters. It then performs the search on multiple groups in parallel. This method is based on minimum intersections among filters; however, it adds filters for partially overlapped filters in one set. In addition, it performs the search on all sets generated, either in parallel or sequentially; hence, it increases the search time and power consumption.

The authors in [2] address the problem of finding multiple matches in a TCAM by proposing the multi-match using discriminators (MUD) algorithm. In this algorithm, the extra bits per TCAM entry are used for the required encoding. For getting the next matching result from the TCAM after filter f_j, the search on all the entries after index j is performed. To accomplish this, the search key is expanded to prefixes that correspond to the range greater than j. The authors use the idea that along with each TCAM entry, a discriminator field that encodes the index of that entry is stored. The MUD algorithm provides high speed results for multiple matches. However, it deploys sophisticated encoding on TCAM entry databases, making it difficult to decode the data to their original values.

Entry-invalidation scheme, also discussed in [2], is one of the earliest and simplest schemes. In this method, a valid bit in addition to the header fields is associated with each TCAM entry. Initially, all entries have their valid bits set to "1". Searches are performed multiple times to find all matching entries. Each time a match is found, the valid bit is set to "0" for that matching entry. The same search key is applied again until all matches are found. In the entry-invalidation scheme, the state of the filter database is modified when the algorithm is applied. This characteristic makes it impractical for multi-threaded packet processors that are used nowadays, in which multiple packet processing threads to access to the TCAM devices at the same time is required.

The BV-TCAM architecture introduced in [9] combines the TCAM and the Bit Vector (BV) algorithm to address the problem of packet classification for network intrusion detection. The authors use the tree-bitmap implementation of the BV approach for the source and destination port lookup, while using a TCAM for the search of the other header fields. They use the un-encoded TCAM to report the matching status of the corresponding TCAM entry. The authors implement their classification engine in an FPGA. This approach improves the search and cost by compressing the data. However, the exact performance and architecture are not reported.

III. Intersection-Driven Partitioning Strategy

Intersection among filters in the database mainly results in multiple matches. Informally speaking, intersection is defined as having filters that are subset of one another, such that some filters completely overlap others. Therefore, partitioning the filter set based on filter intersections, and performing the TCAM search on a partition, can significantly improve the performance.

A. Maximum Intersection Partitioning Scheme

The Maximum Intersection Partitioning scheme or MXIP which is explained in this section, partitions the filters in the database such that each partition would hold the maximum number of intersections among its filters. This way all possible matches for a packet will be concentrated within one partition only. In addition, partitions will be disjoint, i.e. any pair of partitions do not have any overlap in the filters that they contain. Since there would always be a number of filters that do not have any intersection with any other filter, one last partition is needed in which all these *distinct* filters can be placed.

• Partitioning Heuristic:

Let $f_i[w-1:0]$ and $f_j[w-1:0]$ denote two filters of bit-width w. We define the term *distance* between the two filters as:

$$d_{i,j} = \sum_{k=0}^{w-1} f_i[k] \otimes f_j[k] \quad (1)$$

where, $\otimes$ in Equation 1 is a three-valued operation, defined using *XOR* operation, as follows:

$$a \otimes b = \begin{cases} 0 & \text{if at least one of } a \text{ or } b \text{ is a } \textit{don't-care} \\ a \oplus b & \text{otherwise} \end{cases} \quad (2)$$

Suppose F refers to the set of all filters, and P_m denotes the m-th partition. In every step the first element in F (i.e. f_1) is the seed of partition. The process grows the partition around the seed based on the maximum intersection partitioning heuristic. N_p would be the total number of partitions generated based on the MXIP scheme. n_m is the total number of filters in partition P_m. Hence, N_p partitions $(P_1, P_2, \cdots, P_m, \cdots, P_{Np})$ will be formed, from which the last partition (P_{Np}) is a collection of all distinct filters.

Figure 2 illustrates an example of a small filter set of 10 filters partitioned based on maximum intersections. Filters are assumed to be 8-bits long for simplicity. Filters f_1, f_4, and f_{10} have zero distance, hence they can form one partition, i.e. P_1. Also, filters f_8 and f_9 have zero distance with filter f_{10}, therefore they are also placed in P_1. Filters f_5 and f_6 make a zero distance with filter f_2, and form partition P_2. Finally, filters f_3 and f_7 that have no zero distance with any other filters, form the distint filter collection, and are placed in a separate partition (P_3). In this example, if the search key "11010010" arrives, partition P_1 would be chosen and the search is performed on a relatively small set containing partition P_1 only.

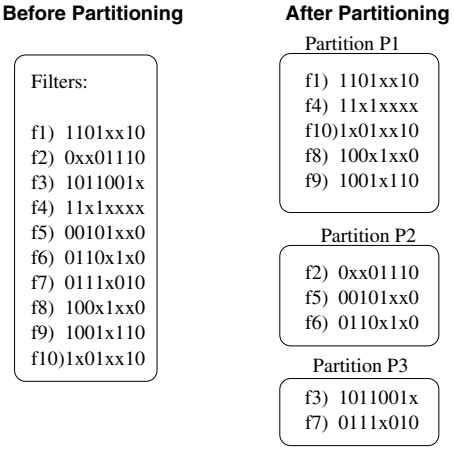

Figure 2. Example of Maximum Intersection Partitioning.

Maximum intersection partitioning would ensure that all possible matches for a given search key are located in one partition. One possible architecture of a multi-match packet classifier using the MXIP approach is illustrated in Figure 3. All filters in each partition are placed in one TCAM module. The single-match priority encoder unit is replaced by a Multi-Match Prioritizer (MPZ) circuit along with an address encoder. The MPZ unit is a customized prioritizer circuit that gives all the match lines in a prioritized sequence. The MPZ and encoder circuit

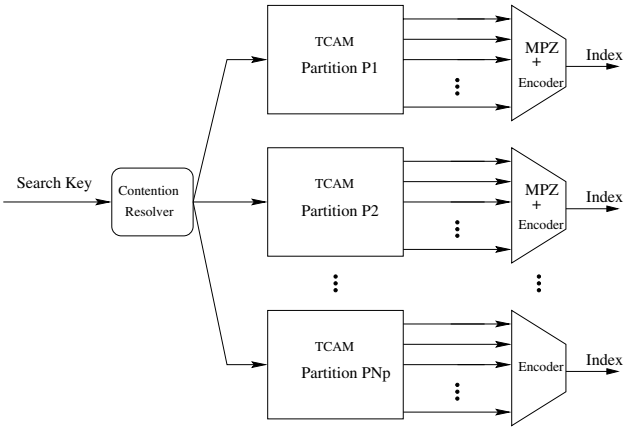

Figure 3. A classifier engine using MXIP approach.

connected to the TCAM provide the addresses of the r matches in at most r cycles [13]. Note that the last partition (P_{Np}) does not need a MPZ unit since it would result in at most one match, and therefore an address encoder is sufficient. Having a contention resolver in this architecture is not required as all partitions can be searched in parallel. However, as we discuss in Section IV.A, having such resolver can minimize power consumption.

B. Maximum-Minimum Intersection Partitioning Scheme

By partitioning the database based on maximum number of intersections among filters, multiple matching filters will be positioned within one partition. All filters in one partition are placed into a single TCAM and the MPZ connected to the TCAM match lines can provide all r matches in r cycles. However, this approach is costly since it involves additional hardware circuitry.

By further partitioning the maximum intersected filters intelligently, all r matching addresses can be found in one cycle. The second partitioning is based on *minimum intersections* among the filters in each partition. This time, sub-partitions are generated for as many number of completely overlapping filters. Filters in each sub-partition should have a distance greater than zero among each other. This indicates that after minimum-intersection partitioning there are no two filters that have a hundred percent overlapping in each sub-partition, unlike the maximum intersection partitioning where overlaps were concentrated in one partition.

Assume that N_p is the number of partitions generated by the maximum intersection partitioning method. P_m denotes the m-th partition based on maximum intersection partitioning. This procedure provides $P_{m,z}$ that refers to the z-th partition in P_m generated by the minimum intersection partitioning (MNIP) method.

If all sub-partitions generated by the MNIP approach are placed in separate TCAMs, minimum-intersection partioning would result in finding all matches in one cycle. This is because of the fact that if the entire filter set is partitioned in such a way that each partition would result in at most one match for a given packet, since all partitions are searched in parallel, all matching

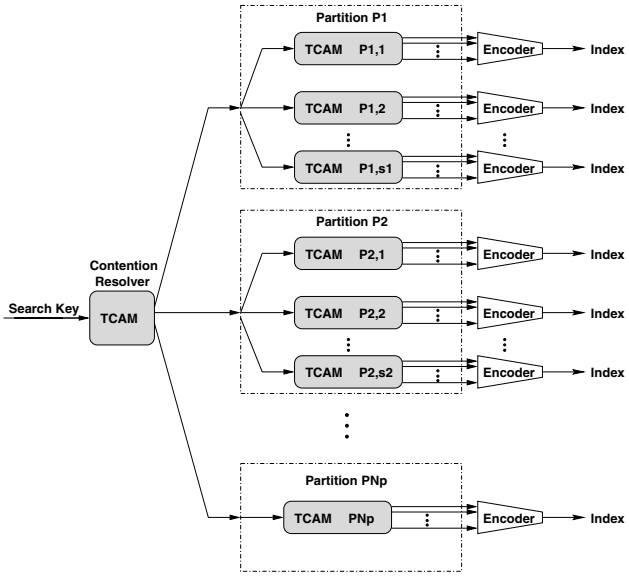

Figure 4. A classifier engine using MX-MN-IP approach.

addresses are ensured to be provided in only one cycle. The classifier architecture that employs the Maximum-Minimum-Intersection-Partitioning (MX-MN-IP) scheme is shown in Figure 4. Matching indices can be stored in a SRAM and can be further processed by other networking units.

Note carefully that in this method the conventional (off-the-shelf) TCAM cell chips can be used to perform the parallel search, which makes it very cost effective. Nowadays TCAM vendors provide the ability to search one or several smaller blocks in a single TCAM chip in parallel [12]. Partitions, constructed by the MXIP scheme, are further partitioned into smaller sets. The small sets (i.e. $P_{m,z}$) are placed into TCAM blocks.

IV. Hardware Optimization and Implementation

In our system large partitions will not be required. Maximum number of possible filters (that MXIP pushes into the same partition) depends on number of fields considered. Using practical filter sets, 2-field and 5-field classification produce around 150 and 8 matches, respectively [12]. Therefore, due to nature of MXIP method, the expected (average) size of partitions will be in the range of 8 to 150. The MX-MN-IP approach is highly efficient since:

- It provides all matching addresses in at most 1-cycle.
- It does not add any extra filter to the whole set, unlike others, e.g. the SSA scheme [12], that add new filters for partially overlapping filters in each partition. This feature makes the MX-MN-IP approach, much more memory efficient and scalable.
- By enabling the TCAM search on one partition and disabling others, the power consumption is significantly reduced. Power consumption is directly proportional to the number of entries being searched in parallel in a TCAM. The MXIP method effectively partitions the database, hence for each search key, only a small portion is being searched, while all others remain idle.

A. Contention Resolver

To achieve the power savings mentioned by partioning, a *Contention Resolver Unit* should be designed so that the search mechanism would result in enabling the TCAM search on one partition, while disabling others. An identification code for each partition (a representative of the filters in that partition) can be defined to facilitate choosing the right partition.

Let $ID_m[k]$ denote the bit position k for the ID code of a particular partition (i.e. P_m), where $0 \leq k \leq w-1$ and $1 \leq m \leq N_p - 1$ and n_m ($|P_m|$) denotes the number of filters in the *m-th* partition generated by MXIP. $ID_m[k]$ will be 0, 1 or x if $f_i[k]'s$ are all 0s, all 1s or include xs, respectively. The ID code indicates how the filters in one partition are intersected. Hence, if for each partition a unique ID code is generated, when the packet arrives, an initial search based on searching these ID codes in parallel would result in one ID. All the ID codes can be placed in a small TCAM, as shown in Figures 3 and 4. The arriving packet is compared against these IDs, and the match line of matching ID would enable the partition it represents. Therefore, only one partition (TCAM) performs the search, while others are remained idle. Since there is one partition (the last one) containing distinct filters with no zero distance among them, the ID encoding cannot be applied to this partition. In other words, for this partition no unique ID can be defined. As a result, this partition should be searched only if there is no match found at other partitions.

To be more clear, Figure 5 shows the minimum-intersected partitions for the example and how they are placed in TCAMs. Figure 5 also shows the ID codes for the partitions generated for the example. Note that partition P_3 is the set of distinct filters, and thus does not hold any unique ID. The ID codes for partitions P_1 and P_2 are placed in a TCAM unit to perform the initial search. As shown, ID codes are generated based on the union of all maximum intersected filters in one partition. If packet "11010010" arrives, the initial TCAM (contention resolver TCAM) would enable the first maximum-intersected partition, and the search key would be sent to all TCAMs generated based on the second partitioning heuristic. This packet matches filters f_1, f_4 and f_{10}, and their addresses are provided in one cycle.

B. Updating Filters

The partitioning approach based on maximum or minimum intersections should provide valid partitioning results for multiple matches, even after insertion of a new filter and deletion of an existing one.

- **Filter Deletion:** Updating the packet classification database may result in the same partitions when one filter is deleted. However, if a filter that provided zero distance with a few set of filters, is deleted, given that those set of filters have at least one nonzero distance with each other, that partition should be split into two separate partitions. This is due to the fact that the filter that combined some filters together, no longer exists. Hence, some set of filters

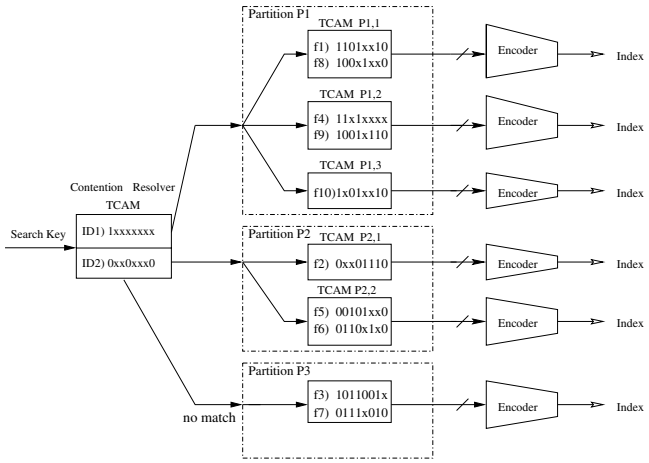

Figure 5. Classifier engine using the MX-MN-IP approach along with the Contention Resolver.

do not have any zero distance with the other sets in that partition. This leads to creating a new partition, where the set of filters are separated. In our running example, if filter f_{10} is deleted from the P_1 set, filters f_1 and f_4 do not have any intersection with filters f_8 and f_9 anymore. Therefore, partition P_1 would be divided into two partitions; one containing filters f_1 and f_4, and the other containing filters f_8 and f_9.

- **Filter Addition:**
  In most cases, adding a new filter does not change the partitions very much. This filter must be compared against the filters of each partition, to see where a zero distance is generated. The new filter should be added to the partition in which it makes a zero distance. However, in case this filter has a zero distance with more than one partition, filters in all those partitions are to be combined into one partition.
- **ID Update:** Updating, may also alter the ID codes in the contention resolver. The ID code for each partition can be generated only after all filters in that partition are known. Based on the ID encoding, the insertion or deletion of a filter may cause a bit position in the ID code to change from 0 or 1 to x or vice versa as reflected earlier in this section.

C. Power Saving

Performing the TCAM search on only a small portion of the entire database can significantly save power consumption due to the frequently charging and discharging of the highly capacitive match line. Power consumption is directly proportional to the number of entries being searched in parallel in a TCAM. The MXIP method effectively partitions the database, hence for each packet, only a small portion is being searched, while all others remain idle. In addition, since each TCAM (partition) would result in at most one match, the priority encoder unit would not be required anymore. This not only reduces the cost, but also further improves the power saving. The priority encoder unit, which is a power hungry unit in a conventional TCAM structure, would be removed. Instead, a conventional address encoder can be used to provide the addresses of the matching results.

The Maximum-Minimum-Intersection partitioning approach can also be used in packet processors where the conventional single-match packet classification is desired. Traditionally, TCAMs perform the search and provide the index of the highest priority match in one cycle. However, our approach performs the TCAM search on a small fraction of the entire filter set, and thus reduces the power consumption by at least one order of magnitude.

TABLE I
PARTITIONING STATISTICS

r_{max}	$\Delta\%$	$\|P\|_{max}$	$\|P\|_{avg}$	N_p	$\|P_{Np}\|$
12	30	12	9	88	4245
8	30	8	6	124	4225
4	30	4	3	263	4090
8	10	8	6	42	4742
8	20	8	6	81	4487
8	50	8	6	202	3705
8	70	8	6	289	3171

V. EXPERIMENTAL RESULTS

A. Partitioning Implementations

We have applied our partitioning scheme to various randomly generated filter sets that were designed to have the same characteristics as real filters sets, e.g. numbers, sizes and structures of filters. Table I shows the average statistics of the partitions generated. The filter sets were assumed to have 5000 filters of bit-width 150 each, which is typical for real databases. The second column shows the percentage of filters that generate multi-match results, i.e. $\Delta = \frac{\sum_{m=1}^{Np-1} |P_m|}{\sum_{m=1}^{Np} |P_m|}$. The maximum size of partitions is $|P|_{max}$ and $|P|_{avg}$ is the average size of partitions. N_p is the total number of partitions generated by the MXIP scheme.

As observed from the table, most partitions have very few entries, and most filters are in the distinct filter collection. This significantly reduces the power consumption by about 2-3 orders-of-magnitude as in multi-match applications the last big partition is rarely awakened.

B. Speedup

The speed of our architecture is high compared to other conventional hardware based designs. Delay time of a conventional TCAM can be written as:

$$T_{TCAM} \approx T_{word} + T_{PE} \quad (3)$$

where T_{word} is the delay of the TCAM word and T_{PE} is the delay of the priority encoder. For large TCAMs: $T_{word} \approx T_{PE} \approx T_{TCAM}/2$ [14]. Our approach thoroughly removes the need of the priority encoder unit, and hence T_{PE} does not come into picture. The delay of our design consists of the delay of the contention resolver TCAM plus the MNIP search time along with the regular address encoder circuit delay. Therefore:

$$T_{MX-MN-IP} \approx T_{CR} + T_{Pm} + T_{Encoder} + T_{P_{Np}} \quad (4)$$

It can empirically be shown that the search delay of a small TCAM (i.e. CR unit and TCAMs holding sub-partitions) is around $T_{TCAM}/4$ compared to large conventional TCAMs, and that the encoder unit would also have a latency of approximately $T_{TCAM}/4$. In the MX-MN-IP approach the worst case is assumed to be searching one of the maximum intersected partitions and searching the set of distinct filters afterwards. Hence:

$$T_{MX-MN-IP} \approx T_{TCAM}/4 + T_{TCAM}/4 + T_{TCAM}/4 + T_{TCAM}/4 \quad (5)$$

Thus, the MX-MN-IP performs the multi-match classification in 1-cycle: $T_{MX-MN-IP} \approx T_{TCAM}$.

The Maximum-Minimum-Intersection partitioning approach can find all matches for a given packet and provide the matching addresses in at most 1-cycle. Practically, the CR TCAM would be very small and hence, the resolver unit does not add significant delay. The priority encoder unit is removed from all TCAMs in the design. These facts ensure the 1-cycle system performance. A conventional multi-match TCAM based approach would spend at least $r \cdot (7 \times T_{TCAM})$ cycles for finding r matches [2]. Our approach finds r-matches in one conventional TCAM cycle, which is far better than software and hardware approaches. Hence, we obtain speedup of at least $S = \frac{r \cdot (7 \times T_{TCAM})}{T_{TCAM}} = 7r$ for finding r matches, where r would be the maximum number of possible matches in a filter set for a given search key. For a maximum of 8 matches we achieve speedup of 56 compared to the conventional approach, which is more than an order-of-magnitude higher. Note that in our architecture, performance does not degrade when the number of matches increases. Scalability feature makes our design attractive for high speed packet processing. For similar reasons our design can be used in single-match packet classification for high speed network processors.

C. Cost

The cost of a conventional TCAM can be defined as: $A_{TCAM} \approx A_{word} + A_{PE}$. The area of our classifier engine can be written as: $A_{MX-MN} \approx A_{word} + A_{Encoder} + A_{CR}$. The area of the encoder and the small contention resolver unit is less than a conventional priority encoder unit. Hence, the overall cost (area) of our classifier is approximately the same as a conventional (off-the-shelf) TCAM-based classifier.

VI. CONCLUSION

We proposed a TCAM-based architecture for multi-match packet classification. All multiple matches can be found in at most one conventional TCAM cycle. Our design uses the concept of maximum and minimum intersection among filters to efficiently partition the entire filter set. Using MX-MN-IP scheme, we achieve speedup of 1-2 order of magnitude higher compared to conventional TCAM based methods. Our approach is highly efficient for the future's network processors where performance of packet processors would become a bottleneck. Power consumption was reduced by one order of magnitude or more, due to performing the TCAM search on a small portion of the packet filter set.

REFERENCES

[1] K. Zheng, H. Che, Z. Wang, and B. Liu, "TCAM-Based Distributed Parallel Packet Classification Algorithm with Range-Matching Solution," *INFOCOM'05*, 2005.

[2] K. Lakshminarayanan, A. Rangarajan and S. Venkatachary, "Algorithms for Advanced Packet Classification with Ternary CAMs," *ACM SIGCOMM'05*, Aug. 2005.

[3] E. Spitznagel, D. Taylor and J. Turner, "Packet Classification Using Extended TCAMs," *Proceedings of the 11th IEEE International Conference on Network Protocols, pp. 120-131*, Nov. 2003.

[4] H. Liu, "Efficient Mapping of Range Classifier into Ternary-CAM," *Proceedings of the 10th Symposium on High Performance Interconnects Hot Interconnects, HOTI '02*, Aug. 2002.

[5] SNORT Network Intrusion Detection System, *www.snort.org*.

[6] F. Yu, R. H. Katz and T. V. Lakshman, "Efficient Multimatch Packet Classification and Lookup with TCAM," *IEEE Computer Society*, Jan. 2005.

[7] D. E. Taylor and E. W. Spitznagel, "On Using Content Addressable Memory for Packet Classification," *Technical Report WUCSE-2005-9*, March 2005.

[8] F. Yu, R. H. Katz and T. V. Lakshman, "Gigabit Rate Packet Pattern-Matching Using TCAM," *Proceedings of the 12th IEEE International Conference on Network Protocols, pp. 174-183*, 2004.

[9] H. Song and J. W. Lockwood, "Efficient Packet Classification for Network Intrusion Detection Using FPGA," *Proceedings of the 2005 ACM/SIGDA 13th International Symposium on Field-Programmable Gate Arrays*, Feb. 2005.

[10] N. F. Huang, W. E. Chen, J. Y. Luo and J. M. Chen, "Design of Multi-field IPv6 Packet Classifiers Using Ternary CAMs," *IEEE Conference on Global Telecommunications, vol. 3, pp. 1877-1881*, Nov. 2001.

[11] N. F. Huang, K. B. Chen and W. E. Chen, "Fast and Scalable Multi-TCAM Classification Engine for Wide Policy Table Lookup," *19th IEEE International Conference on Advanced Information Networking and Applications, vol. 1, pp. 792-797*, March 2005.

[12] F. Yu, T. V. Lakshman, M. A. Motoyama and R. H. Katz., "SSA: A Power and Memory Efficient Scheme to Multi-Match Packet Classification," *ACM Proceedings of the 2005 Symposium on Architecture for Networking and Communications Systems ANCS '05, pp. 105-113*, Oct. 2005.

[13] M. Faezipour and M. Nourani, "Design and Implementation of a TCAM-Based Architecture for Multi-Match Packet Classification," *Technical Report, UTDEE-12-2005*, Dec. 2005.

[14] M. J. Akhbarizadeh, M. Nourani and C. D. Cantrell, "Segregating the Encompassing Prefixes to Enhance the Performance of Packet Forwarding Engines," *IEEE Global Telecommunications Conference, GLOBECOM'04, pp. 1612-1616*, Nov.-Dec. 2004.