



OC-3072 packet classification using BDDs and pipelined SRAMs

Amit Prakash, Adnan Aziz
ECE Department
The University of Texas at Austin

Hot-Interconnects 2001



Outline

- Background
 - motivation, analytic formulation
 - previous approaches: software and hardware
- Our approach
 - logic synthesis using BDDs: target pipelined SRAM architecture
 - partition rules into priority classes to do classification on multiple fields
- Discussion
 - Key ideas, results, tradeoffs, comparisons



Motivation

- IP forwarding
- QoS Differentiation
- Firewalling
- Statistics Gathering
- Billing



Problem statement

- IP forwarding (best Prefix matching)
 - Find the port corresponding to longest prefix that matches destination address
- Packet classification on multiple fields
 - Condition: prefix, range, equality.
 - Predicate: collection of conditions on different fields of packet.
 - Rule: priority, predicate and an action.
 - Find the action corresponding to highest priority rule



Previous approaches

- IP forwarding
 - Tries, binary search on hash tables
 - Prefix Expansion
 - T-CAMs
- Classification on multiple fields
 - Hierarchical Tries
 - Heuristics to beat the space-time tradeoff
 - T-CAMs, ASICs



Our approach

- IP forwarding is a mapping of 32 bits of IP address to set $\{0,1,\dots,p-1\}$ where n is number of output port.
- Usually $p \leq 256$, can be encoded in 8 bits.
- Need to implement a combinational function with 32 input and 8 output efficiently.



Our approach

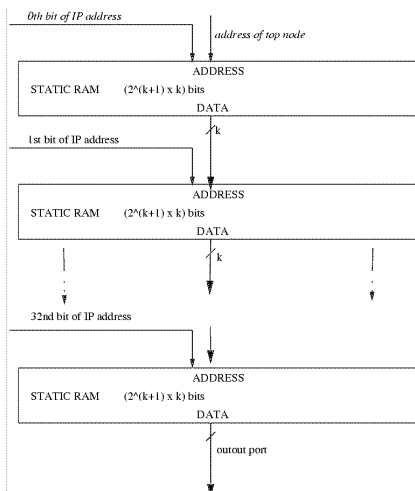
- Logic synthesis techniques to optimize the implementation to improve delay and area
- Use reconfigurable hardware to map the function
- First try with FPGAs did not work



Binary decision diagrams

- Split a function recursively
 $f(x_1, x_2, \dots, x_n) =$
 $x_1 \cdot f(1, x_2, \dots, x_n) + \bar{x}_1 \cdot f(0, x_2, \dots, x_n)$
- Use fixed order of variables
- Collapse two nodes if they represent the same function (can be exponential reduction)

Use of cascaded SRAMs



- Each SRAM Bank used to store nodes of one level
- Nodes consist of pointer to nodes in next level
- Address lines are driven by data lines of previous SRAM and one input bit

Extensions

- Allocate 16K nodes at each level=>can skip first 14 levels
- Almost all prefixes are 24bit long=>can skip last 10 levels
- Use MVBDDs to handle arbitrary nodes



Performance

- Evaluated with routing table of MAE-WEST
 - 57768 prefixes
 - 62 next hop addresses
 - 40995 BDD nodes (18803 nodes for trie)
 - Largest SRAM bank needed :28KB for 6803 nodes (160KB for 31714 nodes in trie)
 - Total SRAM needed 200KB
 - Throughput with state of the art SRAM 1 packet every 1 ns =>1000Mpps (OC-3072)
 - Latency 11 ns
 - 20s to build the data-structure on a 450Mhz Pentium III



Classification on multiple fields

- Direct approach: build BDD for classification function
 - Experimentally does not work – size of BDD blows up
 - Total ordering on rules => huge amount of state passed down the pipeline



Coping with BDD

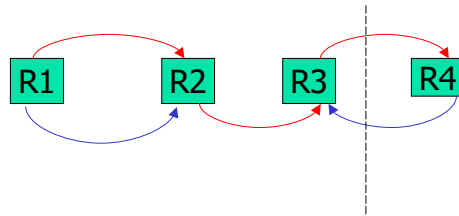
- Invert the problem – when will BDD size for classification function be linear in number of rules ?
 - order fields according to where they appear in BDD
 - define $p1 < p2$ iff $p1$ is a proper prefix of $p2$
 - $P1 <> p2$ iff $!(p1 < p2) \ \& \ !(p2 < p1)$
 - Define a predicate $C1 < C2$ iff, for some i , $C1(i) < C2(i)$ and $C1(j) <> C2(j)$ for $j < i$
 - Make sure if $C1 < C2$ then $C2$ has a higher priority than $C1$



Coping with BDD blowup

- Semantics
 - User assigns priority level to each rule from small predefined set of priorities, e.g., $\{0, 1, \dots, 7\}$
 - Multiple matches => highest priority applied
 - Classifier free to break ties in any fashion, uses $<$ relation defined earlier
- Architecture
 - Use multiple pipelined SRAM architectures to evaluate BDDs, one for each priority level
 - Use a priority encoder to decide which unit wins

A slightly better semantics



Partition the rules based on specified partial order

Latency vs Space

- 104 cycle latency for 5-tuple classification might be unacceptable
- Can cut latency by a factor f by increasing space requirement $2^f/f$ using multi-way split
- For 1024 Rules and 8 levels(128 rule per level)
 - 21 cycles latency (104/4)
 - 21 levels of SRAM
 - Each node splits 16-way (16 bytes)
 - Each bank size = $128 \times 16 = 2\text{KB}$
 - Number of SRAM banks = $8 \times 21 = 168$



Key ideas

- View classification rules as description of a boolean function
- Use logic synthesis ideas (BDDs)
- Use of pipelined SRAMs
 - Gives high throughput
 - Individual banks become much smaller allowing use of very fast memories
- Partition rule set intelligently to get linear size BDDs