Range-enhanced packet classification to improve computational performance on field programmable gate array

Anita Ponnuswamy, Manju Devi
Department of Electronics and Communication Engineering, The Oxford College of Engineering,
Visvesvaraya Technological University, Bangalore, India

Article Info

Article history:

Received May 15, 2021
Revised May 31, 2022
Accepted Jun 29, 2022

Keywords:

Field programmable gate array
Header fields
Matching operations
Packet classification
Range bit-vector encoding
Prefix range
Range enhanced
Ruleset

ABSTRACT

Multi-filed packet classification is a powerful classification engine that classifies input packets into different fields based on predefined rules. As the demand for the internet increases, efficient network routers can support many network features like quality of services (QoS), firewalls, security, multimedia communications, and virtual private networks. However, the traditional packet classification methods do not fulfill today's network functionality and requirements efficiently. In this article, an efficient range enhanced packet classification (REPC) module is designed using a range bit-vector encoding method, which provides a unique design to store the precomputed values in memory. In addition, the REPC supports range to prefix features to match the packets to the corresponding header fields. The synthesis and implementation results of REPC are analyzed and tabulated in detail. The REPC module utilizes 3% slices on Artix-7 field programmable gate array (FPGA), works at 99.87 Gbps throughput with a latency of 3 clock cycles. The proposed REPC is compared with existing packet classification approaches with better hardware constraints improvements.

Corresponding Author:

Anita Ponnuswamy
Department of Electronics and Communication Engineering, The Oxford College of Engineering,
Visvesvaraya Technological University
Bangalore, India
Email: anita.p.research@gmail.com

1. INTRODUCTION

The network routers provide many features like quality of services (QoS), scheduling, access control, and security. These features in network routers can differentiate between different packets and decompose or classify them into the flow. The collection of packets has the same header features with different payloads known as flow mechanisms. Network routers provide packets with a proper flow mechanism that agrees on the set of predefined rules. The classifier collects these rules, and each rule in the packet classifier defines the flow of the packet, and it belongs to which field [1]. The packet classifier has met metrics like fats updating, fields used, memory requirements, flexibility, and search speed to improve network computational performance. In general, packet classification approaches are classified based on multiple field search techniques like Tuple space, decision-tree (DT) based, decomposition, and exhaustive search. The tuple space approach includes rectangle search, pruned tuple search, and tuple search algorithms. Similarly, the decision tree classification approach includes a grid of tries, Hypercuts, HiCuts, and modular packet classification algorithms. The decomposition approach includes cross producing, parallel bit-vector, aggregated bit-vector, and recursive-flow classification (RFC). Finally, the exhaustive search contains ternary content addressable memory (TCAM), emulated TCAM, linear search, and bit map insertion approaches [1]–[3].

The TCAM packet classifier approach provides high throughput. Still, it lags with higher power consumption by using parallel architectures, and also range-encoding methods are not meet the computational requirements. The example of range encoding for the Packet classification system is represented in Figure 1, which improves the system performance with more significant power reduction. Firstly, convert the packet header (IP/TCP) information to an encoded packet search key using a lookup table (LUT). Then, the encoded ranges are available in the rule table. For example, the internet protocol (IP) source address value (32 bit) is converted into 110 using LUT, and it has to match the encoding rule (4th entry of the rule table is matched) according to the rule table [4]. With further advancement, multi-range encoding methods are used to solve the storing range problems in the TCAM classification approach and also overcome the drawbacks of the single-range field methods [5], [6].

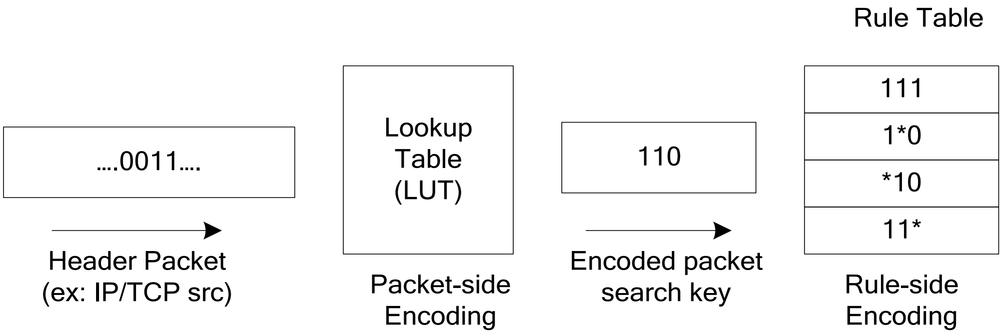

Figure 1. An example of a 1-dimensional range encoding for packet classification system [4]

The hardware-based packet classification approaches like TCAM and hardware accelerators are used in most network applications using wired speed classification. These hardware architectures use ASIC/FPGA-based algorithms for packet classification to improve the development cost and network performance. In addition, the most recent hardware packet classification approaches are used in most network applications like network security systems [7] and gateway designs [8]. These approaches provide greater flexibility with high performance and suit real-time network applications.

In this article, an efficient range enhanced packet classification (REPC) module is designed to improve the computational performance of the network system on the field programmable gate array (FPGA) platform. The present work supports range to prefix features and not dependent on any rule set while classifying the packets using different fields. The proposed REPC is explained in detail with its hardware architecture in section 2. Section 3 provides the results and performance analysis of the REPC module and also comparative analysis with improvements. Finally, section 4 concludes the overall work with improvement.

An overview of the existing packet classification methods with different application approaches are discussed below and: Linan *et al.* [9] present the improved cutting-based multi-dimensional packet classification method. The HyperCut is considered a multi-dimensional packet classification (PC) method, which improves the analysis of filter sets and statistics of the HyperCut algorithm. The decision-making process is incorporated to estimate the performance with IPv6 classification and achieves the best throughput and searching performances. In [10], [11] explain the scalable multi-field PC module suitable for multi-core processors. The design includes three field types, namely, preprocess, search, and merge types. The range tree building process the prefix range and matches with the help of a hash table. The range-tree search approach is used to find the range match using Cuckoo hashing and finally merge these types using bit-vector (BV) based bitwise AND operation. Wang and Hengkui [12] describe the TCAM based PC approach with a better packet forwarding rate. The TCAM approach decreases the TCAM memory by reducing the space effectively for the repetitive information's using unique rules. Hsieh and Weng [13] present a multi-dimensional cutting approach in scalable many-field PC with the help of selective bit-concatenation. This work addresses the issues of ruleset sparsity exploitation and rule-filed dependencies. Finally, Li and Yu [14] describe the online-flow level PC approach for a multi-core processor. This approach is designed using a flow-table and decision tree (FTDT) and classifies the incoming packets according to the flow table. This method is compared with the HyperCut approach with superior improvement in throughput.

Brack *et al.* [15] present the network packet classification approach using a just-in-time (Jit) vector algorithm. The Jit algorithm is build using the bit-vector classification method and generates the code output instantly at run time. Zheng *et al.* [16] explain the PC method with total prefix length-based clustering, which partitions the rules into different cluster modules and building the quad-tree to achieve better memory utilization and search speed. This method also achieves the high-speed PC by providing dynamic updates of rule sets. Finally, Harada *et al.* [17] present the inclusive rules-based PC with better acceleration. This method provides a proper rule reconstruction approach, optimizing the rule set by overwriting it as an inclusive rule. Li and Shao [18] present the RFC method with memory compression for network processing devices. This method uses the multiple-match table (MMT) as a flow table that provides excellent scalability and feasibility in PC. Yu *et al.* [19] present the SRAM-based PC approach in memory access using a bit selection approach. The selected bits are used as an input key to access the ruleset. Finally, Huang *et al.* [20] explain the hybrid PC approach using the hash table and geometric space partition (HGSP). The HGSP provides better classification speed by maintaining the same accuracy by incorporating a parallel hash table. In addition, the HGSP reduces around 40% matching time than the HiCuts algorithms.

2. RANGE ENHANCED PACKET CLASSIFICATION

The range enhanced packet classification is similar to a simple bit vector approach. The multiple pipelined stages are introduced when the ruleset is large. The header extractor provides a different address, which passes through many operating stages for range fields in each pipelined stage. These range fields are processed through to each pipelined stage of the prefix fields. The hardware architecture of the REPC Module is represented in Figure 2. The REPC mainly contains a packet generation unit (PGU), header extractor unit (HEU), range bit-vector encoding (RBVE) module for source and destination address along with ports, and lastly, a matching unit.

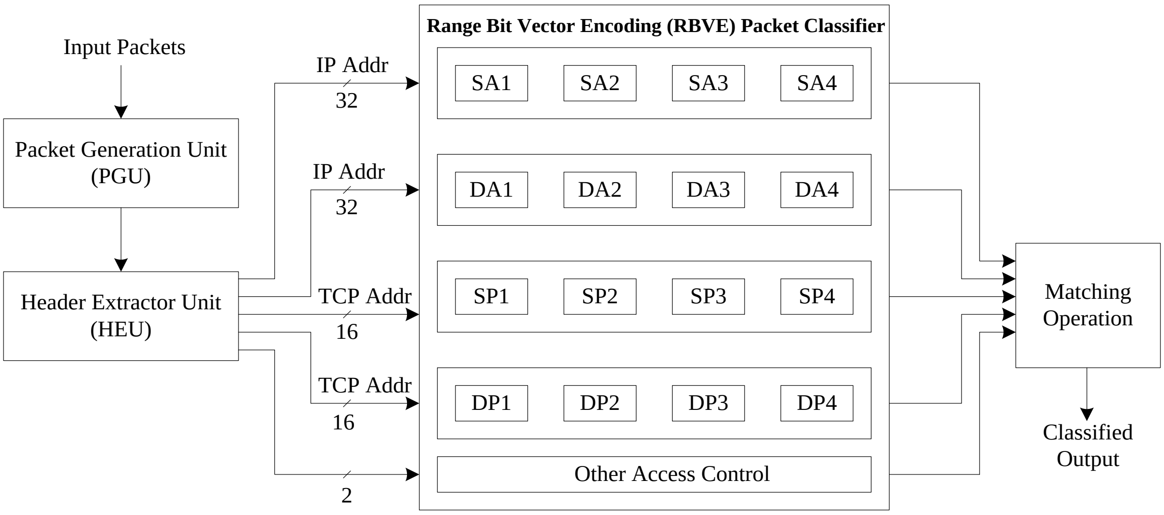

Figure 2. Hardware architecture of REPC module

The PGU receives the incoming data sequentially and generates valid packets. The PGU also checks the possible error conditions to avoid future problems in the packet classification process. The PGU was initiated with the start of the packet (Sop) and ended with end of packet (EoP), which is the last packet valid packet received for the classification process. The HEU receives the valid packets from the PGU and processes them further for the generation of header fields. The HEU generates IP and transmission control protocol (TCP) header fields. The IP field is used further as the 32-bit source and destination addresses. Similarly, TCP Field generates the the16-bit source and destination ports. TCP field also generates the other access control initialization like virtual light at night (LAN) and user-defined protocol (UDP). These IP and TCP header fields are used in REPC. The field lengths used in REPC are tabulated in Table 1. The REPC supports all 5 fields: IP source address, IP destination address, TCP source port, TCP destination port, virtual LAN (VLAN), and UDP access control. In addition, each entry in the table has a specific value and prefix.

The range bit-vector encoding method also follows the same procedure as conventional packet classification rules. Each rule table has its source and destination address, containing 32-bit ranges, and similarly, source and destination ports containing 16-bit ranges. For source and destination address (b), the 32-bit range is divided into four parts. Each part is an 8-bit range. Similarly, for source and destination port (b), the 16-bit range is divided into four parts. Each part is a 4-bit range. The RBVE method, the 32-bit range, is decomposed into many d-bit sub-ranges. *d* is a fixed stride size, and it can be 1, 2, 4, and 8. There are *j* d-bit sub-ranges used as pipelined stages of *j* stages for each fixed stride size *d*. where j=32/d. The *d* is fixed to 8 for source and destination address range fields, so the number of pipelined stages used in the RBVE Module is *j*=4. Similarly, *d* is fixed to 4 for source and destination ports range fields, so the number of pipelined stages used in the RBVE module is *j*=16/4=4. The RBVE module architecture in general for each field is represented in Figure 3.

Table 1. Field lengths used in REPC

Fields used	Bits used
IP source address	32
IP Destination address	32
TCP source port	16
TCP destination port	16
VLAN access control	1
UDP access control	1

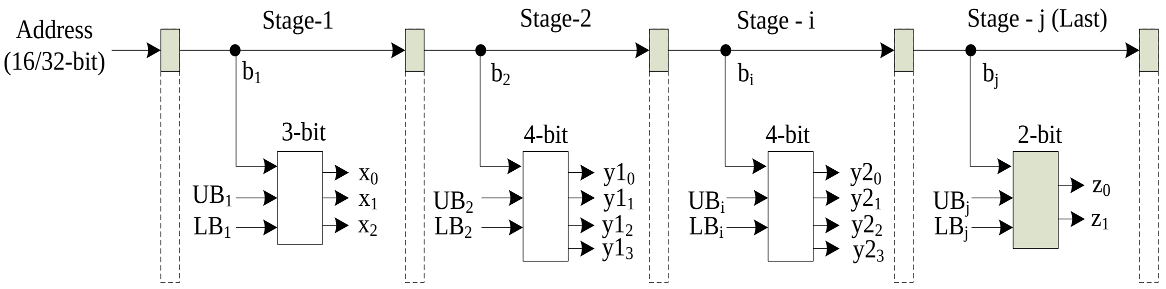

Figure 3. General RBVE module for each field

The upper bound (UB) and lower bound (LB) are the predefined ranges. These are denoted to 8-bits for each pipeline stage in source and destination address fields. Similarly, the UB and LB are predefined and have 4-bit values in each pipeline stage in source and destination port fields. The [LB, UB] are divided into four 4-bit/8-bit sub-ranges [LBi, UBi], where *i*=1 to 4 for source and destination port/address fields respectively. The input address (b) is 16/32-bits is divided into four sub-address bi where *i*=1 to 4 for source and destination port/address fields, respectively.

For each pipelined stage, the output signals description for RBVE is tabulated in Table 2. The output signals 000 and 111 will always indicate mismatch and match conditions for (LB1<b1<UB1) and (LB1>b1>UB1), respectively. The output signal 001 will match only on (UB1<LB1) and the same for the next following stages. Similarly, the output signal 010 will match only on (b1=UB1=LB1) and the same for the next following stages. The output signal 100 will match only on (LB1>UB1) and the same for the next following stages.

Table 2. Each stage output signals description

Output Bits	Description
000	Mismatched, but not depending on the next following stages
001	Matched, depending only on UB of the next following stages
010	Matched, depending on both LB and UB of the next following stages
100	Matched, depending only on LB of the next following stages
111	Matched, but not depending on the next following stages

The output signal generation for each stage in the RBVE method is tabulated in Table 3. In the RBVE method, the 1st stage, intermediate and last stages use 3-bit, 4-bit, and 2-bit outputs. The 16-bit/32-bit address (*bi*) is input to the RBVE module with fixed stride *d*. The 3-bit outputs ($x_2x_1x_0$) are computed in the 1st stage using b_1, LB_1, and UB_1. Similarly, the 4-bit outputs (y3y2y1y0) are computed in the intermediate stage using *bi*, LBi, and UBi, where *i* is 1 to *j*. Finally, the 2-bit outputs (z_1z_0) are computed in the last stage using *bj*, *LBj*, and *UBj*. Once stage outputs are generated in the RBVE method, perform as in (1)-(5) to obtain the match results.

$$\mathrm{m1} = \mathrm{x0\&x1\&x2} \tag{1}$$

$$\mathrm{m2} = (\mathrm{x0\&y10})|(\mathrm{x2\&y12}) \tag{2}$$

$$\mathrm{m3} = (\mathrm{x0\&y11\&y20})|(\mathrm{x2\&\,y13\,\&y22}) \tag{3}$$

$$\mathrm{m4} = (\mathrm{x0\&y11\&\,y21\&\,z0})|(\mathrm{x2\&y13\&y23\&z1}) \tag{4}$$

$$\mathrm{m5} = (\mathrm{x1\&(y10\&\,y12)\&(y20\&\,y22)\&(z0|z1)}) \tag{5}$$

Table 3. RBVE output generation for each stage

If condition	3-bit output ($x_2x_1x_0$) for 1st stage
Initialization $x_2x_1x_0$=000	
(LB_1>b_1) & (b_1>UB_1)	111
(b_1=UB_1) & (UB_1<LB_1)	001
(b_1=UB_1) & (b_1=LB_1)	010
(b_1=LB_1) & (LB_1> UB_1)	100
If condition	**4-bit output (y3y2y1y0) for intermediate stage**
Initialization $y_3y_2y_1y_0$=0000	
b_i >LB_i	0001
b_i=LB_i	0010
b_i<UB_i	0100
b_i=UB_i	1000
If condition	**2-bit Output (z_1z_0) for the last stage**
Initialization z_1z_0=00	
(b_j=LB_j) & (b_j>LB_j)	01
(b_j=UB_j) & (b_j<UB_j)	10

The final range enhanced matched output for the source address (M_{sa}) is obtained by performing the OR operation of (1)-(5), and it is represented in (6).

$$\mathrm{Msa} = \ \mathrm{m1} \mid \mathrm{m2}|\ \mathrm{m3} \mid \mathrm{m4} \mid \mathrm{m5} \tag{6}$$

The same RBVE method is applied to the destination IP address field, source, and destination ports to obtain the corresponding matched results for a destination address (M_{da}), source port (M_{sp}), destination port (M_{dp}). The additional control signal information (VLAN and UDP) is also accessed by initializing proper protocol specifications and generating the matched results. The final classified output is obtained for all the above fields by concatenating with them, and it is represented in (7).

$$\text{Classified Output} = \{\mathrm{Mvlan}, \mathrm{Mudp}, \mathrm{Mdp}, \mathrm{Msp}, \mathrm{Mda}, \mathrm{Msa}\} \tag{7}$$

The complete range enhanced packet classification is prototyped on the FPGA platform, which provides high throughput for rule sets in different range fields. The memory consumption is also reduced by utilizing proper rule sets with different ranges for packet classification. In addition, the REPC module uses the unique values in each field, which is relatively less than the rule set size and provides more excellent performance in packet classification.

3. RESULTS AND DISCUSSION

The REPC module is designed and prototyped on Artix-7 FPGA. The REPC architecture is modeled using Verilog-HDL on the Xilinx ISE environment and simulated using the Model-sim 6.5f simulator. The Hardware design constraints like chip area, time, power, Latency, and throughput are analyzed for REPC on Artix-7 FPGA and tabulated in Table 4.

The REPC utilizes 3,090 slices, 1,910 LUT's, and 1,880 LUT-FF pairs on Artix-7 chip. The REPC works at 492.804 MHz with a minimum period of 2.029 ns. The REPC utilizes 0.098 W total power with the inclusion of 0.016 W dynamic power using an X-power analyzer. The REPC executed and obtains the classified output within 3 clock cycles in the simulation environment. The REPC throughput was calculated based on latency, many packets injected, and the maximum frequency of the design. The throughput of REPC is 99.87 Gbps.

Table 4. Artix-7 FPGA resource summary for REPC module

Resources	Utilization
Chip area	
Slice registers	3,090
Slice LUT's	1,910
LUT-FF pairs	1,880
Time	
Minimum period (ns)	2.029
Max. frequency (MHz)	492.804
Power	
Dynamic Power (W)	0.016
Total Power (W)	0.098
Latency and throughput	
Latency (Clock cycles)	3
Throughput (Gbps)	99.87

The performance metrics comparison of proposed REPC with existing packet classification approaches is analyzed and tabulated in Table 5. The distributed cross production of field labels (DCFL) is one of the packet classification approaches [21], dependent on the rule set majorly, and used 90 byte/rule as a memory. The DCFL utilizes 5 clock cycles and works at 19 Gbps without range to prefix features. The bit-vector ternary content addressable memory (BV-TCAM) is the hybrid combination of the BV and TCAM approach [22], which is used to prefix the exact values lookup tables with large rule sets. The BV-TCAM works at 75 Gbps by utilizing 11 clock cycles without range to prefix features.

Table 5. Performance metrics comparison

Packet classifier designs	Memory (Byte/rule)	Latency	Throughput (Gbps)	Ruleset dependencies	Range-to-prefix
DCFL [21]	90	5	19	High	No
BV-TCAM [22]	154	11	75	High	No
Emulated TCAM [23]	24	1	64	Low	Yes
Stride-BV [24]	52	31	111	No	No
REPC (This work)	16	3	99.87	No	Yes

The emulated TCAM [23] with memory-efficient architecture analyzes different key length searches with fewer rules. The emulated TCAM works at 64 Gbps on Stratix-series FPGA and utilizes only 1 clock cycle latency. The stride BV packet classification [24] supports scalable, modular features and works at 111 Gbps throughput. The stride BV consumes a latency of 31 clock cycles and without range to prefix feature along with independent ruleset features. The proposed REPC supports the range to prefix feature for different rules without any dependencies. The REPC supports 16 byte/rule memory, works at 99.87 Gbps on Artix-7 FPGA with a latency of 3 clock cycles. The proposed REPC provides better performance metrics than the existing packet classification approaches [21]–[24].

The range enhanced packet classifier module is compared with BV packet classifier [25] and modified BV packet classifier [26]. The comparison results are tabulated in Table 6. The compared packet classifiers are also implemented on Artix-7 FPGA. The BV packet classifier utilizes 3,636 slices, 2,641 LUT’s, consumes 0.103 W total power, executes at 5 clock cycles, and obtains 61.94 Gbps throughput. The modified BV packet classifier utilizes 3,110 slices, 2,167 LUT’s, consumes 0.1 W total power, executes at 4 clock cycles, and obtains 74.95 Gbps throughput. However, the REPC module provides better performance overhead, like 15.01% in slices, 27.67% in LUT’s, 4.85% in total power, 40% in latency, and 37.97% in throughput than the BV packet classifier module [25]. Similarly, the REPC module provides better performance overhead, like 1% in slices, 11.85% in LUT’s, 2% in total power, 25% in latency, and 21.94 % in throughput than the modified BV packet classifier module [26].

The BV packet classifier is a conventional classifier approach that consumes more resources and more time to classify the packet. In contrast, the modified BV packet classifier approach is an extension of BV packet classifier, including source and destination ports and range search. The modified BV packet classifier is a better resource than the BV packet classifier. The proposed REPC provides better performance metrics than the existing BV packet classifier and modified BV packet classifier.

Table 6. Range enhanced packet classifier module comparison with existing packet classifiers [25], [26]

Resources	BV packet classifier [25]	Modified BV packet classifier [26]	RE packet classifier
FPGA device	Artix-7	Artix-7	Artix-7
Slices	3,636	3,110	3,090
LUT's	2,641	2,167	1,910
Total power (W)	0.103	0.1	0.098
Latency (Clock cycles)	5	4	3
Throughput (Gbps)	61.94	74.95	99.87

4. CONCLUSION

In this article, an efficient range is enhanced. REPC method is designed using RBVE approach and prototyped on Artix-7 FPGA. The REPC generates the packets using a PGU, followed by a HEU. HEU generates the IP and TCP header fields used as a source/ destination address and ports, respectively. The RBVE approach receives the IP/TCP address and provides the matched output-based rules and prefix fields. The REPC method supports Range to prefix features and not dependent on the ruleset. The REPC utilizes a 3% chip area (slices), works at 492.804 MHz, and consumes 0.098 W total power on Artix-7 FPGA. The REPC provides better throughput of 99.87 Gbps by consuming a latency of 3 clock cycles. The proposed REPC is also compared with existing packet approaches with better overhead in design constraints and performance metrics.

REFERENCES

[1] D. E. Taylor, "Survey and taxonomy of packet classification techniques," *ACM Computing Surveys*, vol. 37, no. 3, pp. 238–275, Sep. 2005, doi: 10.1145/1108956.1108958.

[2] A. Yudhana, A. Fadlil, and E. Prianto, "Performance analysis of hashing methods on the employment of app," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 5, pp. 3512–3522, Oct. 2018, doi: 10.11591/ijece.v8i5.pp3512-3522.

[3] B. Adil, L. Chakir, and E. Q. Abderrahime, "CHN and swap heuristic to solve the maximum independent set problem," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 7, no. 6, pp. 3583–3592, Dec. 2017, doi: 10.11591/ijece.v7i6.pp3583-3592.

[4] X. He, J. Peddersen, and S. Parameswaran, "LOP_RE: Range encoding for low power packet classification," in *2009 IEEE 34th Conference on Local Computer Networks*, Oct. 2009, pp. 137–144, doi: 10.1109/LCN.2009.5355199.

[5] M. H. Dahri, M. H. Jamaluddin, M. Inam, and M. R. Kamarudin, "Mutual coupling reduction between asymmetric reflectarray resonant elements," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 3, pp. 1882–1886, Jun. 2018, doi: 10.11591/ijece.v8i3.pp1882-1886.

[6] O. Erdem and A. Carus, "Range tree-linked list hierarchical search structure for packet classification on FPGAs," in *2013 International Conference on Reconfigurable Computing and FPGAs (ReConFig)*, Dec. 2013, pp. 1–6, doi: 10.1109/ReConFig.2013.6732324.

[7] W. Pak and Y.-J. Choi, "High performance and high scalable packet classification algorithm for network security systems," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2015, doi: 10.1109/TDSC.2015.2443773.

[8] S. P. Guruprasad and B. S. Chandrasekar, "Performance evaluation of network gateway design for noc based system on FPGA platform," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 9, 2019, doi: 10.14569/IJACSA.2019.0100937.

[9] C. Linan, L. Zhaowen, M. Yan, H. Xiaohong, and L. Chunqiang, "Multidimensional packet classification with improved cutting," in *2014 4th IEEE International Conference on Network Infrastructure and Digital Content*, Sep. 2014, pp. 409–413, doi: 10.1109/ICNIDC.2014.7000335.

[10] Y. Qu, S. Zhou, and V. K. Prasanna, "Scalable many-field packet classification on multi-core processors," in *2013 25th International Symposium on Computer Architecture and High Performance Computing*, Oct. 2013, pp. 33–40, doi: 10.1109/SBAC-PAD.2013.29.

[11] Y. R. Qu, S. Zhou, and V. K. Prasanna, "A decomposition-based approach for scalable many-field packet classification on multi-core processors," *International Journal of Parallel Programming*, vol. 43, no. 6, pp. 965–987, Dec. 2015, doi: 10.1007/s10766-014-0325-6.

[12] K. Wang and W. Hengkui, "TCAM-PC: Space-efficient TCAM-based packet classification with packet-forwarding-rate constraints," in *2015 12th IEEE International Conference on Electronic Measurement and Instruments (ICEMI)*, Jul. 2015, pp. 260–264, doi: 10.1109/ICEMI.2015.7494168.

[13] C.-L. Hsieh and N. Weng, "Scalable many-field packet classification using multidimensional-cutting via selective bit-concatenation," in *2015 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, May 2015, pp. 187–188, doi: 10.1109/ANCS.2015.7110133.

[14] W. Li and X. Yu, "An online flow-level packet classification method on multi-core network processor," in *2015 11th International Conference on Computational Intelligence and Security (CIS)*, Dec. 2015, pp. 407–411, doi: 10.1109/CIS.2015.104.

[15] S. Brack, S. Hager, and B. Scheuermann, "JitVector: Just-in-time code generation for network packet classification," in *2015 IEEE 40th Conference on Local Computer Networks (LCN)*, Oct. 2015, pp. 161–164, doi: 10.1109/LCN.2015.7366296.

[16] S. Zheng, X.-A. Bi, and J. Luo, "An efficient total prefix length-based clustering packet classification algorithm," in *2016 International Conference on Network and Information Systems for Computers (ICNISC)*, Apr. 2016, pp. 46–49, doi: 10.1109/ICNISC.2016.020.

[17] T. Harada, K. Tanaka, and K. Mikawa, "Acceleration of packet classification via inclusive rules," in *2018 IEEE Conference on Communications and Network Security (CNS)*, May 2018, pp. 1–2, doi: 10.1109/CNS.2018.8433137.

[18] X. Li and Y. Shao, "Memory compression for recursive flow classification algorithm in network packet processing devices," in

2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Oct. 2018, pp. 1502–1505, doi: 10.1109/IAEAC.2018.8577888.

[19] W. Yu, S. Sivakumar, and D. Pao, “Pseudo-TCAM: SRAM-based architecture for packet classification in one memory access,” *IEEE Networking Letters*, vol. 1, no. 2, pp. 89–92, Jun. 2019, doi: 10.1109/LNET.2019.2897934.

[20] J. Huang, Y. Lu, and K. Guo, “A hybrid packet classification algorithm based on hash table and geometric space partition,” in *2019 IEEE Fourth International Conference on Data Science in Cyberspace (DSC)*, Jun. 2019, pp. 587–592, doi: 10.1109/DSC.2019.00095.

[21] D. E. Taylor and J. S. Turner, “Scalable packet classification using distributed crossproducting of field labels,” in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, 2005, vol. 1, pp. 269–280, doi: 10.1109/INFCOM.2005.1497898.

[22] 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-FPGA ’05*, 2005, doi: 10.1145/1046192.1046223.

[23] C. A. Zerbini and J. M. Finochietto, “Performance evaluation of packet classification on FPGA-based TCAM emulation architectures,” in *2012 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2012, pp. 2766–2771, doi: 10.1109/GLOCOM.2012.6503535.

[24] T. Ganegedara, W. Jiang, and V. K. Prasanna, “A scalable and modular architecture for high-performance packet classification,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 5, pp. 1135–1144, May 2014, doi: 10.1109/TPDS.2013.261.

[25] A. Ponnuswamy and M. Devi, “Design of a low latency and high throughput packet classification module on FPGA platform,” *International Journal of Innovative Technology and Exploring Engineering*, vol. 9, no. 6, pp. 1468–1474, Apr. 2020, doi: 10.35940/ijitee.F4195.049620.

[26] A. Ponnuswamy and M. Devi, “High performance modified bit-vector based packet classification module on low-cost FPGA,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 5, pp. 3855–3863, Oct. 2021, doi: 10.11591/ijece.v11i5.pp3855-3863.

BIOGRAPHIES OF AUTHORS

Anita Ponnuswamy is a Research Scholar in the department of ECE at The Oxford College of Engineering Bangalore. She has worked as an Assistant professor at CMRIT, Bangalore. She has also worked as Lecturer at GVIT, KGF. She obtained her B. E (ECE) degree in 2002 from (GVIT) Bangalore University, M.Tech degree in VLSI Design and embedded system from CMRIT (Bangalore), pursuing PhD from Visvesvaraya Technological University (VTU), Karnataka. She has almost nine years of academic teaching experience and worked for both NBA and NAAC. She has almost 6 publications in international journals. Her areas of interest are VLSI design, analog, and digital electronics. She can be contacted at email: anita.p.research@gmail.com.

Manju Devi is working as a Professor and Head in the ECE department at The Oxford College of Engineering Bangalore. In addition, she has worked as Vice-Principal and Professor at BTLIT, Bangalore. She obtained her B.E (ECE) degree in 1996 from Anna University, M.Tech degree in Applied Electronics from BMSCE, and PhD from Visvesvaraya Technological University (VTU), Karnataka. She has almost twenty-two years of academic teaching experience and worked for both NBA and NAAC. She has more than 98 publications in international conferences and journals. She is guiding eight students from Visvesvaraya Technological University (VTU), Karnataka. Her areas of interest are VLSI design, analog and mixed-mode VLSI design, and digital electronics. She can be contacted at email: manju3devi@gmail.com.