Packet Classification: From Theory to Practice

Jun Li

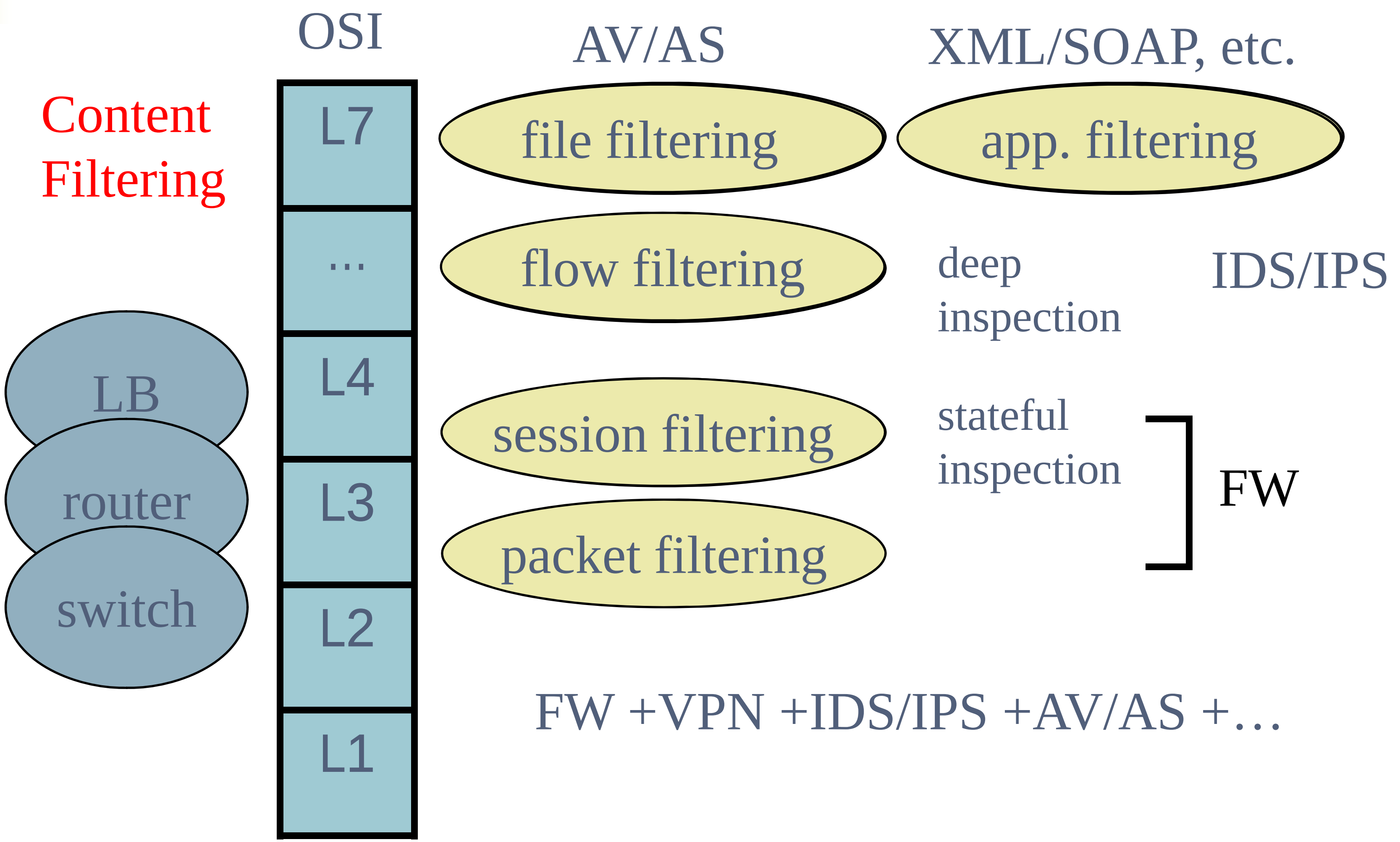
Most contributions from Yaxuan Qi and many other students of mine

Tsinghua Univ., Beijing, China

Outline

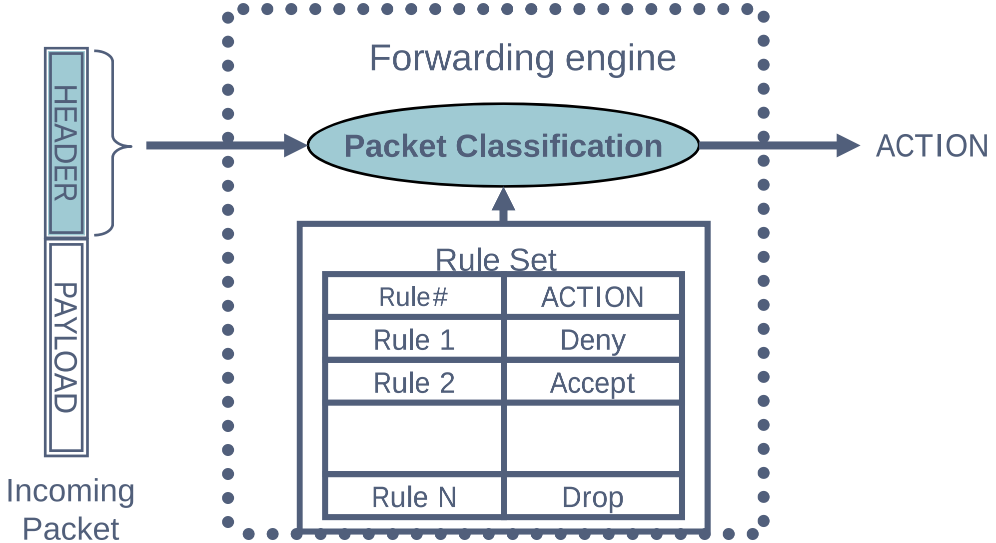
- Packet Classification Introduction
- Review of Classic Algorithms
- Our Recent Advancement
- Conclusion and Discussion

Network Security Gateway



The Packet Classification Problem

- Definition:
  - Given *N* rules, find the action associated with the highest priority rule matching an incoming packet.
- Applications:
  - Access control
  - Quality of service
  - Traffic engineering
  - Intrusion detection
  - …



	Field 1 (sIP)	Field 2 (dPort)	...	Field F (protocol)	Action
Rule 1	166.111.72.50/21	80	...	UDP	Deny
Rule 2	166.168.3.0/24	53	...	TCP	Accept
...	...	...	...	...	...
Rule N	0.0.0.0/0	0~65535	...	ANY	Drop

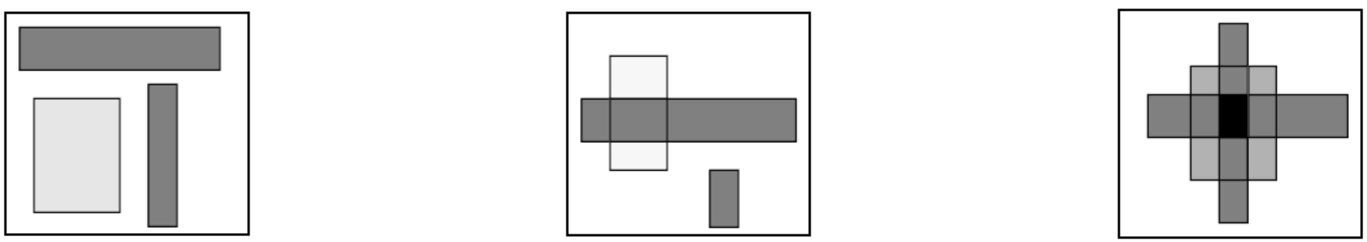
Problem Definition I

- Given a classifier C with N rules, R_j, $1 \leq j \leq N$, where R_j consists of three entities:
  - **Range expressions:** $R_j[i]$, $1 \leq i \leq d$, on each of the d header fields.
  - **Priority:** pri(R_j), indicating the priority of the rule in the classifier. Commonly, 1^{st} policy has the highest priority, N^{th} policy (normally deny all) has the lowest.
  - **An action:** referred to as action(R_j). In firewall, usually there is a default policy as the last policy that matches and denies all.

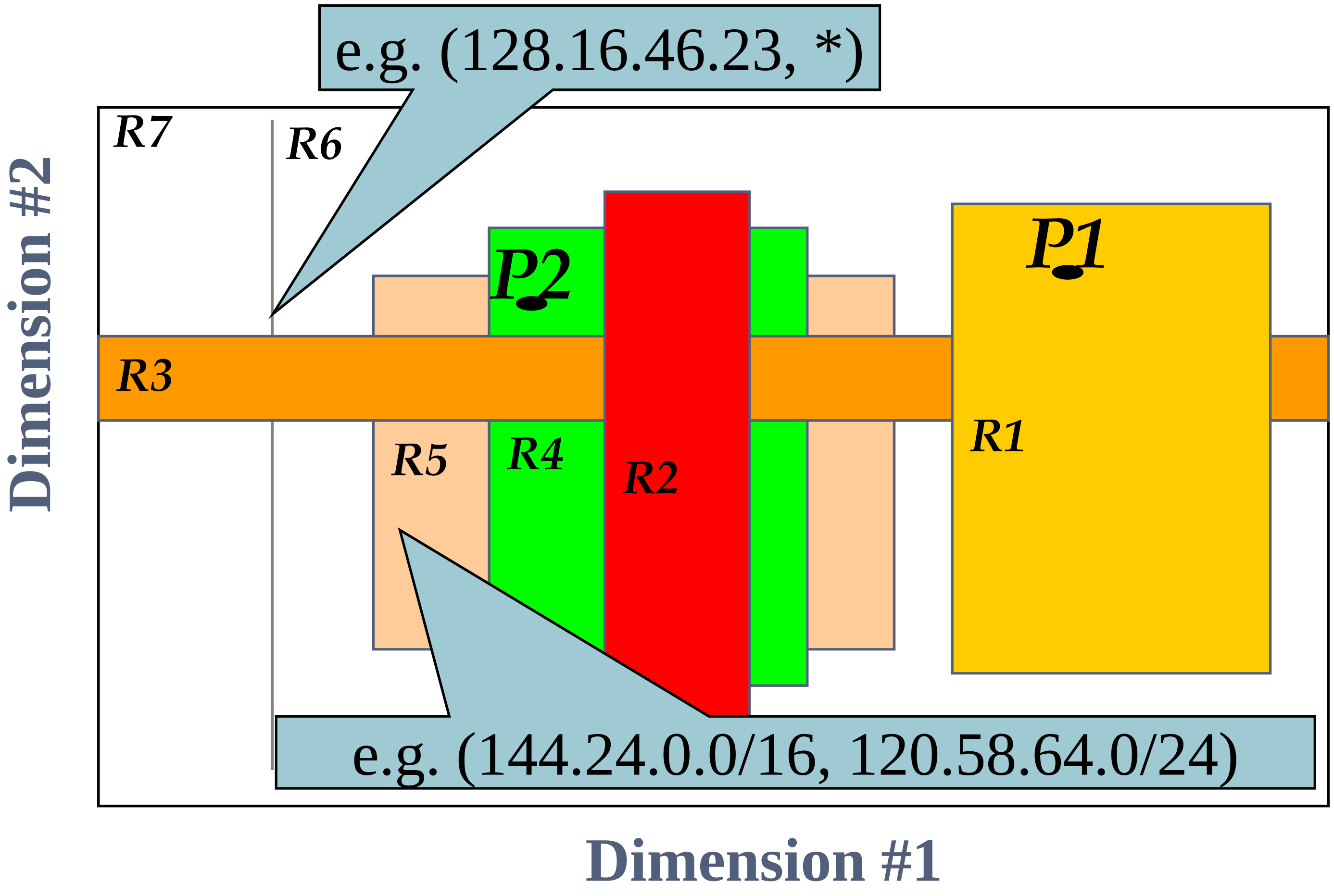
Problem Definition II

- For an incoming packet P with the header considered as a d-tuple of points (P_1, P_2, …, P_d), the *d-dimensional packet classification problem* is to find the rule R_m with the highest priority among all the N rules that match the d-tuple
  - i.e., pri(R_m) > pri(R_j), $\forall j \neq m, 1 \leq j \leq N$, such that P_i matches $R_j[i]$, $1 \leq i \leq d$. In firewall, this is the first matching policy.
  - R_m is called the best matching rule for packet P, therefore action(R_m) is applied to packet P.

Problem Complexity: Theoretically

- Computational Geometry
  - Point Location among N non-overlapping hyper-rectangles in F dimensions
  - Takes either $O(\log N)$ time with $O(N^F)$ space or $O(N)$ space with $O(\log^{F-1} N)$ time
  - E.g. N=1000, F=4: 1000G memory or 1000 times access
  - 
- De-overlapping
  - Each field need up to (2N-1) non-overlapping regions to represent N rules. How about F fields?
- Range-to-Prefix
  - Each rule with ranges in [0, 2^W-1] becomes up to $(2W-2)^F$ rules. How about N rules?

Geometric Interpretation in 2D



Problem Complexity: Practically

TABLE I. THEORETICAL VS. PRACTICAL COMPLEXITY FOR REAL-LIFE RULES

Rule Sets	*# rules*	*# non-over ranges in each field (theoretical)*	*# non-over ranges in sIP (practical)*	*# non-over ranges in dIP (practical)*	*# non-over ranges in sPT (practical)*	*# non-over ranges in dPT (practical)*	*# non-over rectangles (theoretical)*	*# non-over rectangles (practical)*
FW1	269	539	100	111	23	77	8.44×10^{10}	1.97×10^{7}
FW1-100	92	185	19	45	20	48	1.17×10^{9}	8.21×10^{5}
FW1-1K	791	1583	221	314	23	75	6.28×10^{12}	1.20×10^{8}
FW1-5K	4653	9307	3429	5251	23	77	7.50×10^{15}	3.19×10^{10}
FW1-10K	9311	18623	[illegible]	[illegible]	[illegible]	[illegible]	1.20×10^{17}	1.71×10^{11}
ACL1	752	1505	[illegible]	[illegible]	[illegible]	[illegible]	5.13×10^{12}	9.67×10^{6}
ACL1-100	98	197	[illegible]	[illegible]	[illegible]	[illegible]	1.51×10^{9}	4.03×10^{5}
ACL1-1K	916	1833	[illegible]	[illegible]	[illegible]	[illegible]	1.13×10^{13}	1.32×10^{7}
ACL1-5K	4415	8831	[illegible]	[illegible]	[illegible]	[illegible]	6.08×10^{15}	2.42×10^{8}
ACL1-10K	9603	19207	[illegible]	[illegible]	[illegible]	[illegible]	1.36×10^{17}	1.90×10^{9}
IPC1	1550	3101	[illegible]	[illegible]	[illegible]	[illegible]	9.25×10^{13}	3.40×10^{8}
IPC1-100	99	199	[illegible]	[illegible]	[illegible]	[illegible]	1.57×10^{9}	9.49×10^{6}
IPC1-1K	938	1877	[illegible]	[illegible]	[illegible]	[illegible]	1.24×10^{13}	1.70×10^{9}
IPC1-5K	4460	8921	886	2125	59	93	6.33×10^{15}	1.03×10^{10}
IPC1-10K	9037	18075	2377	4604	59	94	1.07×10^{17}	6.07×10^{10}

Note: sIP, dIP, sPT and dPT are source IP, destination IP, source Port and destination Port; FW, ACL, IPC are firewall policies, access control lists, and IP chain rules

worst-case: $5.13\text{x}10^{12}$
practical: $9.67\text{x}10^{6}$

- Fortunately
  - Few application reaches the worst case bound
  - Real-life rule sets have some inherent data-structures

Outline

- Packet Classification Introduction
- Review of Classic Algorithms
- Our Recent Advancement
- Conclusion and Discussion

Existing Work: Basic Ideas

- Divide-and-Conquer
  - **Space Decomposition**
    - Decompose the search space into multiple sub-spaces using a set of axis-orthogonal hyper-planes. Each sub-space is associated with a sub-set of rules.
  - **Recursion Scheme**
    - Recursively apply the space decomposition, the original problem is divided into a series of sub-problems with smaller search space and fewer rules.

Example Rule Set

2-field rules with overlap

R1 has the highest priority

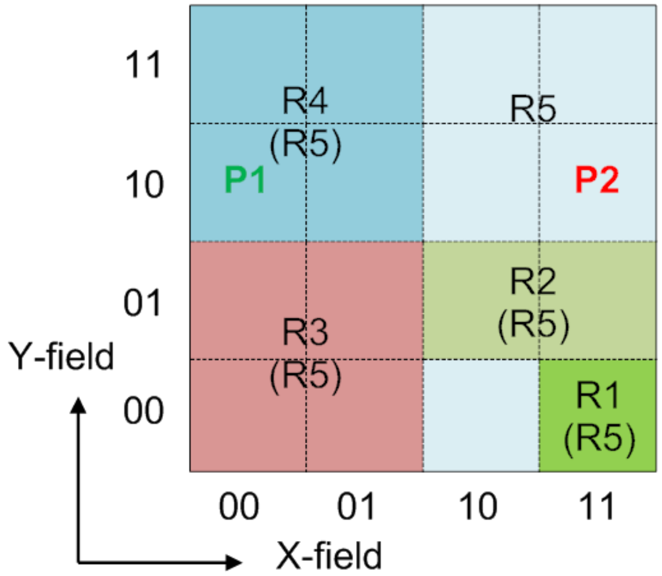
R5 is the default rule

2 Points

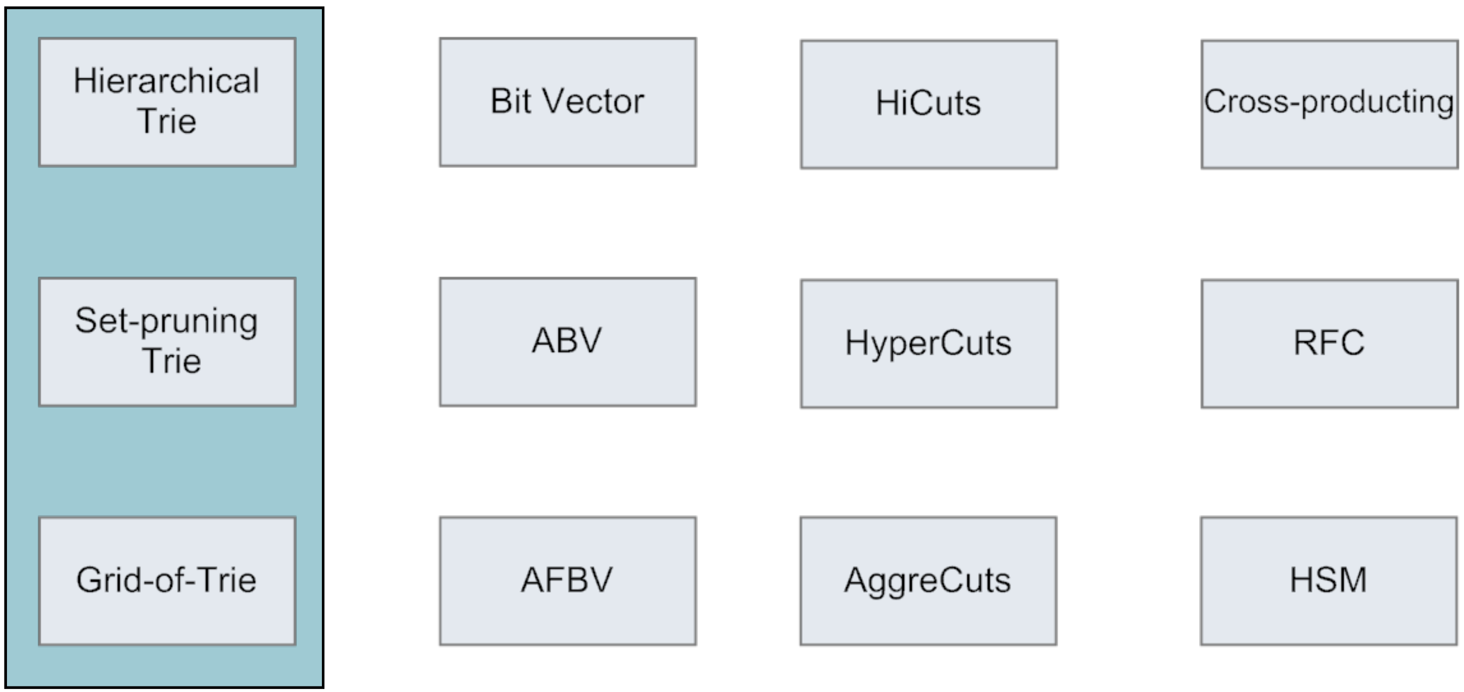
P1: (00, 10) → R4

P2: (11, 10) → R5

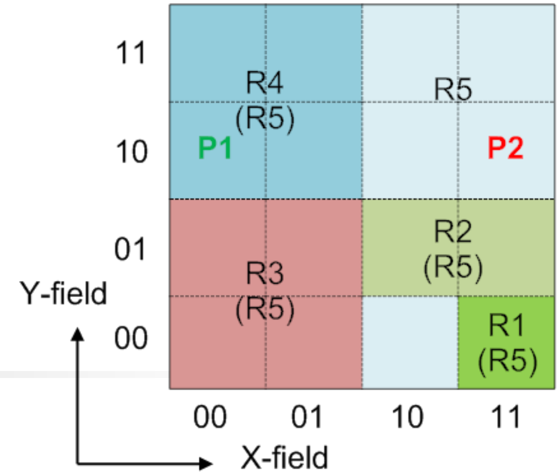
	X-field	Y-field
R1	11	00
R2	1*	01
R3	0*	0*
R4	0*	1*
R5	*	*

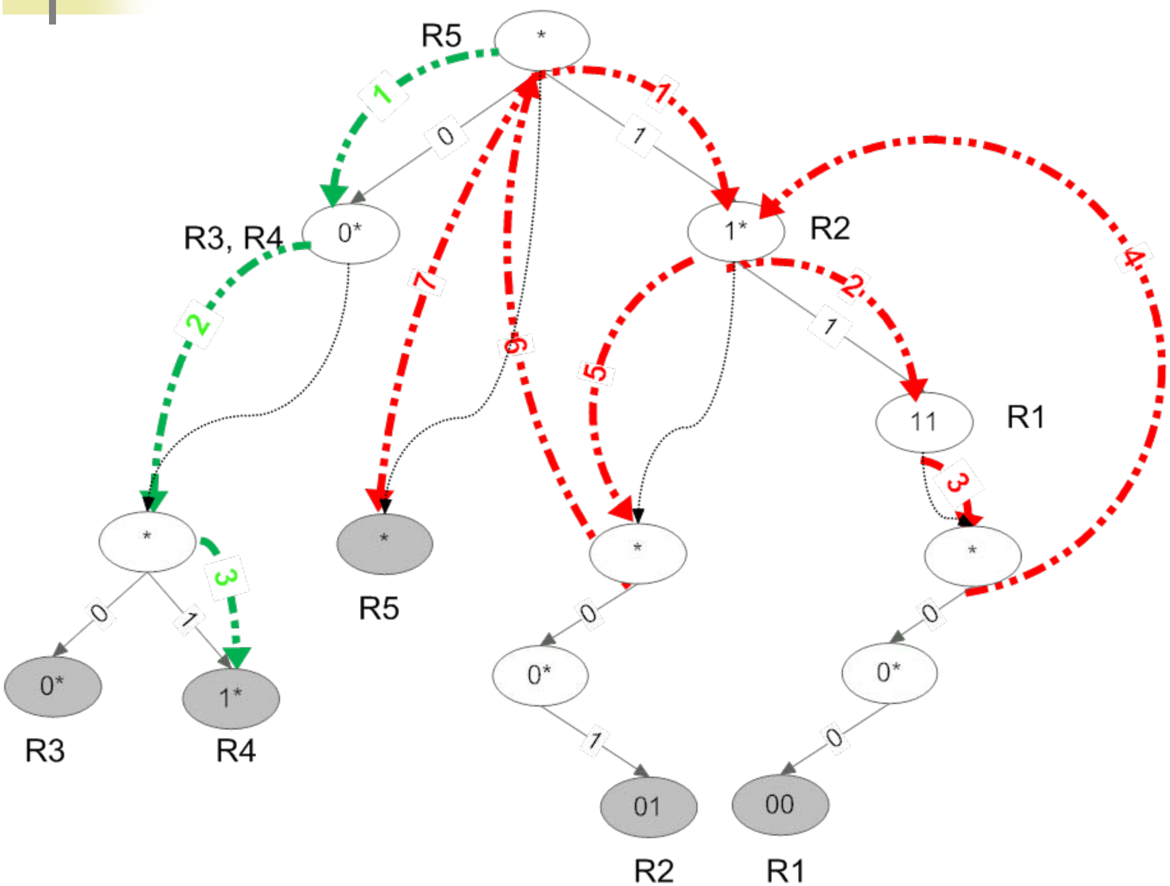


Packet Classification Algorithms



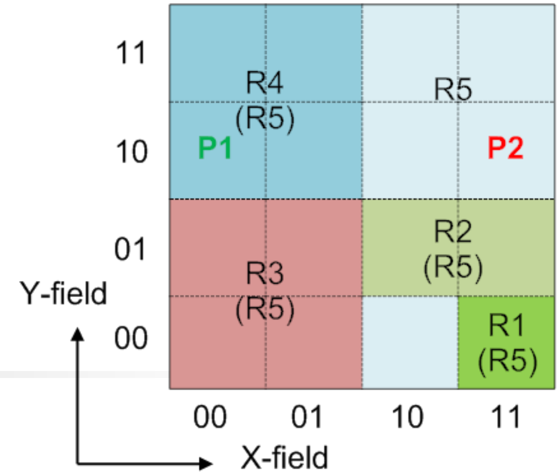
Hierarchical Tries

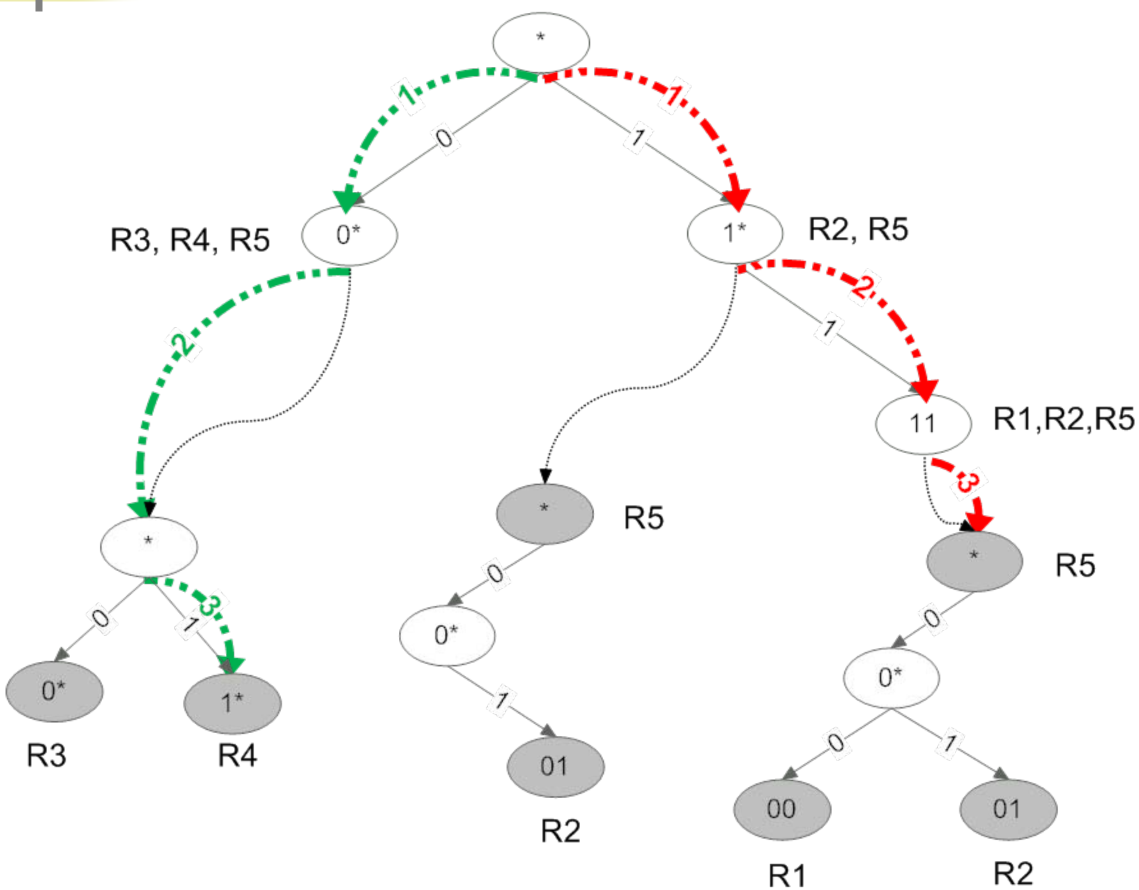




- Back-tracking search: O(Wd) time
- Each rule appears only once: O(N) space

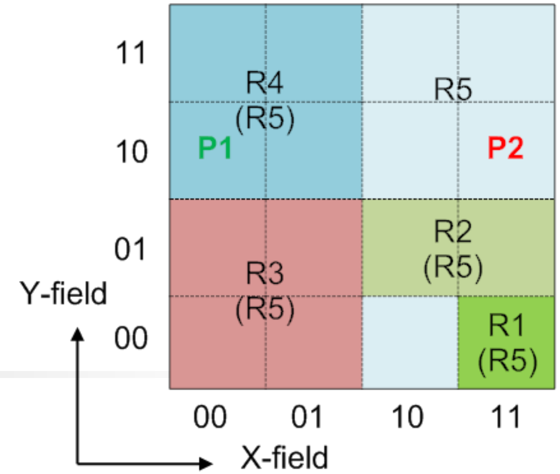
Set-pruning Tries

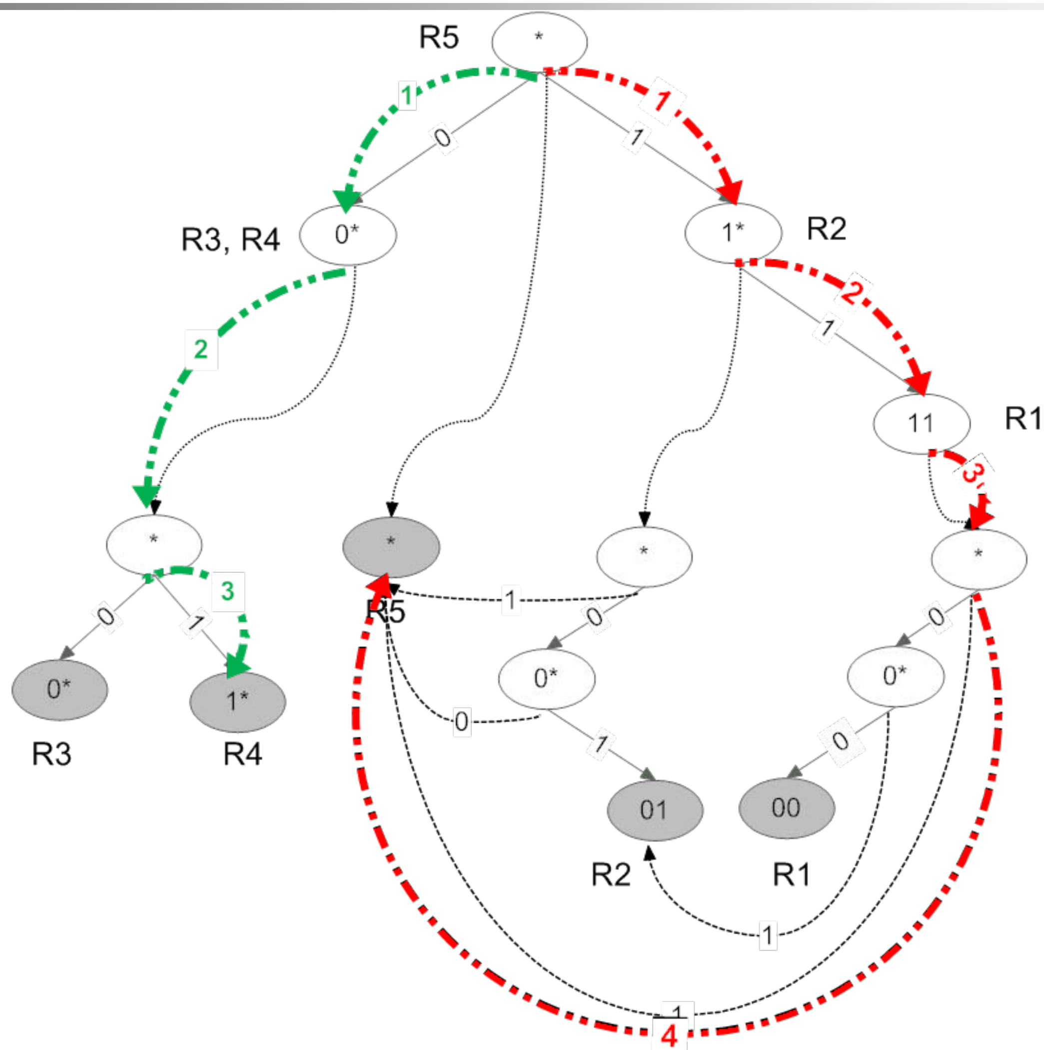




- No back-tracking search: O(dW) time
- Rules may appear multiple times: O(Nd) space

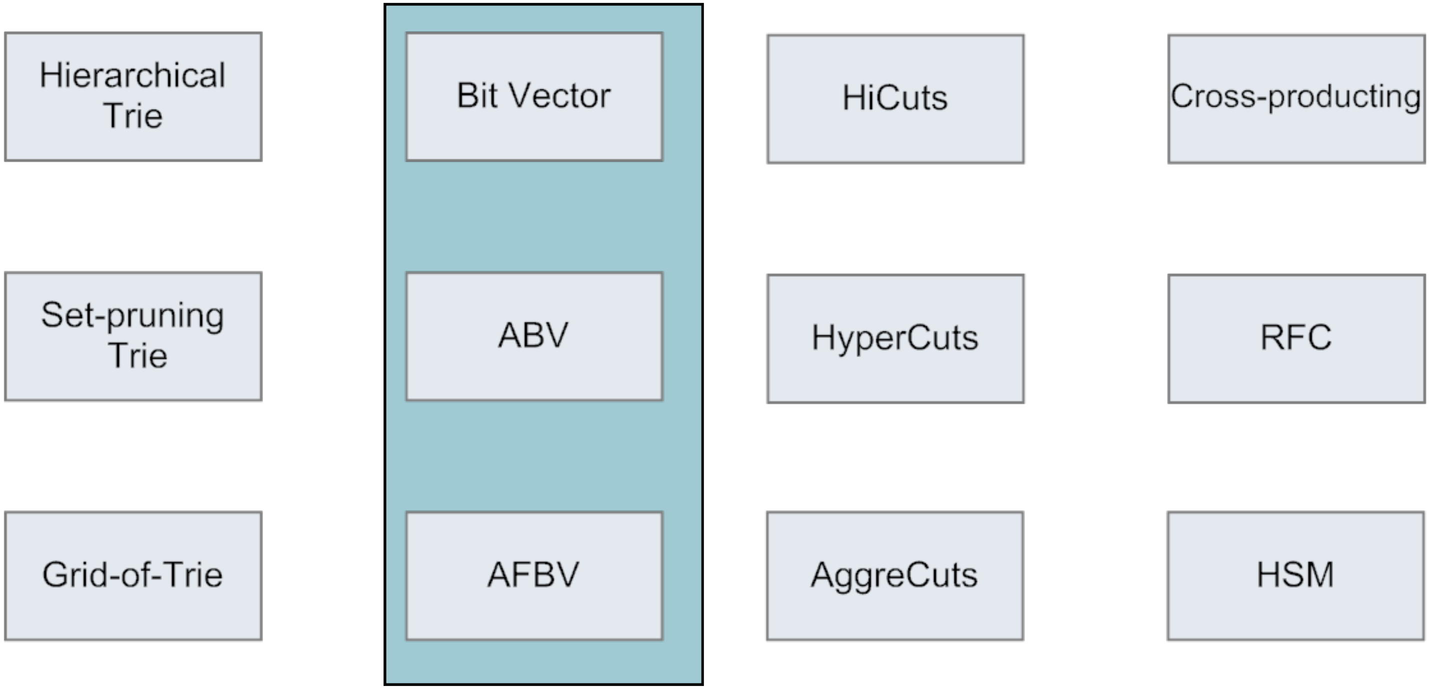
Grid of Tries



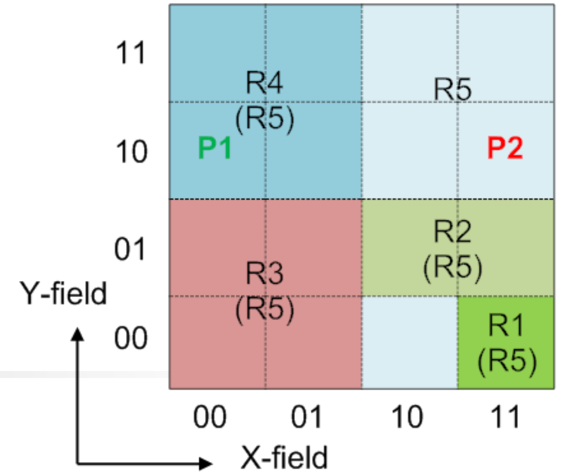


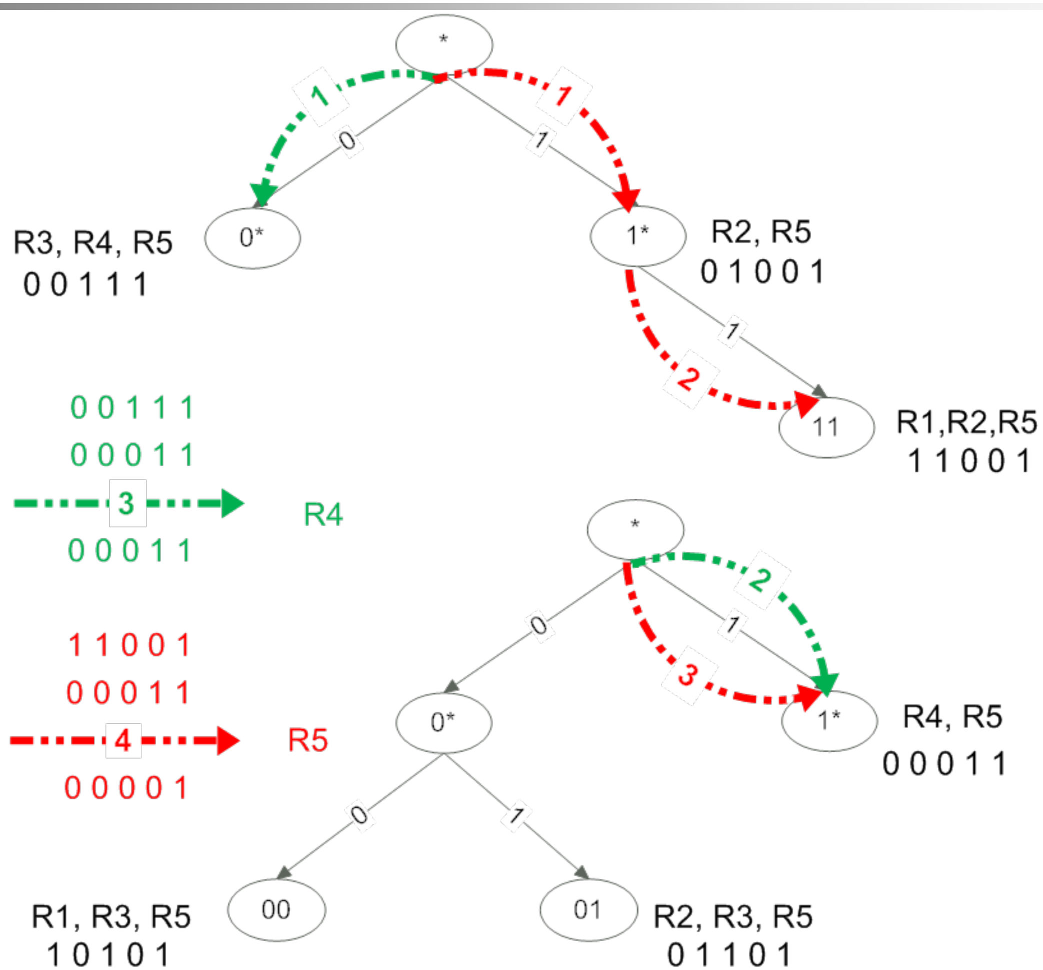
- No back-tracking search: $O(dW)$ time
- Rules appear only once: $O(N)$ space
- Only work for 2-D scheme

Packet Classification Algorithms



Bit Vector



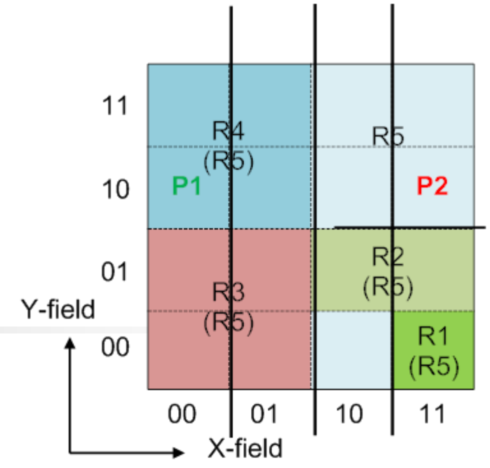


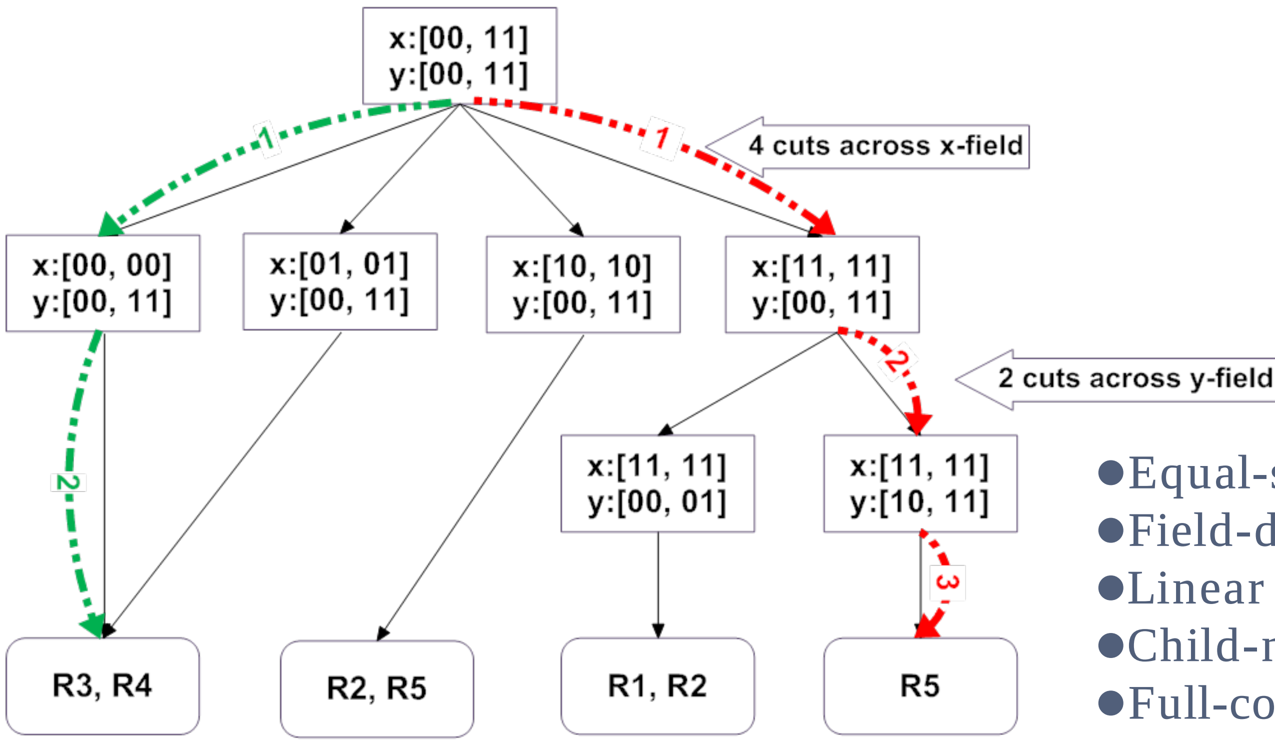
- Field-independent search: $O(dW+N)$ time
- Bitmap Storage: $O(dN^2)$ space
- Bitmap comparison is time-consuming

Packet Classification Algorithms

Hierarchical Trie	Bit Vector	HiCuts	Cross-producting
Set-pruning Trie	ABV	HyperCuts	RFC
Grid-of-Trie	AFBV	AggreCuts	HSM

HiCuts

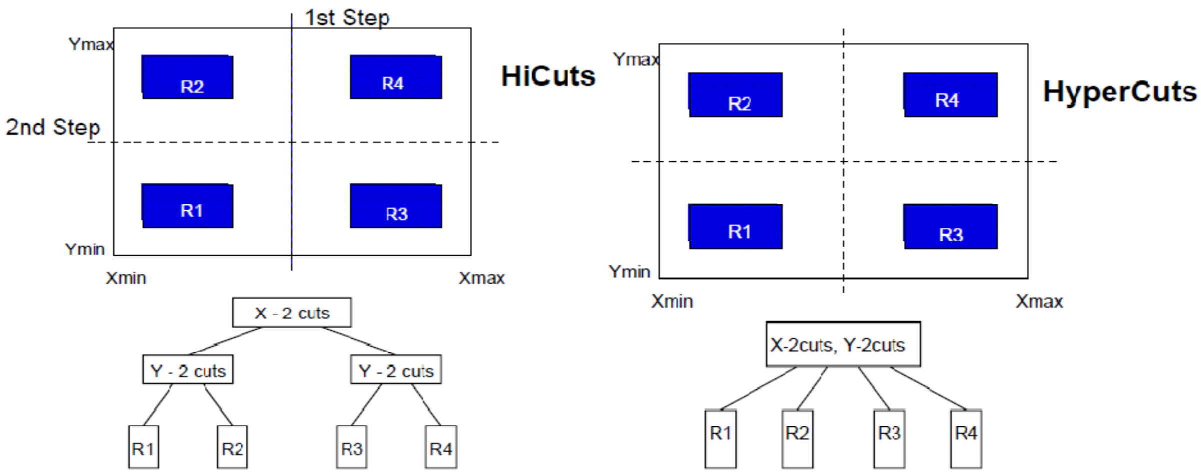




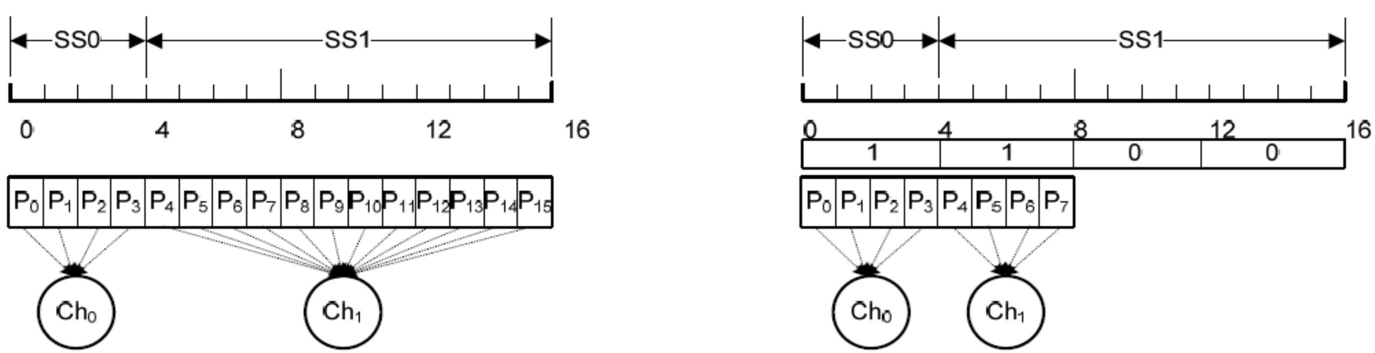
- Equal-sized space partition
- Field-dependent search
- Linear search at leaf-node
- Child-node merge
- Full-covered region check

HyperCuts and AggreCuts

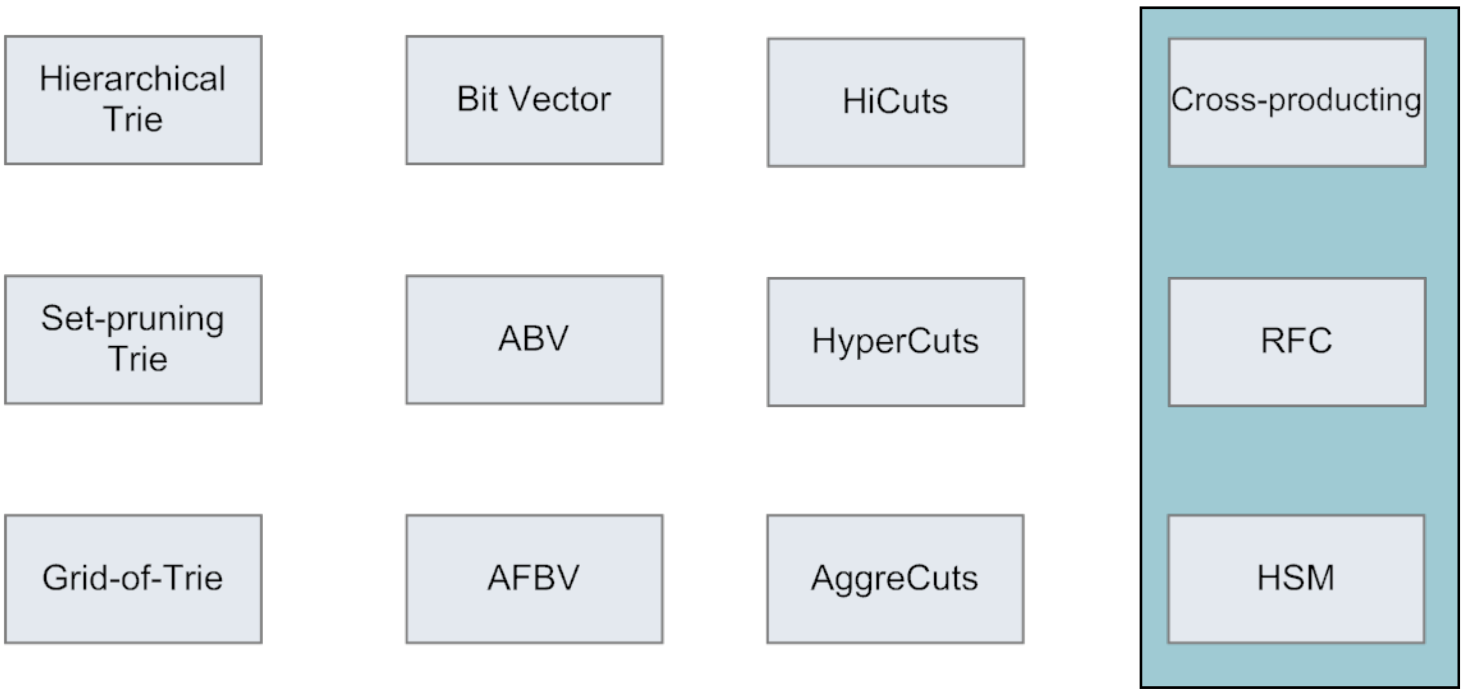
HyperCuts: multi-dimensional cuttings



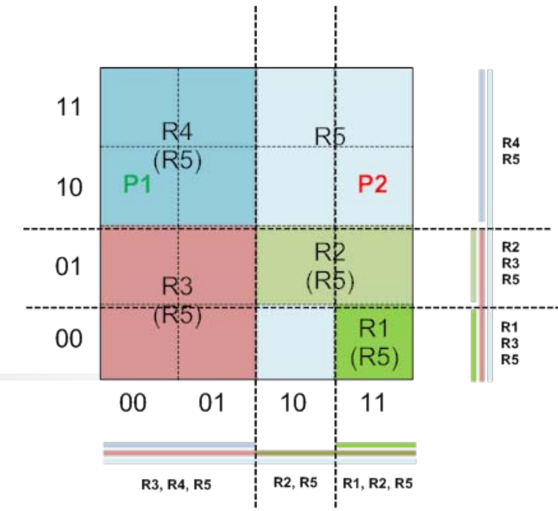
AggreCuts: pointer array compression

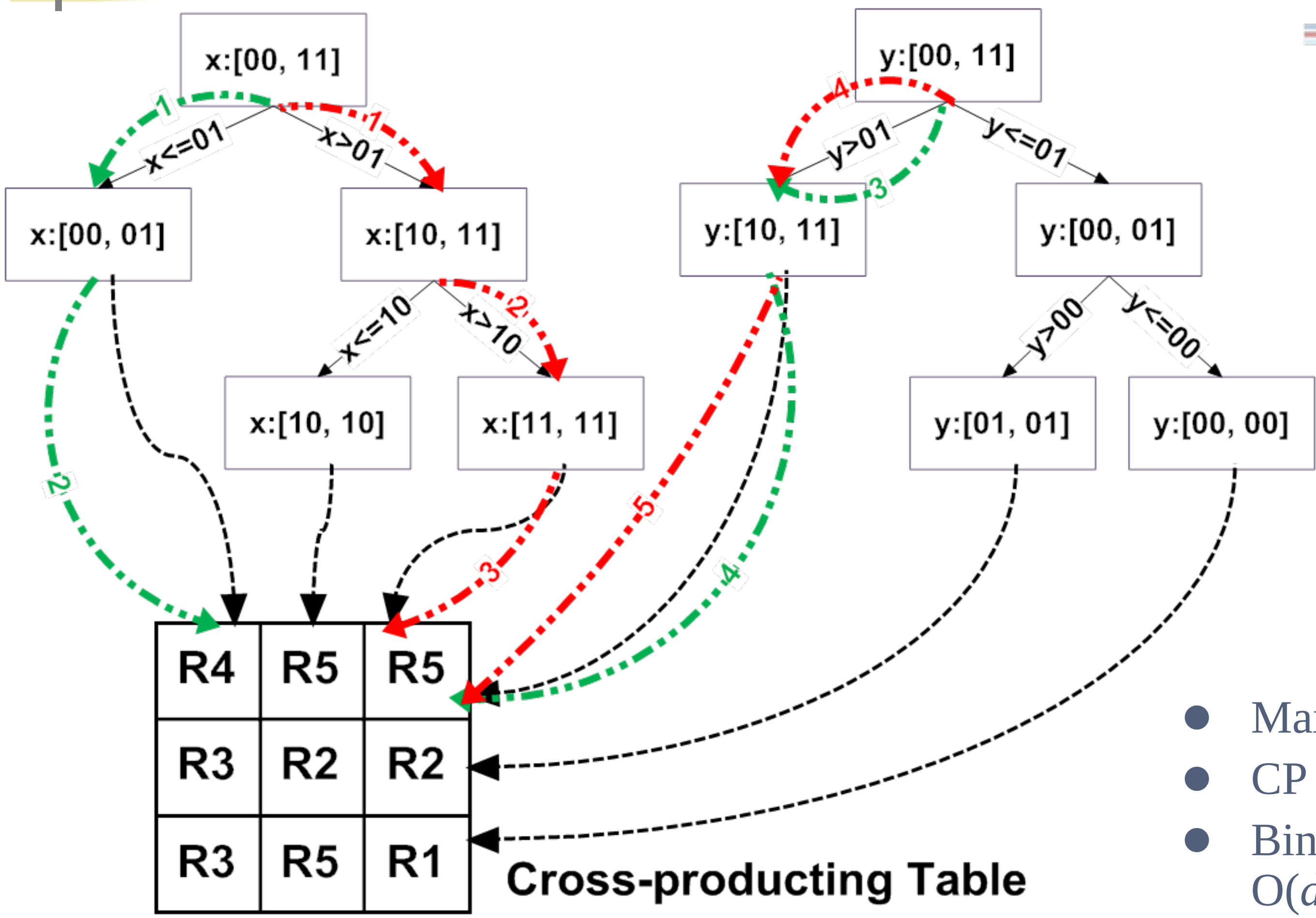


Packet Classification Algorithms



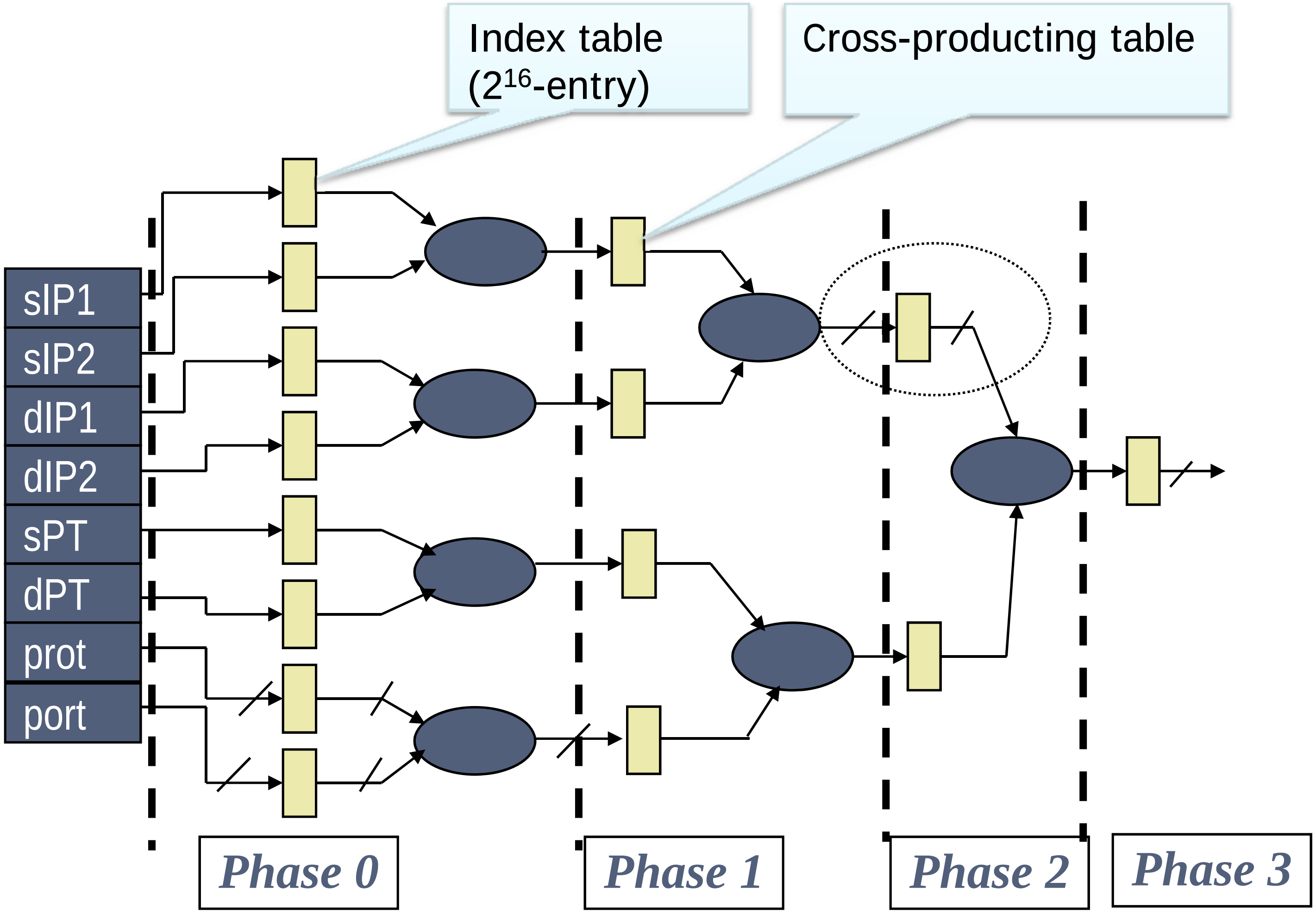
Cross-producting





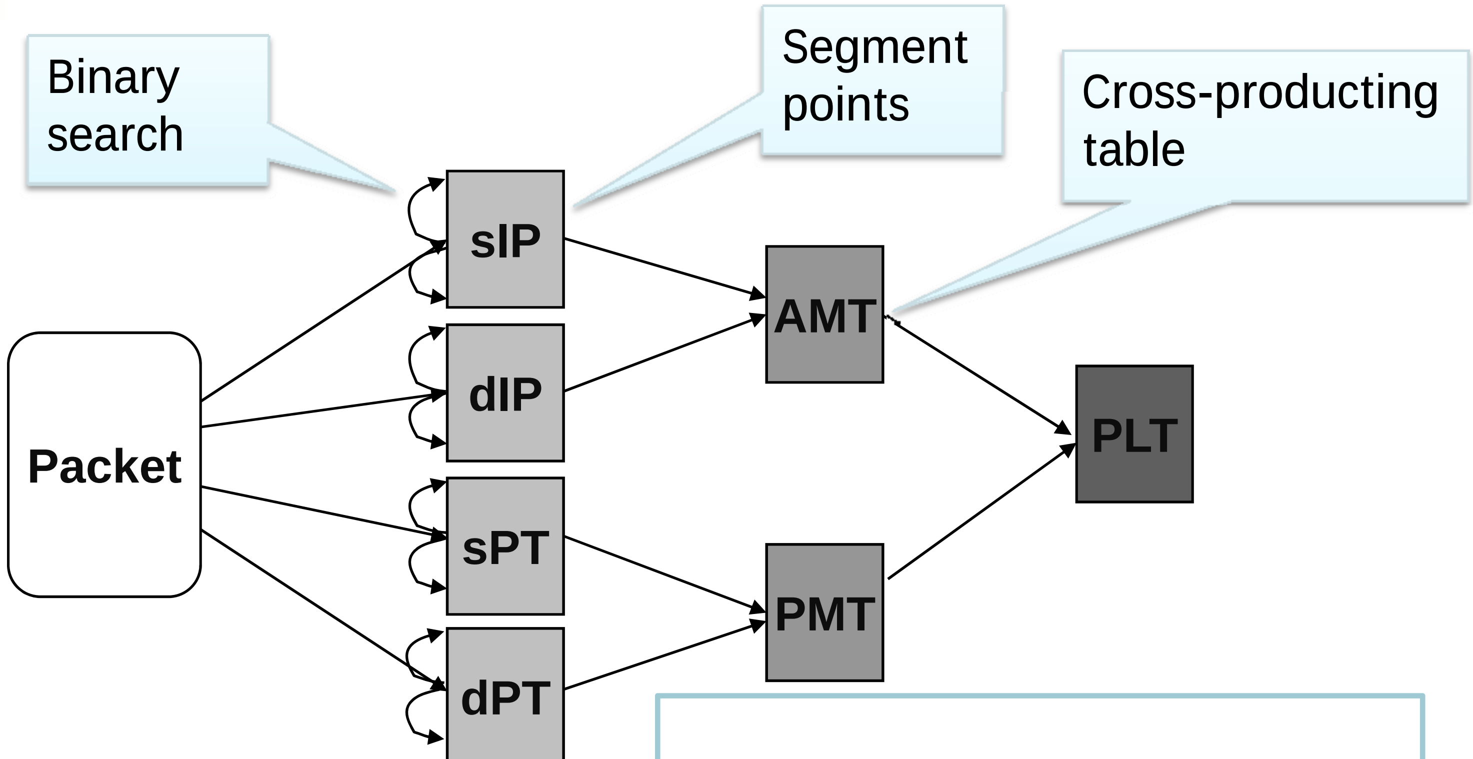
- Max#segments: $2N+1$
- CP table size: $O(N^d)$
- Binary search: $O(d*\log(N))$

RFC



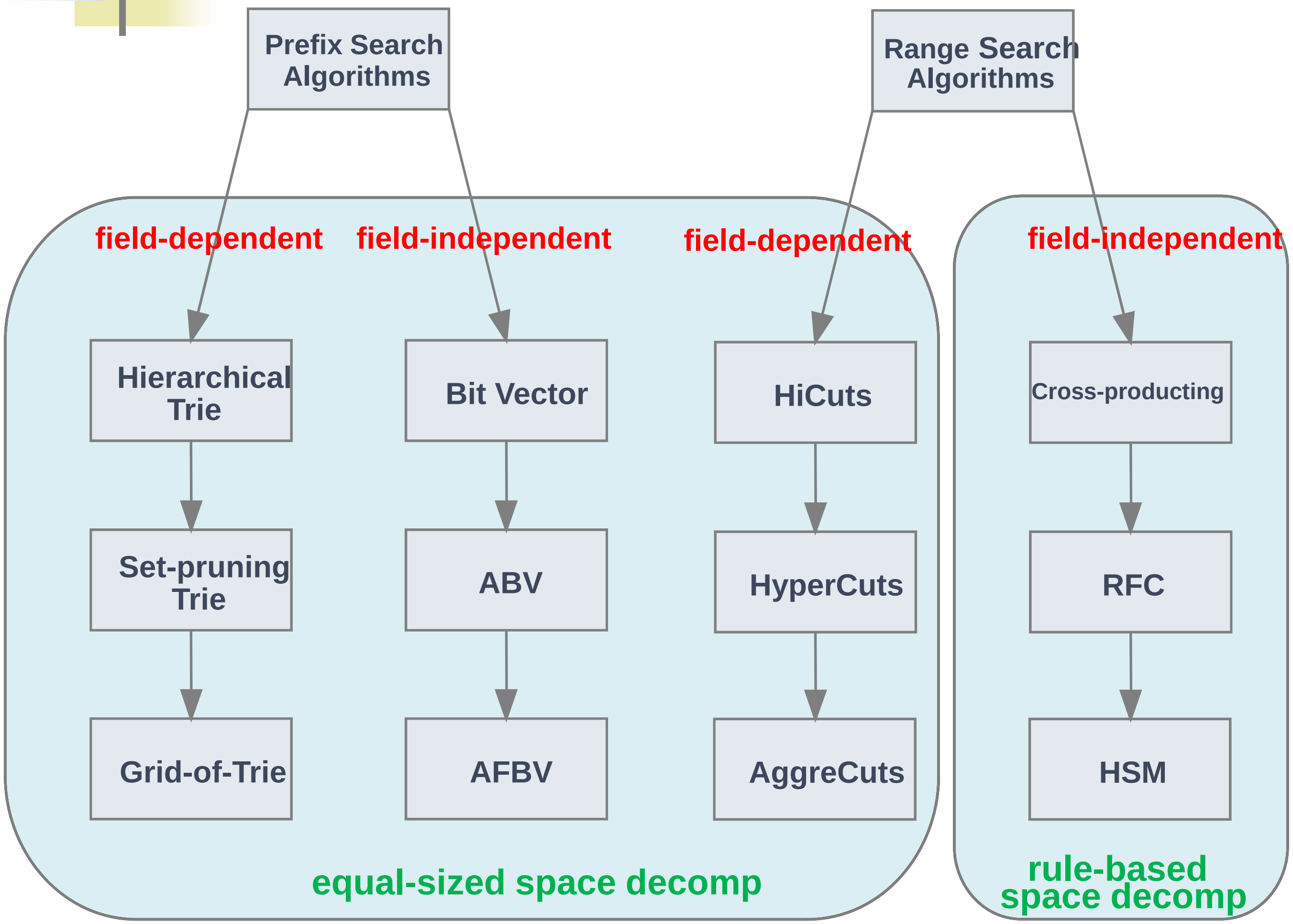
Index table
(2^{16}-entry)
Cross-producting table
sIP1
sIP2
dIP1
dIP2
sPT
dPT
prot
port
Phase 0
Phase 1
Phase 2
Phase 3

HSM



AMT: Address Mapping Table
PMT: Port Mapping Table
PLT: Policy Lookup Table

Summary: Taxonomy



Decomp \ Recur	rule-based	equal-sized
field-independent	✓ HSM	✓ BV
field-dependent	?	✓ HiCuts

Our recent advancement

Outline

- Packet Classification Introduction
- Review of Classic Algorithms
- Our Recent Advancement
- Conclusion and Discussion

HyperSplit: Motivation

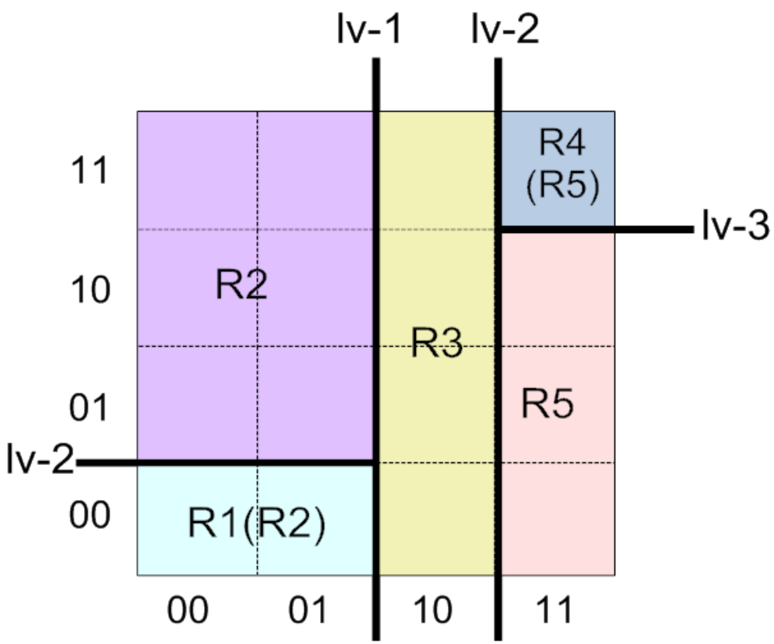
- High-speed classification
  - Speed is the most important performance metric
  - Speed should be bounded in the worst case
- Modest memory storage
  - Memory storage cannot exceed the overall system memory size
  - Modest memory storage enables the use of fast memory technology

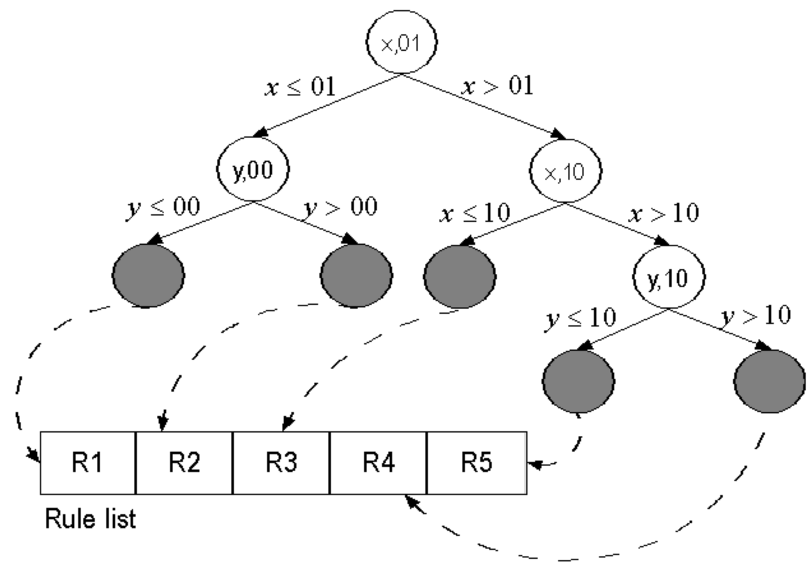
HyperSplit: Ideas

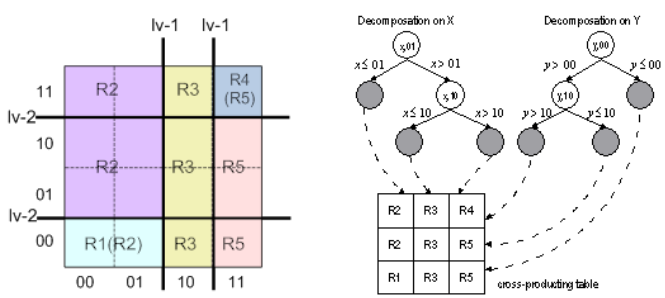
- Rule-based space decomposition
  - Binary splitting: **O (log(N))** search time
  - Intelligently select the splitting point
    - Heuristic-1: select the mid-value point
    - Heuristic-2: select the mid-segment point
    - Heuristic-3: select the weighted mid-segment point
- Field-dependent recursion scheme
  - Always select the most discriminative field to apply decomposition
    - For Heuristic-1 and Heuristic-2: select the field with the largest number of segments
    - For Heuristic-3: select the field with minimized average weight
  - Termination conditions
    - There're less than ***Thresh*** rules in current search space
    - The current search space is fully covered by all the current rules

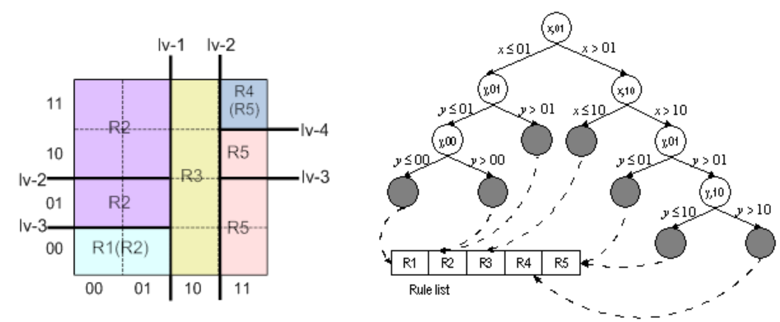
Data structure design and optimization also important, see paper.

HyperSplit: Example





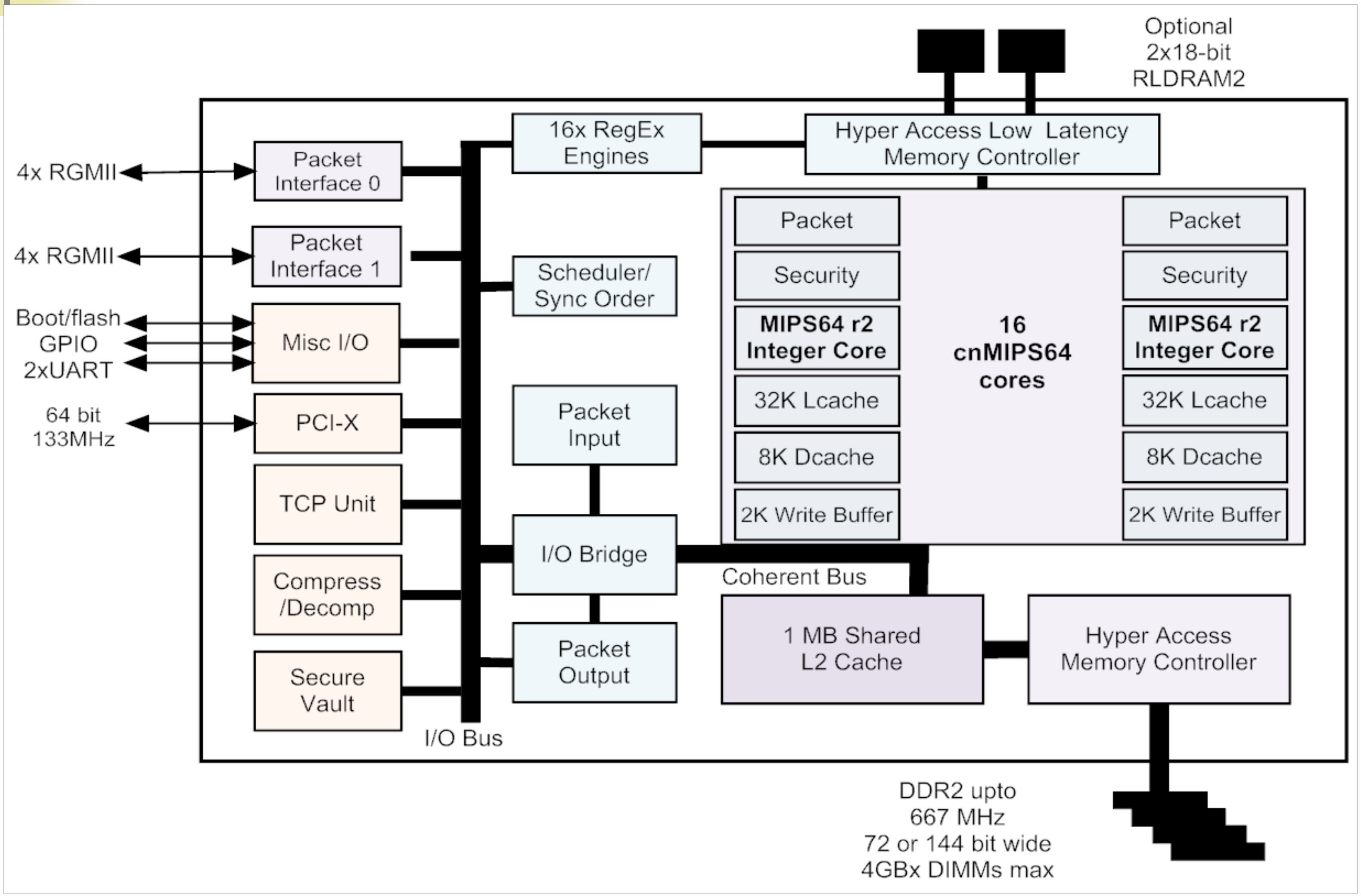




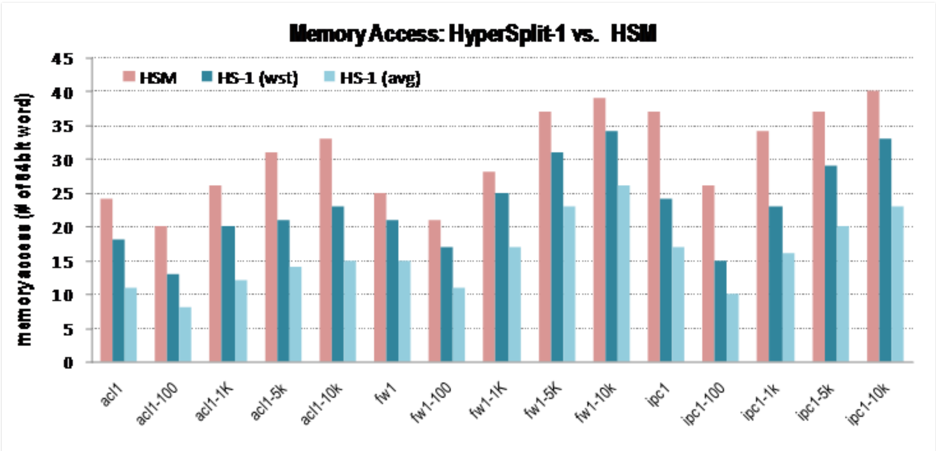
Data-set and Test-bed

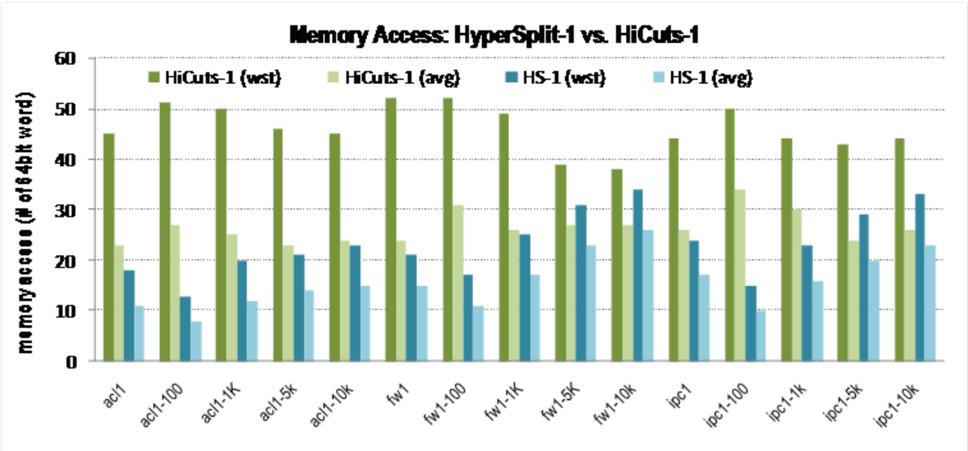
- Algorithms
  - HyperSplit-1, HyperSplit-8, HiCuts-1, HiCuts-8 and HSM
- Data-set
  - WUSTL Evaluation of Packet Classification Algorithms
    - 100~10K real-life 5-tuple firewall, ACL and IP Chain rules
    - http://www.arl.wustl.edu/~hs1/PClassEval.html
- Test-bed
  - Memory access, usage, and preprocessing time: 2.0GHz dual-core with 4GB DDRII memory runing Ubuntu 8.04LTS
  - Throughput: Cavium OCTEON 3860 multi-core processor runing in “Simple Executive” mode
  - SmartBit packet generator

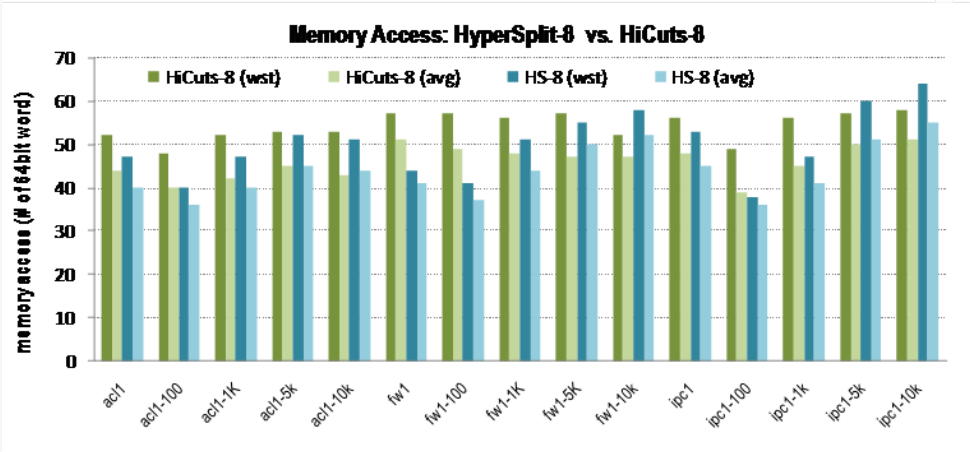
Cavium OCTEON 3860



Memory Access

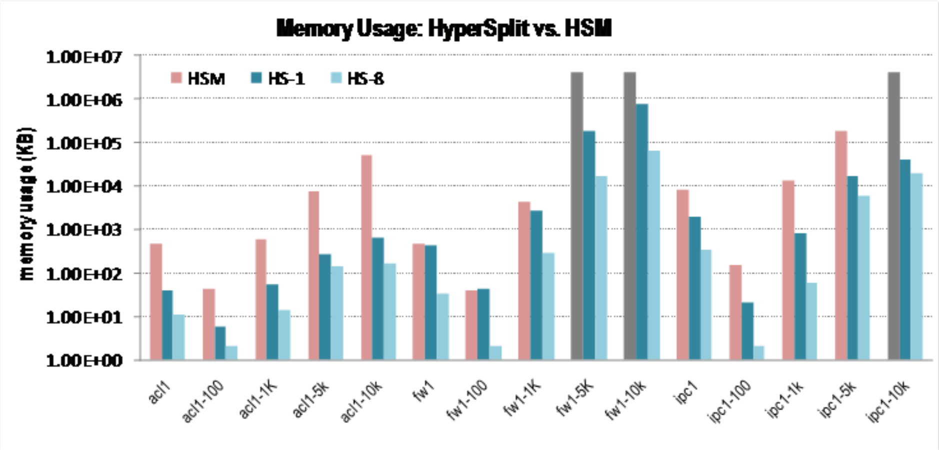


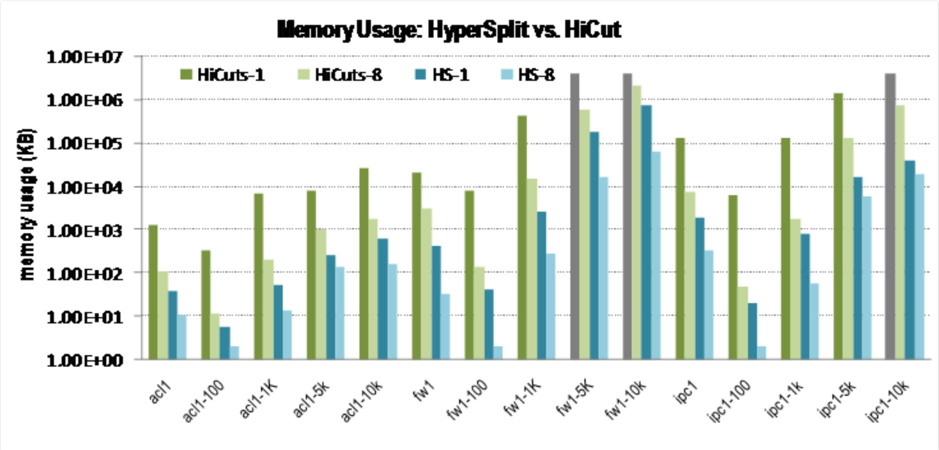




- HyperSplit-1 vs. HSM
  - 20~50% less access
- HyperSplit-1 vs. HiCuts-1
  - 50~80% less access
- HyperSplit-8 vs. HiCuts-8
  - 10~30% less access

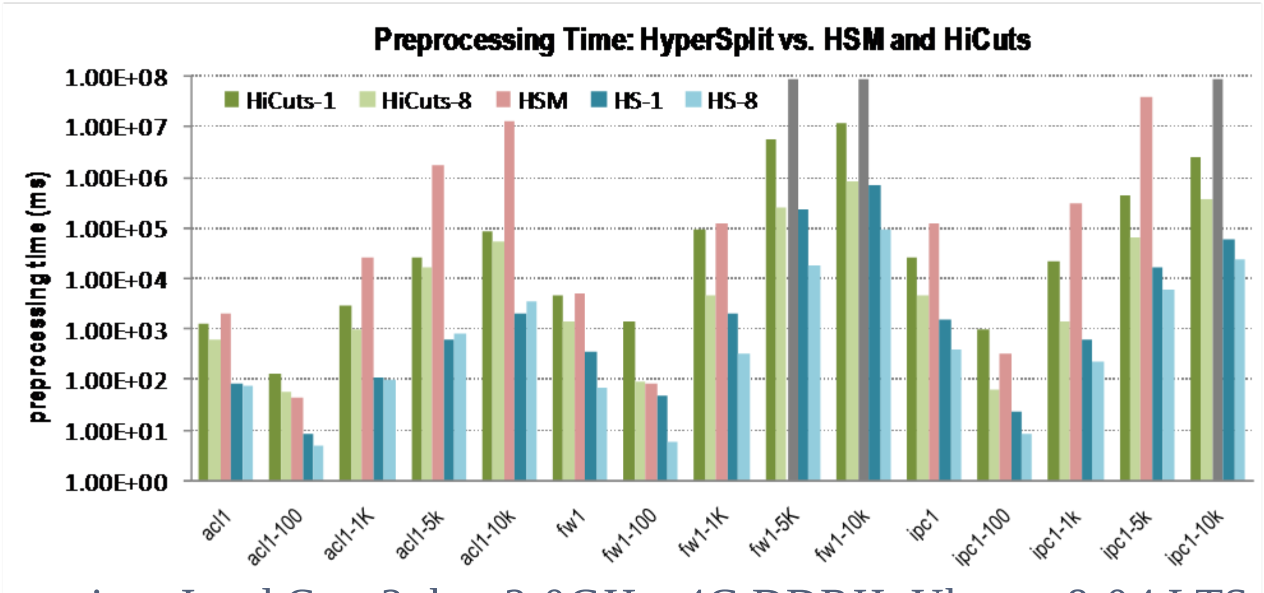
Memory Usage





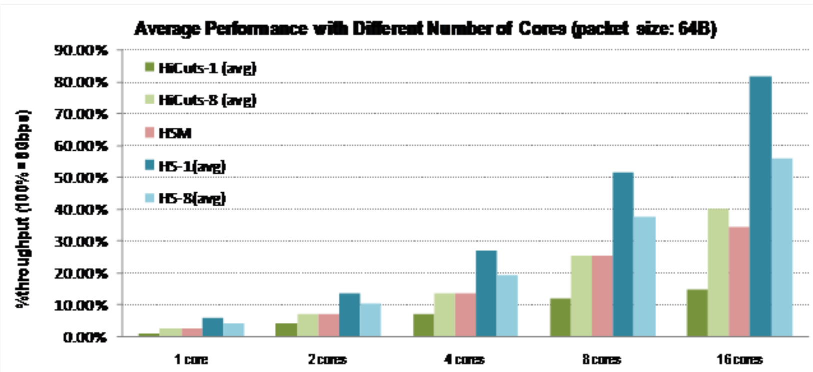
- HyperSplit-1 vs. HSM
  - 1~2 orders less memory;
  - HSM fails for fw1-5k, fw1-10k and ipc1-10k (due to 4GB+ memory usage)
- HyperSplit-1 vs. HiCuts-1
  - 1~2 orders less memory;
  - HiCuts-1 fails for fw1-5k, fw1-10k and ipc1-10k (due to 4GB+ memory usage)
- HyperSplit-8 vs. HiCuts-8
  - 1~2 orders less memory; successful for all rule-sets.

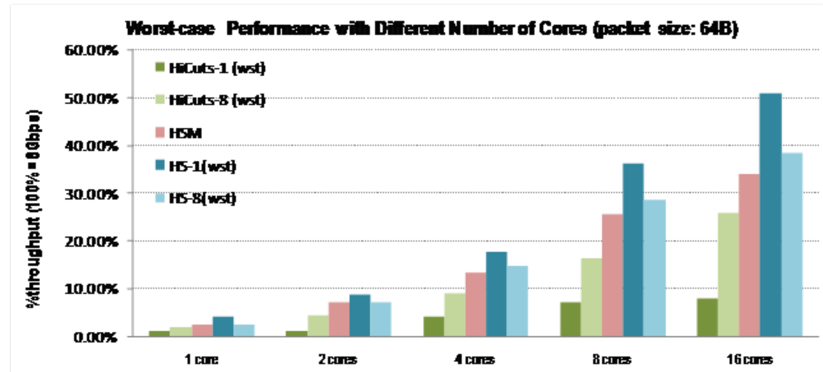
Preprocessing Time

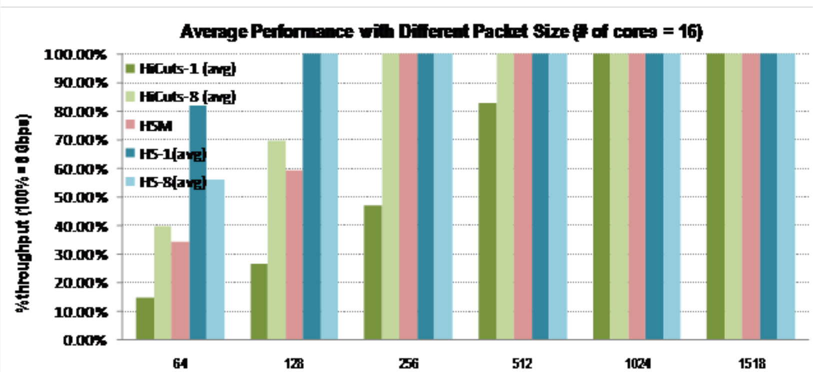


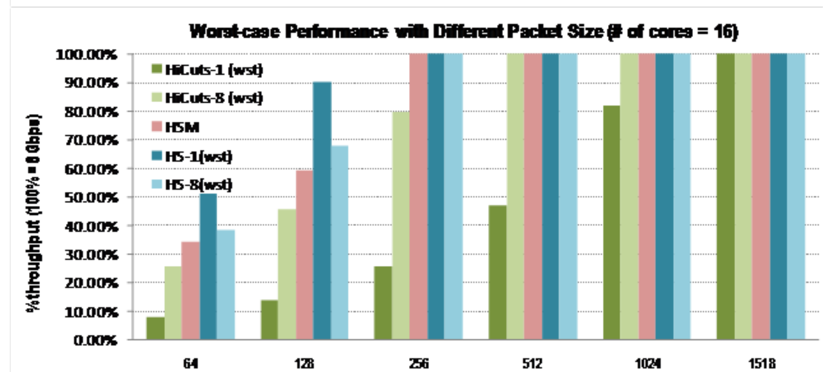
- Pre-processing: Intel Core2 duo 2.0GHz, 4G DDRII, Ubuntu8.04 LTS
- HyperSplit-1 vs. HSM
  - 1~4 orders less time
  - HSM fails for fw1-5k, fw1-10k and ipc1-10k (e.g. 24 hours for fw1-10k, and failed)
- HyperSplit-1 vs. HiCuts-1
  - 1~2 orders less time
  - HiCuts-1 fails for fw1-5k, fw1-10k and ipc1-10k (e.g. 4 hours for fw1-10k, and failed)
- HyperSplit-8 vs. HiCuts-8
  - About 1 order less time

Throughput









- 64B packet-size test with different # of cores:
  - HyperSplit: 6.4/4.2Gbps with 16 cores; HSM: 2.4Gbps; HiCuts: 1.1/3.2Gbps
- Variable packet-size test with 16 cores:
  - HyperSplit: 8Gbps with 128B+ packets; HSM: 256B+; HiCuts: 1024B+/256B+

Outline

- Packet Classification Introduction
- Review of Classic Algorithms
- Our Recent Advancement
- Conclusion and Discussion

Conclusion and Discussion

- Conclusion
  - Theoretically: Explicit worst-case search time O(log$N$)
  - Practically: 6.4Gbps on OCTEON3860, apply to all data sets @WUSTL
- Discussion
  - Adaptive to different memory hierarchy rather than the L2-DRAM coherent memory system?
  - Policy-based switching rather than routing?
  - Application identification rather than flow classification?

References

F. Geraci, M. Pellegrini, and P. Pisati “Packet Classification via Improved Space Decomposition Techniques”, in IEEE INFOCOM, 2005.

M. H. Overmars and A. F. van der Stappen, “Range Searching and Point Location among Fat Objects”, Journal of Algorithms, 21(3), 1996.

P. Gupta and N. McKewon, “Algorithms for Packet Classification”, IEEE Network, March/April, 2001.

D. E. Taylor, “Survey & Taxonomy of Packet Classification Techniques”, Technical Report, Washington University in Saint-Louis, USA, 2004.

M. E. Kounavis, A. Kumar, H. Vin, R. Yavatkar, and A. T. Campbell, “Directions in Packet Classification for Network Processors”, Proc. of the 2nd Workshop on Network Processors (NP2), 2003.

S. Singh, F. Baboescu, G. Varghese, and J. Wang, “Packet Classification Using Multidimensional Cutting”, Proc. of ACM SIGCOMM, 2003.

B. Xu, D. Jiang, and J. Li, “HSM: A Fast Packet Classification Algorithm”, Proc. of the 19th International Conference on Advanced Information Networking and Applications (AINA), 2005.

P. Gupta and N. McKeown, “Packet Classification Using Hierarchical Intelligent Cuttings”, Proc. Hot Interconnects, 1999.

P. Gupta and N. McKeown, “Packet Classification on Multiple Fields”, Proc. ACM SIGCOMM, 1999.

Y. X. Qi, B. Xu, F. He, B. H. Yang. J. M. Yu, and J. Li, “Towards High-performance Flow-level Packet Processing on Multi-core Network Processors”, Proc. of ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), 2007.

Y. X. Qi, B. Xu, F. He, X. Zhou, J. M. Yu, and Jun Li, “Towards Optimized Packet Classification Algorithms for Multi-Core Network Processors”, Proc. of the 2007 International Conference on Parallel Processing (ICPP), 2007.

D. Liu, B. Hua, X. Hu, and X. Tang, “High-performance Packet Classification Algorithm for Many-core and Multithreaded Network Processor”, Proc. of the 6th IEEE International Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES 2006), 2006.

T.Y.C Woo “A Modular Approach to Packet Classification: Algorithms and Results”, in IEEE INFOCOM, 2000.

D. E. Taylor and J. S. Turner, "Scalable packet classification using distributed crossproducting of field labels," in Proc. IEEE INFOCOM, Mar. 2005, pp. 269-280.

D.E. Taylor and J.S. Turner, “ClassBench: a packet classification benchmark,” INFOCOM 2005, vol.3, pp. 2068-2079, 13-17 March 2005.

H. Lim and J. H. Mun, “High-Speed Packet Classification Using Binary Search on Length”, Proc. of ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), 2007.

Our Recent Publications

- [1] Yaxuan Qi, Lianghong Xu, Baohua Yang, Yibo Xue, and Jun Li, Packet Classification Algorithms: From Theory to Practice, Proc. of the 28th IEEE **INFOCOM**, 2009.
- [2] Fei He, Yaxuan Qi, Yibo Xue, and Jun Li, Load Scheduling for Flow-based Packet Processing on Multi-core Network Processors, Proc. of the 20th IASTED International Conference on Parallel and Distributed Computing and Systems (**PDCS**), 2008.
- [3] Yaxuan Qi, Zongwei Zhou, Baohua Yang, Fei He, Yibo Xue, and Jun Li, Towards Effective Network Algorithms on Multi-core Network Processors, Proc. of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems (**ANCS**), 2008. (short paper)
- [4] Bo Xu, Yaxuan Qi, Fei He, Zongwei Zhou, Yibo Xue, and Jun Li, Fast Path Session Creation on Network Processors, Proc. of the 28th International Conference on Distributed Computing Systems (**ICDCS**), 2008.
- [5] Xin Zhou, Bo Xu, Yaxuan Qi, and Jun Li, MRSI: A Fast Pattern Matching Algorithm for Anti-virus Applications, Proc. of the 7th International Conference on Networking (**ICN**), 2008.
- [6] Yaxuan Qi, Bo Xu, Fei He, Baohua Yang, Jianming Yu, and Jun Li, Towards High-performance Flow-level Packet Processing on Multi-core Network Processors, Proc. of the 3rd ACM/IEEE Symposium on Architectures for Networking and Communications Systems (**ANCS**), 2007.
- [7] Yaxuan Qi, Bo Xu, Fei He, Xin Zhou, Jianming Yu, and Jun Li, Towards Optimized Packet Classification Algorithms for Multi-Core Network Processors, Proc. of the 2007 International Conference on Parallel Processing (**ICPP**), 2007.
- [8] Yaxuan Qi, Baohua Yang, Bo Xu, and Jun Li, Towards System-level Optimization for High Performance Unified Threat Management, Proc. of the 3rd International Conference on Networking and Services (**ICNS**), 2007. (best paper award)

Thanks!
Questions?