Towards High-performance Flow-level Packet Processing on Multi-core Network Processors

Yaxuan Qi (presenter), Bo Xu, Fei He,
Baohua Yang, Jianming Yu and Jun Li

ANCS 2007, Orlando, USA

Network Security Lab, RIIT
Tsinghua University

Outline

- Introduction
- Related Work on Multi-core NP
- Flow-level Packet Processing
  - Flow Classification
  - Flow State Management
  - Per-flow Packet Ordering
- Summary

Outline

- Introduction
- Related Work on Multi-core NP
- Flow-level Packet Processing
  - Flow Classification
  - Flow State Management
  - Per-flow Packet Ordering
- Summary

Introduction

- Why flow-level packet processing with high-performance?
  - increasing sophistication of applications
    - stateful firewalls
    - deep inspection in IDS/IPS
    - flow-based scheduling in load balancers
  - continual growth of network bandwidth
    - oc192 or higher link speed
    - 1 million or more concurrent connections

Introduction

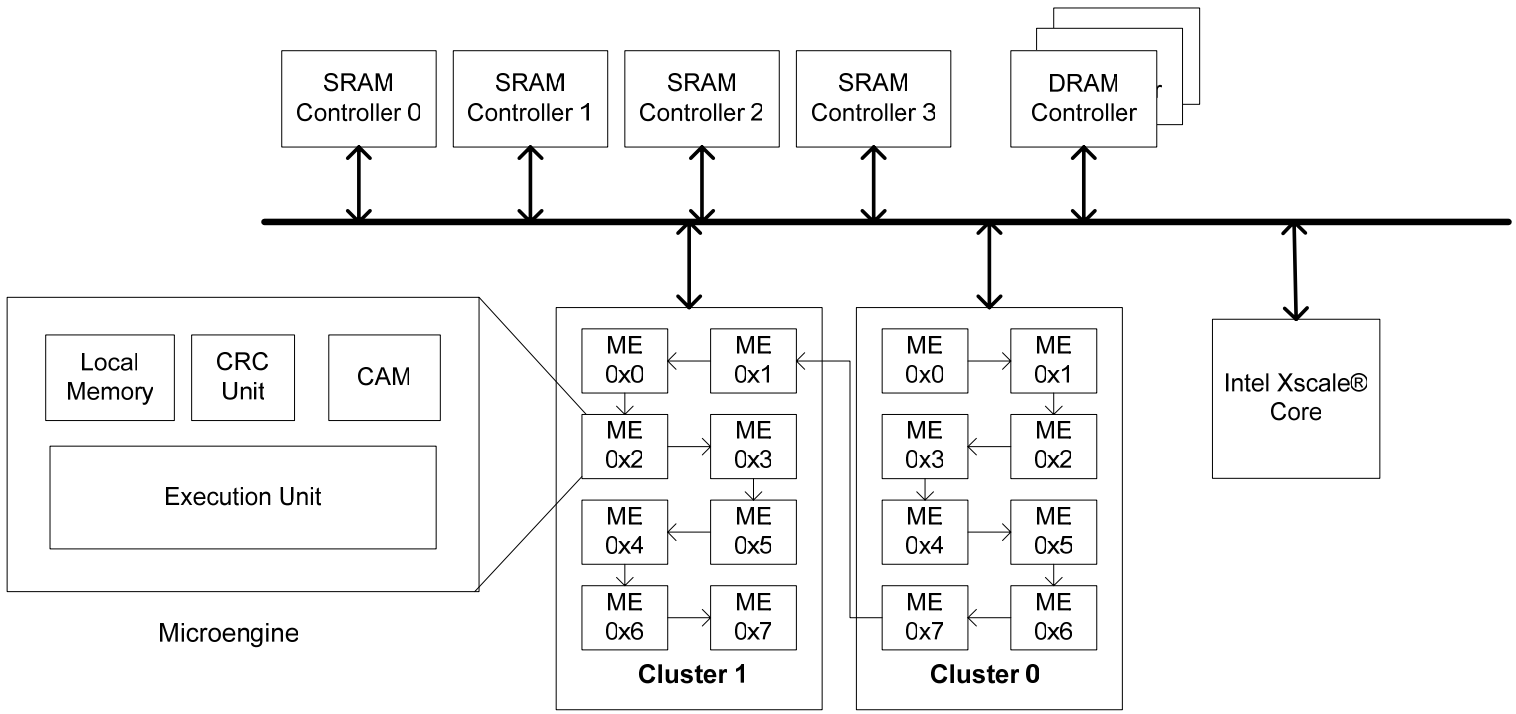
- Problems in flow-level packet processing:
  - Flow classification:
    - Importance: access control and protocol analysis
    - difficulty: high-speed with modest memory
  - Flow state management:
    - Importance: stateful firewall and anti-DoS
    - Difficulty: fast update with large connections
  - Per-flow packet order-preserving:
    - Importance: content inspection
    - Difficulty: mutual exclusion and workload distribution

Outline

- Introduction
- Related Work on Multi-core NP
- Flow-level Packet Processing
  - Flow Classification
  - Flow State Management
  - Per-flow Packet Ordering
- Summary

Related Work on Multi-core NP

- Intel IXP2850



Related Work on Multi-core NP

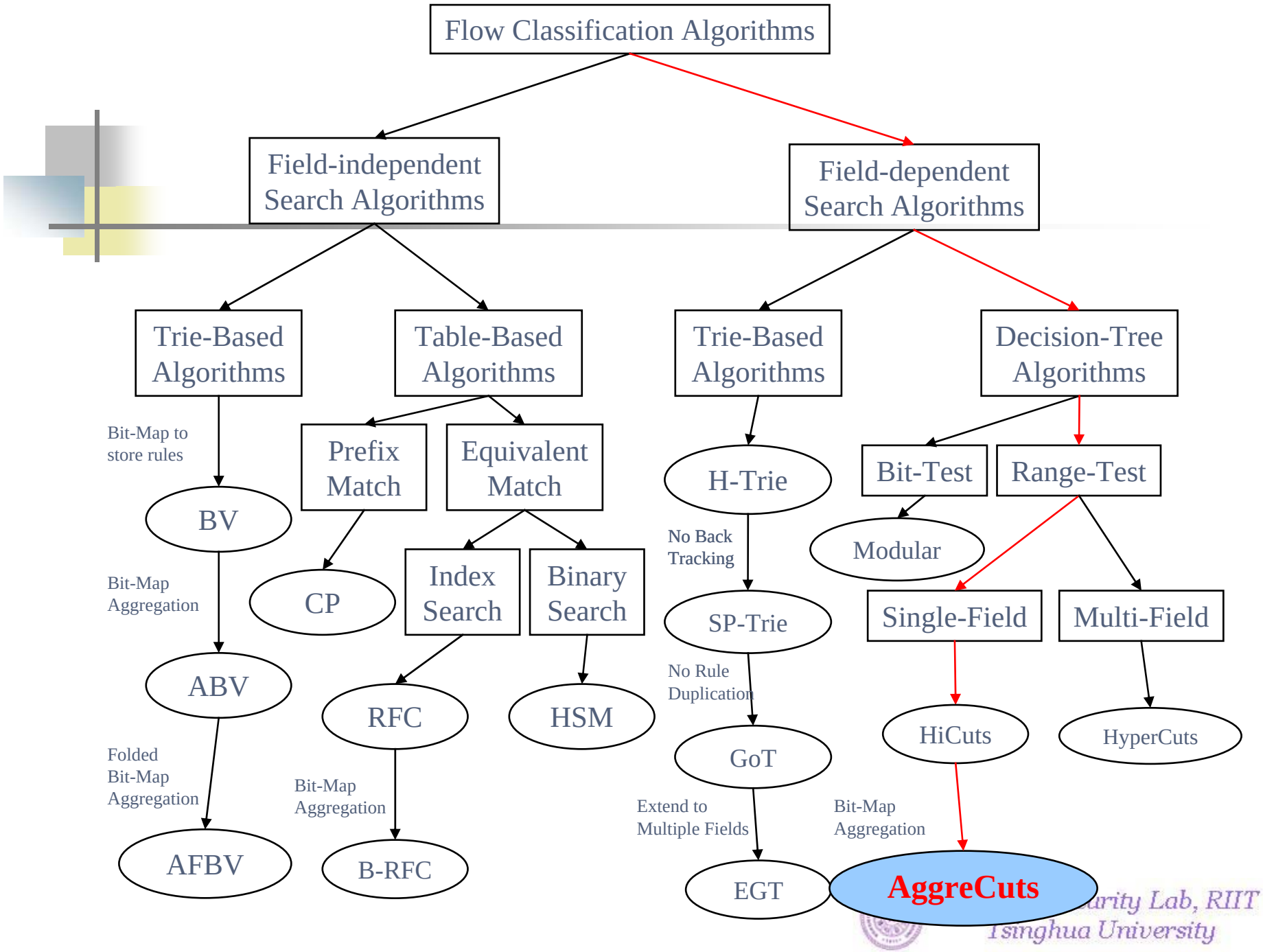
- Programming Challenges:
  - Achieving a deterministic bound on packet processing operation
    - line rate constraint
    - clock cycles to process the packet should have an upper bound
  - Masking memory latency through multi-threading:
    - memory latencies are typically much higher than the amount of processing budget
  - Preserving packet order in spite of parallel processing:
    - extremely critical for applications like media gateways and traffic management.

Outline

- Introduction
- Related Work on Multi-core NP
- Flow-level Packet Processing
  - Flow Classification
  - Flow State Management
  - Per-flow Packet Ordering
- Summary

Flow Classification

- Related work:
  - D. Srinivasan and W. Feng, Lucent Bit-Vector
    - On the Intel IXP1200 NP
    - Only support 512 rules
  - D. Liu, B. Hua, X. Hu and X. Tang, Bitmap RFC
    - Achieves near line speed on the Intel IXP2800.
    - 100MB+ SRAM memory for thousands of rules
  - Our study, Aggregated Cuttings (AggreCuts)
    - Near line speed on IXP2850
    - Consumes less than 10MB SRAM

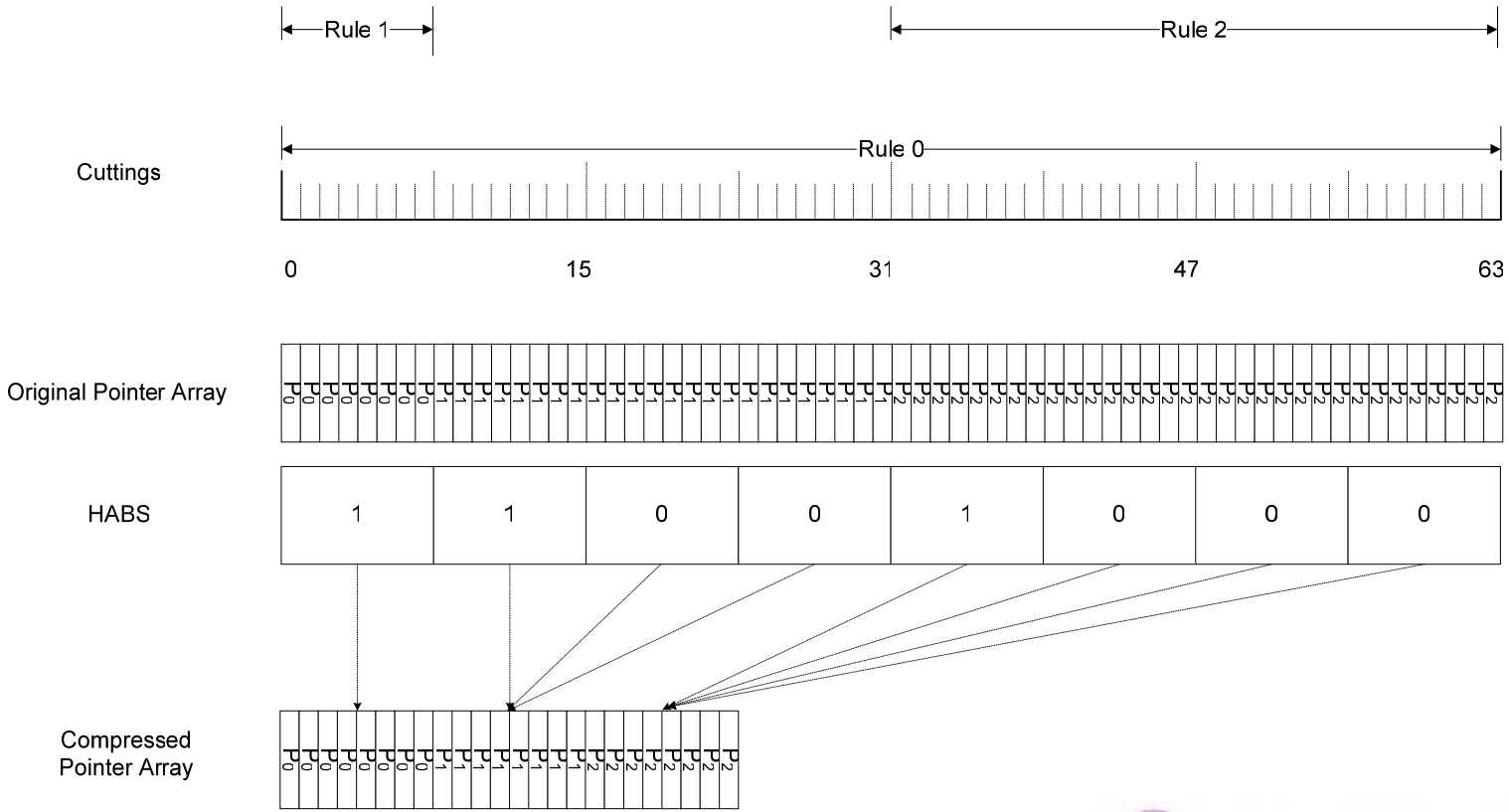


Flow Classification

- Why not HiCuts?
  - Non-deterministic worst-case search time
    - Due to heuristics used for #cuts
  - Excessive memory access
    - due to linear search on leaf-nodes (8Rules, <3Gbps on IXP28xx)
- Our motivations:
  - Fix the number of cuttings at internal-nodes:
    - If the number of cuttings is fixed to 2^w, then a worst-case bound of $O(W/w)$ is achieved (where W is header width, and w is stride)
  - Eliminate linear search at leaf-nodes:
    - Linear search can be eliminated if we "keep cutting" until every sub-space is full-covered by a certain set of rules.
- Consider the common 5-tuple flow classification problem
  - W=104, set w=8, then the worst-case search time
    - 104/8=13 (nearly the same as RFC)
  - No linear search is required

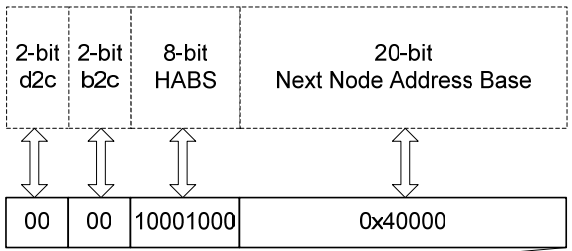
Flow Classification

- Space Aggregation



Flow Classification

- Data-structure



01	00	11000000	0x50000
00	01	10000000	0x52000
11	00	10100000	0x55000
10	00	10100001	0x58000

11	00	10100000	0x60000
11	00	10100000	0x60100
01	01	10110100	0x62000
10	00	10001000	0x62500

01	00	10001001	0x63000
00	02	10000100	0x63700

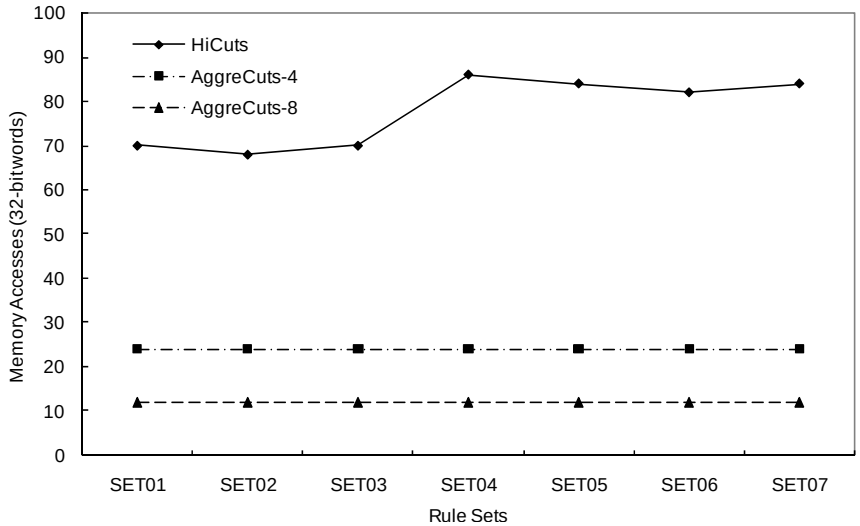
11	01	HABS	0x65000
01	00	HABS	0x65200
01	00	HABS	0x67000
01	00	HABS	0x67800

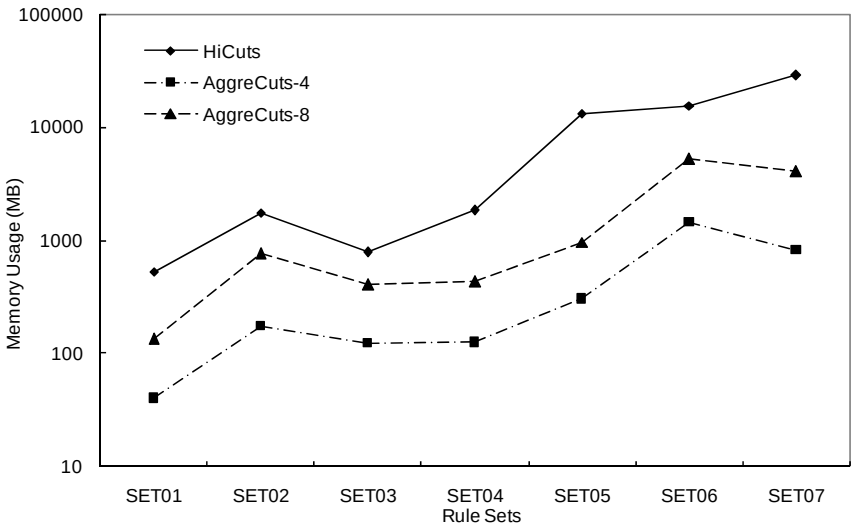
10	01	10101010	0x68200
10	01	10101010	0x68500
00	01	11100000	0x70000
01	00	10000101	0x71000
11	00	11000010	0x71500
00	01	10000010	0x73000

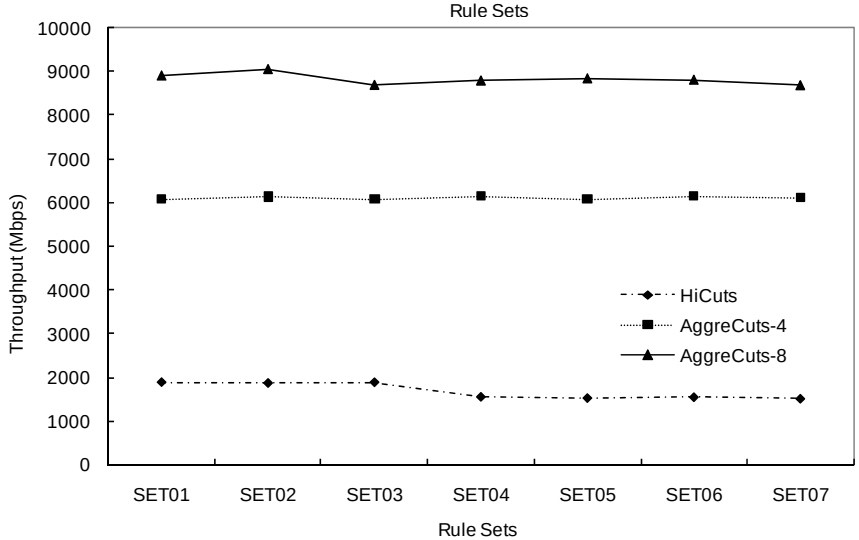
Bits	Description	Value
31:30	dimension to Cut (d2c)	d2c=00: src IP; d2c=01: dst IP; d2c=10: src port; d2c=11: dst port.
29:28	bit position to Cut (b2c)	b2c=00: 31~24; b2c=01: 23~16 b2c=10: 15~8; b2c=11: 7~0
27:20	8-bit HABS	if w=8, each bit represent 32 cuttings; if w=4, each bit represent 2 cuttings.
19:0	20-bit Next-Node CPA Base address	The minimum memory block is $2^w/8*4$ Byte. So if w=8, 20-bit base address support 128MB memory address space; if w=4, it supports 8MB memory address space.

Flow Classification

- Performance Evaluation
  - Memory Usage:
    - an order of magnitude
  - Memory Access:
    - 3~8 times less
  - Throughput on IXP2850:
    - 3~5 times faster







Outline

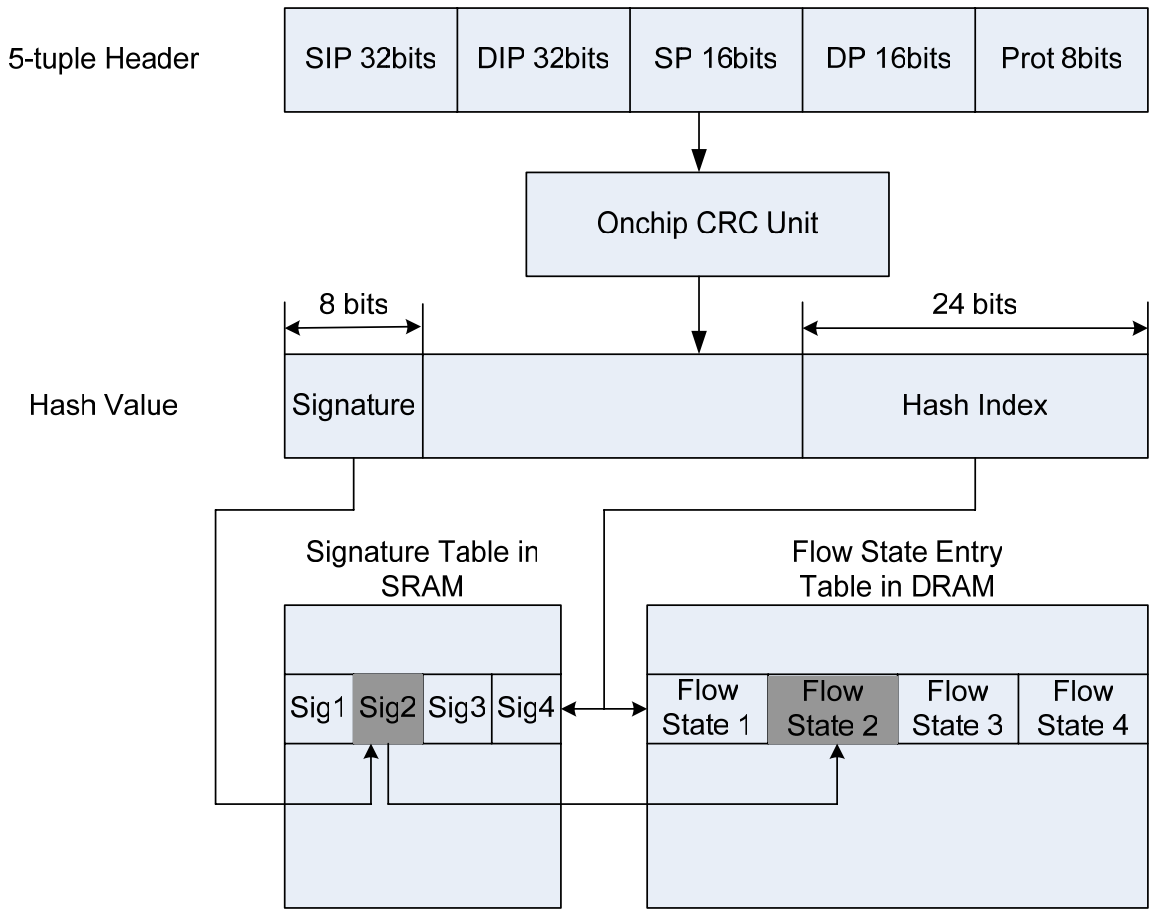
- Introduction
- Related Work on Multi-core NP
- Flow-level Packet Processing
  - Flow Classification
  - Flow State Management
  - Per-flow Packet Ordering
- Summary

Flow State Management

- Flow state management:
  - Problem:
    - A large number of updates over a short period of time
    - Line speed update
  - Solution:
    - Hashing with exact match
    - Collision and computation
- Our aim:
  - Supporting large concurrent sessions with extremely low collision rate
    - More than 10M session
    - Less than 1% collision rate
  - Achieving fast update speed using both SRAM and DRAM
    - Near line speed update rate

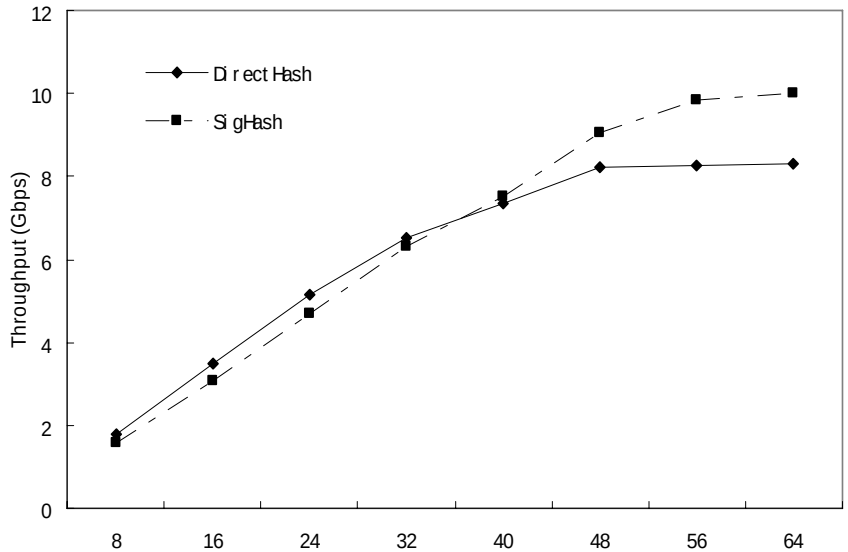
Flow State Management

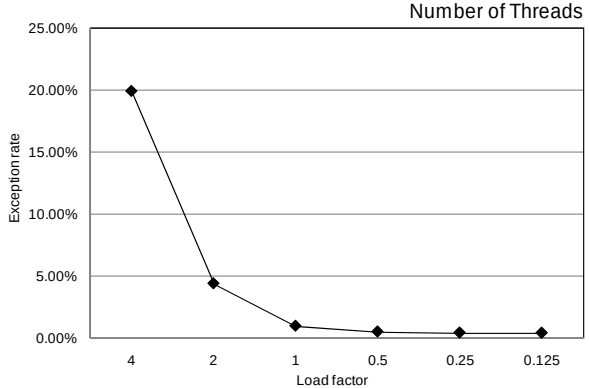
- Signature-based Hashing (SigHash)
  - *m* signatures for *m* different states with same hash value
  - Resolving collision in SRAM (fast, word-oriented)
  - Storing states in DRAM (large, burst-oriented)



Flow State Management

- Performance Evaluation
  - Throughput
    - 10Gbps
  - Connections
    - 10M
  - Collision
    - Less than 1%
    - Depends on different load factors





Outline

- Introduction
- Related Work on Multi-core NP
- Flow-level Packet Processing
  - Flow Classification
  - Flow State Management
  - Per-flow Packet Ordering
- Summary

Per-flow Packet Ordering

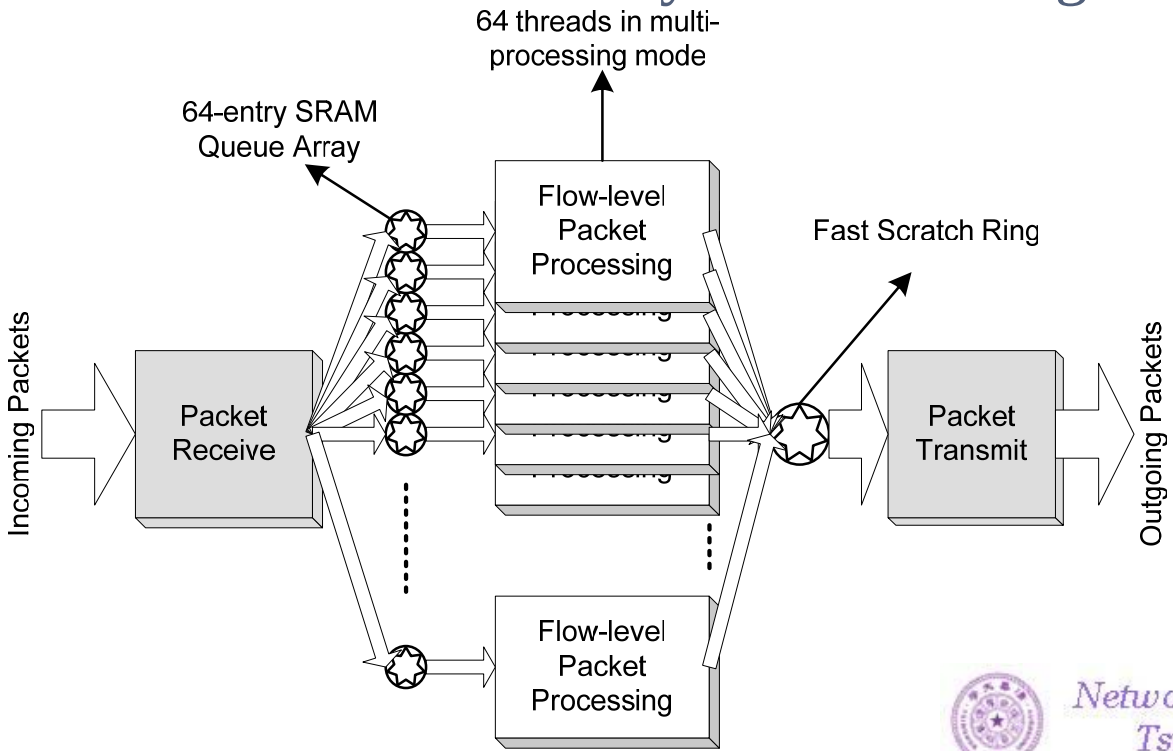
- Packet Order-preserving
  - Typically, only required between packets on the same flow.
- External Packet Order-preserving (EPO)
  - Sufficient for devices processing packets at network layer.
  - Fine-grained workload distribution (packet-level)
  - Need locking
- Internal Packet Order-preserving (IPO)
  - Required by applications that process packets at semantic levels.
  - Coarse-grained workload distribution (flow-level)
  - Do not need locking

Per-flow Packet Ordering

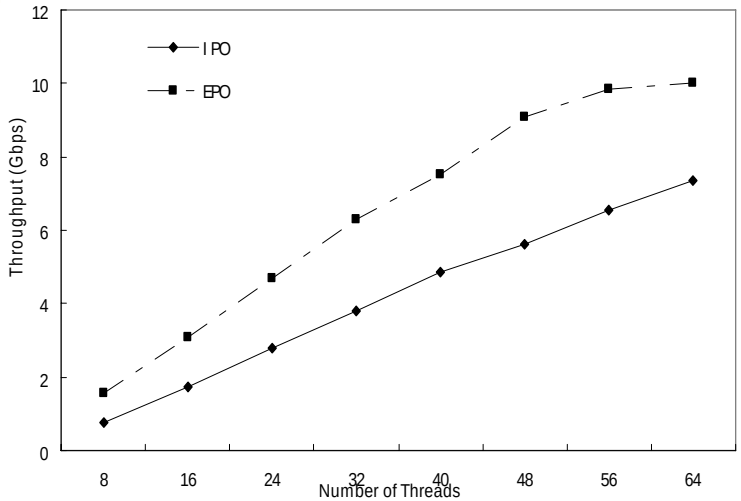
- External Packet Order-preserving (EPO)
  - Ordered-thread Execution
    - Ordered critical section to read the packet handles off the scratch ring.
    - The threads then process the packets, which may get out of order during packet processing.
    - Another ordered critical section to write the packet handles to the next stage.
  - Mutual Exclusion by Atomic Operation
    - Packets belong to the same flow may be allocated to different threads to process
    - Mutual exclusion can be implemented by locking.
    - SRAM atomic instructions

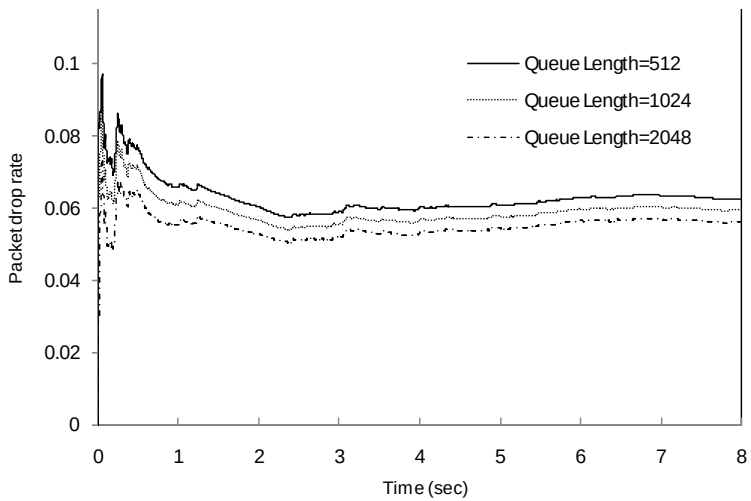
Per-flow Packet Ordering

- Internal Packet Order-preserving (IPO)
  - SRAM Q-Array
  - Workload Allocation by CRC Hashing on Headers



Per-flow Packet Ordering





- Performance Evaluation
  - Throughput
    - EPO is faster, 10Gbps
    - IPO has linear speed up, 7Gbps
  - Workload Allocation
    - CRC is good (though, Zipf-like)
    - While can be better

Outline

- Introduction
- Related Work on Multi-core NP
- Flow-level Packet Processing
  - Flow Classification
  - Flow State Management
  - Per-flow Packet Ordering
- Summary

Summary

- Contribution:
  - An NP-optimized flow classification algorithm :
    - Explicit worst-case search time: 9Gbps
    - hierarchical bitmap space aggregation: upto 16:1
  - An efficient flow state management scheme:
    - Fast update rate: 10Gbps
    - Exploit memory hierarchy: 10M connection, low collision rate
  - Two hardware-supported packet order-preserving schemes :
    - EPO via ordered-thread execution: 10Gbps
    - IPO via SRAM queue-array: 7Gbps
- Future work
  - Adaptive decision tree algorithm on for different memory hierarchy?
  - SRAM SYN-cookie for fast session creation?
  - Flow-let workload distribution?

Thanks ☺
Questions?