

SSA: A Power and Memory Efficient Scheme to Multi-Match Packet Classification

Fang Yu, T.V. Lakshman, Martin Austin Motoyama, Randy H. Katz

Presented by: Amy M. Freestone
Discussion led by: Sailesh Kumar

Packet Classification

- Each incoming packet is compared to a set of filters to find those that match.
- For most traditional applications, the only match returned is the one with the highest priority, such as the longest prefix match.

Multi-Match Packet Classification

- There is an increasing demand for multi-match packet classification.
 - * Multiple counters may need to be updated for each packet.
 - * Network intrusion detection systems may wish to match multiple rules with a packet.

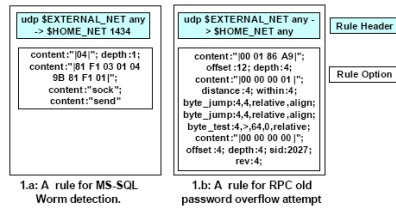


Figure 1. SNORT rule examples.

Figure Source: Yu et al.

3

November 17, 2005 – CSE 7702

Amy M. Freestone

Multi-Match Issues

- It is a potential bottleneck because it is performed on every packet.
- An approach with a deterministic and high lookup rate is necessary to process packets quickly enough.
- TCAMs can perform parallel searches quickly in hardware but are expensive and have a power consumption which grows linearly with the number of parallel searches and grows as the frequency of accesses increases.

4

November 17, 2005 – CSE 7702

Amy M. Freestone

Overview of Existing Solutions

- MUD scheme (Lakshminarayanan et al.)
 - * TCAM memory linearly related to size of filter sets
 - * Requires k lookups to get k matching results, with each entry accessed on each lookup
- Geometric Intersection-based solution
 - * All filter intersections must be entered in the TCAM as new filters.
 - N filters with F fields can theoretically produce $O(N^F)$ intersections.

5

November 17, 2005 – CSE 7702

Amy M. Freestone

Set Splitting Algorithm (SSA)

- SSA splits the filters into a number of separate sets, upon each of which separate TCAM lookups are then performed.
- Benefits
 - * Low memory usage
 - * Low power consumption
 - * Deterministic lookup rates
 - * Support of parallelism
 - * Low update cost

6

November 17, 2005 – CSE 7702

Amy M. Freestone

TCAM Basics

- Fixed-length entries, each with several cells which can be used to store a string
- Sends a bit vector of all matches to a priority encoder, which finds the first match
- Each cell can have one of 3 states: 0, 1, ?

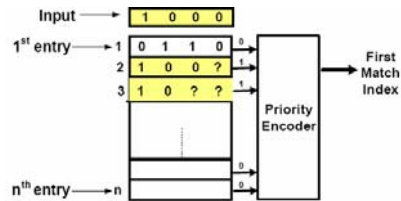


Figure 3. A TCAM.

Figure Source: Yu et al.

7

November 17, 2005 – CSE 7702

Amy M. Freestone

TCAM Limitations

- Costs approximately 30 times more per bit of storage than DDR SRAMs
- Consumes 150 times more power per bit than SRAMs
- Power consumption grows linearly with the number of entries searched in parallel
- Power consumption directly related to number of TCAM accesses

8

November 17, 2005 – CSE 7702

Amy M. Freestone

Different Approaches

- Partitioning the TCAM so that each packet only requires that several partitions be searched
 - * Designed for one dimensional classification
 - * CoolCAMs and load balancing TCAMs
- Organizing TCAM as a two-level hierarchy with an index block which enables or disables the main blocks' query processes
 - * Spitznagel et al.

9

November 17, 2005 – CSE 7702

Amy M. Freestone

Bit Vector Solution

- Remove the priority encoder
- Requires the CPU or NPU to pull out the matching results
 - * $O(N)$ processing complexity
- No simple way to return only the matching results

10

November 17, 2005 – CSE 7702

Amy M. Freestone

Current Industrial Solutions

- Multiple matching in some commercial TCAMs
- Each entry has a valid bit which says whether it should be compared with the entry, with all the valid bits initially set to valid.
- The TCAM reports the first match in the first cycle, sets its valid bit to invalid, and continues the process.
 - * $7k$ cycles are needed to identify k matching results
 - * All entries searched each cycle

11

November 17, 2005 – CSE 7702

Amy M. Freestone

MUD Solutions

- Multi-match Using Discriminators
- 144-bit TCAM entries can hold the 104-bit 5-tuple commonly used of packet classification with room left over for a discriminator
- Discriminator of a packet originally set to don't cares, given value 'greater than j ' after a match is found at index j
- Necessary expansion of 'greater than j ' may cause multiple TCAM lookups to get the next match
- $1+d+(k-2)*(d-1)$ lookups ($d = \log_2 N$) to get k matches
- $1+d^*(k-1)/r$ with DIRPE (r is smaller than N)

12

November 17, 2005 – CSE 7702

Amy M. Freestone

MUD Solutions

- No per-search state in TCAM
 - * Can be used in multi-threaded environments
- TCAM access per packet is linear to number of matching results.
 - * Up to 20 lookups in the worst case
- All entries accessed in every lookup
 - * High power consumption

13

November 17, 2005 – CSE 7702

Amy M. Freestone

Geometric Intersection-based Solutions

- Intersection filters added to handle packets that match multiple filters
- One TCAM lookup sufficient
- N filters with F fields can theoretically produce $O(N^F)$ intersection filters

14

November 17, 2005 – CSE 7702

Amy M. Freestone

Software Solutions

- Most algorithms for single-match
 - * Some can be extended
- Based on usual characteristics of single match filter sets
- At most 20 matching rules for single-match on source and destinations
- Up to 153 matching filters for multi-match on source and destinations because of wildcards

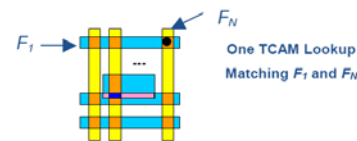
15

November 17, 2005 – CSE 7702

Amy M. Freestone

Separating Filters into Sets

- Need to keep intersections of filters in same set
- Don't need to store intersections of filters from different sets
- Matches on filters from different sets reported separately



Storage cost: N filters + $O(N^2)$ intersection
 Classification speed: 1 TCAM lookup time

Figure 5. Include all intersections in the TCAM.



Storage cost: N filters + 1 intersection (F_2 and F_3)
 Classification speed: 2 TCAM lookups time

Figure 6. Separate filters into two sets and perform TCAM lookups separately.
 Figure Source: Yu et al.

16

November 17, 2005 – CSE 7702

Amy M. Freestone

Separating Filters into Sets

- Various blocks of a TCAM can be searched in parallel.
- Filter sets split so that most intersections are between filters in different sets
- Only consider intersections which differ from original filters and therefore require storage

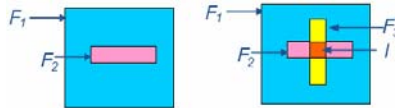


Figure 7. Example of filter intersections.
Figure Source: Yu et al.

17

November 17, 2005 – CSE 7702

Amy M. Freestone

Desired Algorithm

- Automatically separate filters into the smallest number of sets
- Keep the total number of filters within the TCAM's capacity

18

November 17, 2005 – CSE 7702

Amy M. Freestone

Mathematical Formulation

- Want to separate filters into the minimum number of sets satisfying $N + |I| < \text{TCAM size}$, where the residual intersection set I are intersections created by filters in the same set which differ from the filters in that set
 - * NP hard
- Divide filters into two sets where the number of residual intersections is a minimum
 - * Also NP hard (maximum set splitting or maximum hypergraph cut problem)

19

November 17, 2005 – CSE 7702

Amy M. Freestone

Johnson's Algorithm

- Maximum satisfiability algorithm upon which SSA is based
- L is a set of N literal pairs where each literal can have either a true or a false value
- M clauses, each with a subset of literals, each of which is either positive or negative
- Want to find an assignment of L that satisfies the most clauses
- K is minimum number of literals in each clause
 - * Problem known to be NP complete for $K \geq 2$

20

November 17, 2005 – CSE 7702

Amy M. Freestone

Johnson's Algorithm

```

Assign each clause with |C| literals a weight = 2|C|;
While (not all literals assigned weight yet){
Pick any remaining literal Fi;
  If the total weight of clauses containing Fi > those
  containing  $\overline{F}_i$  {
    Assign Fi a true value;
    Remove all clauses containing Fi;
    Double the weight of clauses containing  $\overline{F}_i$ ;
  } else {
    Assign Fi a false value;
    Remove all clauses containing  $\overline{F}_i$ ;
    Double the weight of clauses containing Fi;
  }
}

```

Figure 8. Johnson's algorithm.
Figure Source: Yu et al.

- Approximation algorithm for maximum satisfiability problem
- $O(NM)$ complexity
- Can satisfy at least $(2^K - 1)/2^K$ fraction of the total clauses
 - * $\frac{3}{4}$ when $K = 2$
- Best approximate bound for $K > 2$

21

November 17, 2005 – CSE 7702

Amy M. Freestone

Set Splitting Algorithm (SSA)

Reduce the filter set splitting problem into a max satisfiability problem:

Each filter F_i corresponds to a literal

For each intersection I_j generated by j filters: $F_{x1}, F_{x2}, \dots, F_{xj}$, add two clauses:

$$C = (F_{x1} \vee F_{x2} \vee \dots \vee F_{xj})$$

$$C' = (\overline{F}_{x1} \vee \overline{F}_{x2} \vee \dots \vee \overline{F}_{xj})$$

Run Johnson's algorithm to assign each filter F_i a true value or false value

Put F_i in set one if it is true.

Put F_i in set two if it is false.

Figure 9. Set splitting algorithm (SSA).
Figure Source: Yu et al.

- There are $2M$ clauses if there are M intersections.

22

November 17, 2005 – CSE 7702

Amy M. Freestone

Proof of SSA

- Lemma: If both clauses of an intersection are satisfied, this intersection is no longer needed in the TCAM. (Yu et al.)
- Proof: If both the clause $C = \{F_1 \vee F_2 \vee F_6\}$ and $C' = \{F_1' \vee F_2' \vee F_6'\}$, at least one of F_1, F_2, F_6 is true and at least one false. By the algorithm, they will be in different sets, meaning the intersection need not be included in the TCAM.

23

November 17, 2005 – CSE 7702

Amy M. Freestone

Proof of SSA

- Theorem: SSA can remove at least 50% of the intersections each time the filter set is split into two sets. (Yu et al.)
- Proof: Each clause has at least two literals because it was created by an intersection, making $K \geq 2$. At least $(2^K - 1)/2^K = 3/4$ of the $2M$ clauses are satisfied, which totals $1.5M$. With M intersections, each with 2 clauses, there must be at least $0.5M$ intersections with both clauses satisfied which, according to the lemma, can be removed.

24

November 17, 2005 – CSE 7702

Amy M. Freestone

SSA

- Filter set can be split additional times, decreasing the number of intersections by at least 50% each time
- Time complexity of $O(NM)$
- Literals in simulation chosen based on ratio of positive weight to negative weight, for a complexity of $O(NM + N^2)$

25

November 17, 2005 – CSE 7702

Amy M. Freestone

Simulation Results

- SSA compared with
 - * Existing TCAM approaches
 - MUD
 - Geometric Intersection-based
 - * Software-based solutions
 - EGT-PC
 - HiCuts
- Test sets
 - * SNORT rule header benchmark
 - * Synthesized larger filter sets

26

November 17, 2005 – CSE 7702

Amy M. Freestone

Evaluation Metrics

- *Memory consumption*: total number of TCAM entries
- *Speed*: worst case classification rate represented by maximum number of TCAM lookups per packet
- *Power consumption*: total TCAM entries accessed, the product of the number of TCAM entries accessed for each lookup and the number of lookups for each packet
- *Update costs*: number of newly inserted filters

27

November 17, 2005 – CSE 7702

Amy M. Freestone

Memory Consumption

Table 1. SNORT rule headers statistics.

Version	Release Date	Filter Set Size
2.0.0	4/14/2003	240
2.0.1	7/22/2003	255
2.1.0	12/18/2003	257
2.1.1	2/25/2004	263

Table 2. Total number of extra intersections filters in TCAMs.

Version	Geometric Intersection-based	SSA-2		SSA-4	
		Extra Intersections	Saving	Extra Intersections	Saving
2.0.0	3453	46	98.67%	1	99.97%
2.0.1	3754	47	98.75%	1	99.97%
2.1.0	3758	47	98.75%	0	100%
2.1.1	4067	55	98.65%	0	100%

Table 3. Total number of TCAM entries used.

Version	MUD	Geometric Intersection based		
		SSA-2	SSA-4	
2.0.0	240	3693	286	241
2.0.1	255	4009	302	256
2.1.0	257	4015	304	257
2.1.1	263	4330	318	263

Table Source: Yu et al.

- Geometric Intersection-based solution uses a filter set roughly 10 times that of the original
- MUD uses only as many TCAM entries as the number of original filters
- SSA's TCAM requirements are close to MUD's

28

November 17, 2005 – CSE 7702

Amy M. Freestone

Classification Speed

- Geometric Intersection-based: 1 lookup
- SSA: same number of lookups as filter sets
- MUD: at least 12 lookups
 - * Worst case is $1+d*(12-1)/r$, where d is 8 or 9 and r can be 5 or less, making the worst case as high as 20 lookups
 - * HTTP packet: at least 4 unique filters, 5-9 lookups
 - * Napster packet: 8 unique filters, 9-15 lookups

29

November 17, 2005 – CSE 7702

Amy M. Freestone

Update Cost

Table 4. Update cost in terms of newly inserted filters.

Version	MUD	Geometric Intersection-based		SSA-2		SSA-4	
		Avg	Max	Avg	Max	Avg	Max
2.0.0	1	31.73	157	1.33	17	1.002	2
2.0.1	1	35.24	135	1.34	19	1	1
2.1.0	1	34.71	135	1.36	20	1.002	2
2.1.1	1	36.00	172	1.41	26	1.006	2

Table Source: Yu et al.

- High cost for Geometric Intersection-based because of potential interactions with existing filters
- SSA-2 can have a high max update cost at 20
- SSA-4 has a worst case of 2, nearly as good as MUD

30

November 17, 2005 – CSE 7702

Amy M. Freestone

Power Consumption

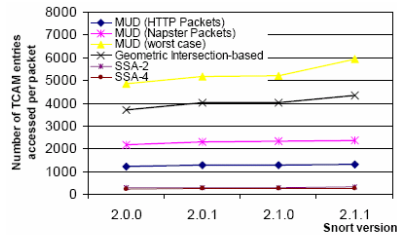


Figure 10. TCAM entries accessed per packet.
Figure Source: Yu et al.

- TCAM accesses for MUD worst case can number 20 times more than the filter set size
- Geometric Intersection-based only makes one access per packet, but has more filters because of the intersections
- SSA makes one access per packet with few additional intersections necessary

31

November 17, 2005 – CSE 7702

Amy M. Freestone

Synthesized Multi-Match Filter Set

Table 5. Total number of extra intersections filters in TCAMs.

Insertion Factor	Geometric Intersection-based	SSA-2		SSA-4	
		Intersections	Saving	Intersections	Saving
0	359	22	93.87%	2	99.44%
0.05	418	20	95.22%	1	99.76%
0.1	488	45	90.78%	3	99.39%
0.2	733	52	92.91%	4	99.45%
0.3	1080	112	89.63%	1	99.91%
0.4	1312	78	94.05%	9	99.31%
0.6	2086	171	91.80%	9	99.57%
0.8	2488	208	91.64%	4	99.84%
1	2883	229	92.06%	7	99.76%

Table Source: Yu et al.

- Test algorithms with larger filter set size (3060) and with varying intersection rates
- Speed and energy results similar to with SNORT database
- SSA-2 faster and more energy efficient than MUD

32

November 17, 2005 – CSE 7702

Amy M. Freestone

Software Solutions

- EGT-PC directly supports multi-match
- Slight modifications made to HiCuts to report all results
- EGT-PC may have to go compare 153 filters individually to a packet
- High number of filter intersections causes HiCuts to copy filters to multiple leaf nodes (3108 times average)
- Excessive duplication causes an SRAM storage requirement larger than the largest single chip SRAM density currently available (~8 MB) and will cause large update causes

Table 6. Applying HiCuts to the SNORT rule set.

Version	Tree Height	Number of Filters in Leaf Nodes	SRAM Used (KB)
2.0.0	18	745,019	41,000
2.0.1	19	803,645	46,297
2.1.0	19	820,415	47,160
2.1.1	18	827,651	49,378

Table Source: Yu et al.

33

November 17, 2005 – CSE 7702

Amy M. Freestone

Conclusions

- Previous TCAM-based approaches to multi-match packet classification had high power consumption of high memory usage.
- When tested with the SNORT rule set, SSA makes a 90% reduction in memory and power consumption.
- SSA has deterministic lookup rates and fast updates.

34

November 17, 2005 – CSE 7702

Amy M. Freestone