Fast Packet Forwarding on Commodity Platforms
Minlan Yu and Jennifer Rexford
Department of Computer Science, Princeton University

1. Motivation
- Comparing high end router and commodity platforms
  | High end routers | Multi-core commodity platforms |
  | Processor       | Separate control plane processor | Multiple general-purpose cores |
  | Memory          | Store routing table near control plane processor | Shared hierarchical memory architecture |
  | Interface       | High speed line cards with packet processing capability | NICs only receive and transmit packets |
  | Switch          | High speed switching fabric | PCI bus (interconnection network) |
- Packet forwarding on multi-core commodity platforms
  - Limitations
    - Packets are forwarded through a detour in the shared memory
    - Memory access becomes the main bottleneck
    - Especially for small-size packets
  - Leverage hierarchical memory architecture
    - Fit data structures into cache for fast access
    - Leverage multiple cores
      - Make packet forwarding easy to parallelize
      - Use multiple cores to forward packets
      - Map routing functions to different cores
- Comparing routing and conventional applications
  - Challenges
    - Routing is more sensitive to delay and delay jitter
    - Routing requires more predictable behavior under a range of workloads
    - A burst of packets with a wide range of destinations
    - Routing changes due to network event
  - Opportunities
    - Modularized routing functions
    - Easy to map to different cores

2. Design and System Architecture
- Packet forwarding on multi-core commodity platforms
  - Limitations
    - Packets are forwarded through a detour in the shared memory
    - Memory access becomes the main bottleneck
    - Especially for small-size packets
  - Leverage hierarchical memory architecture
    - Fit data structures into cache for fast access
    - Leverage multiple cores
      - Make packet forwarding easy to parallelize
      - Use multiple cores to forward packets
      - Map routing functions to different cores

3. Prototyping and Evaluation
- Prototyping on current commodity hardware
  - Implement consistent hashing in NetFPGA
  - Implement Bloom filter modules in Click
  - Evaluate packet forwarding performance under various workloads
- Prototyping hardware-based packet forwarding
  - Implement Bloom filters in NetFPGA
  - Manage routing changes and false positives in commodity hardware

4. Future Enhancement on Commodity Hardware
- Better support for packet forwarding in NIC
  - Use consistent hashing to demultiplex packets
  - To avoid significant changes of address range divisions
- Dedicated cores for packet forwarding
  - Larger cache (>1MB) to store forwarding table
  - Provide APIs for routing software to pin important data structures (e.g., forwarding table) in the cache
- Inter-NIC packet transmission
  - Store packet payload in NIC
  - Forward packet payload directly to the outgoing NIC

- Separate routing functions on different cores
  - Some cores to handle data plane functions
  - Other cores to handle routing changes and control plane functions
- Use multiple cores for packet forwarding
  - Divide forwarding table into address ranges
  - Each core is responsible for one address range
  - NICs demultiplex packets to the appropriate cores
  - Leverage VMDq (Virtual machine device queues) in modern NICs
- Use Bloom filters to fit forwarding table into cache
  - Each core maintains a Bloom filter for each next hop
  - Store the set of prefixes associated with the next hop
  - Adjust Bloom filter size to minimize false-positive rate
  - Perform conventional packet lookup on a false positive.
- Handle routing changes in large, slow memory
  - Maintain one counting Bloom filter corresponding to each bloom filter
  - Store counting Bloom filter in large, slow memory
  - Routing changes do not happen very often
  - Counting Bloom filters require larger memory space

- Separate routing functions on different cores
  - Some cores to handle data plane functions
  - Other cores to handle routing changes and control plane functions
- Use multiple cores for packet forwarding
  - Divide forwarding table into address ranges
  - Each core is responsible for one address range
  - NICs demultiplex packets to the appropriate cores
  - Leverage VMDq (Virtual machine device queues) in modern NICs
- Use Bloom filters to fit forwarding table into cache
  - Each core maintains a Bloom filter for each next hop
  - Store the set of prefixes associated with the next hop
  - Adjust Bloom filter size to minimize false-positive rate
  - Perform conventional packet lookup on a false positive.
- Handle routing changes in large, slow memory
  - Maintain one counting Bloom filter corresponding to each bloom filter
  - Store counting Bloom filter in large, slow memory
  - Routing changes do not happen very often
  - Counting Bloom filters require larger memory space