Flow Caching for High Entropy Packet Fields

Nick Shelly
Nick McKeown

Ethan Jackson
Teemu Koponen
Jarno Rajahalme

Outline

- Current flow classification in OVS
- Problems with “high entropy” packets
- Proposed ideas
- Benchmarking
- Why it works

Definitions

- Flow classification
- Header space and wildcard bits: for the rule with match 10xx, packets 1011 and 1010 have the same action.
- High vs low entropy fields:

Packet 1

	L2		L3		L4	
Port	MAC src	MAC dst	IP src	IP dst	TP src	TP dst
2	00:11:22:33:44:55	00:11:22:33:44:55	69.171.248.18	96.30.52.192	52345	80

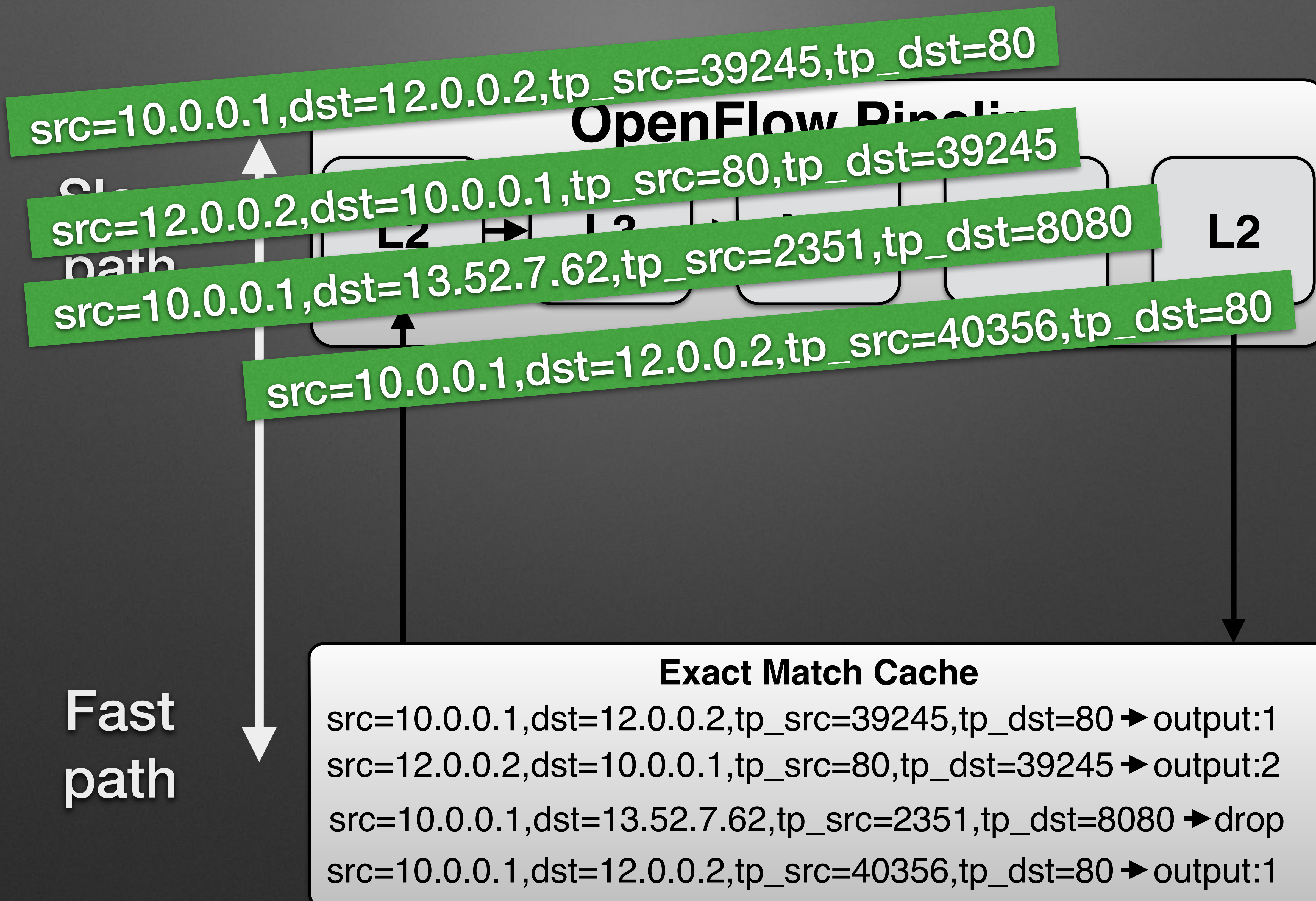
Low entropy fields

High entropy field

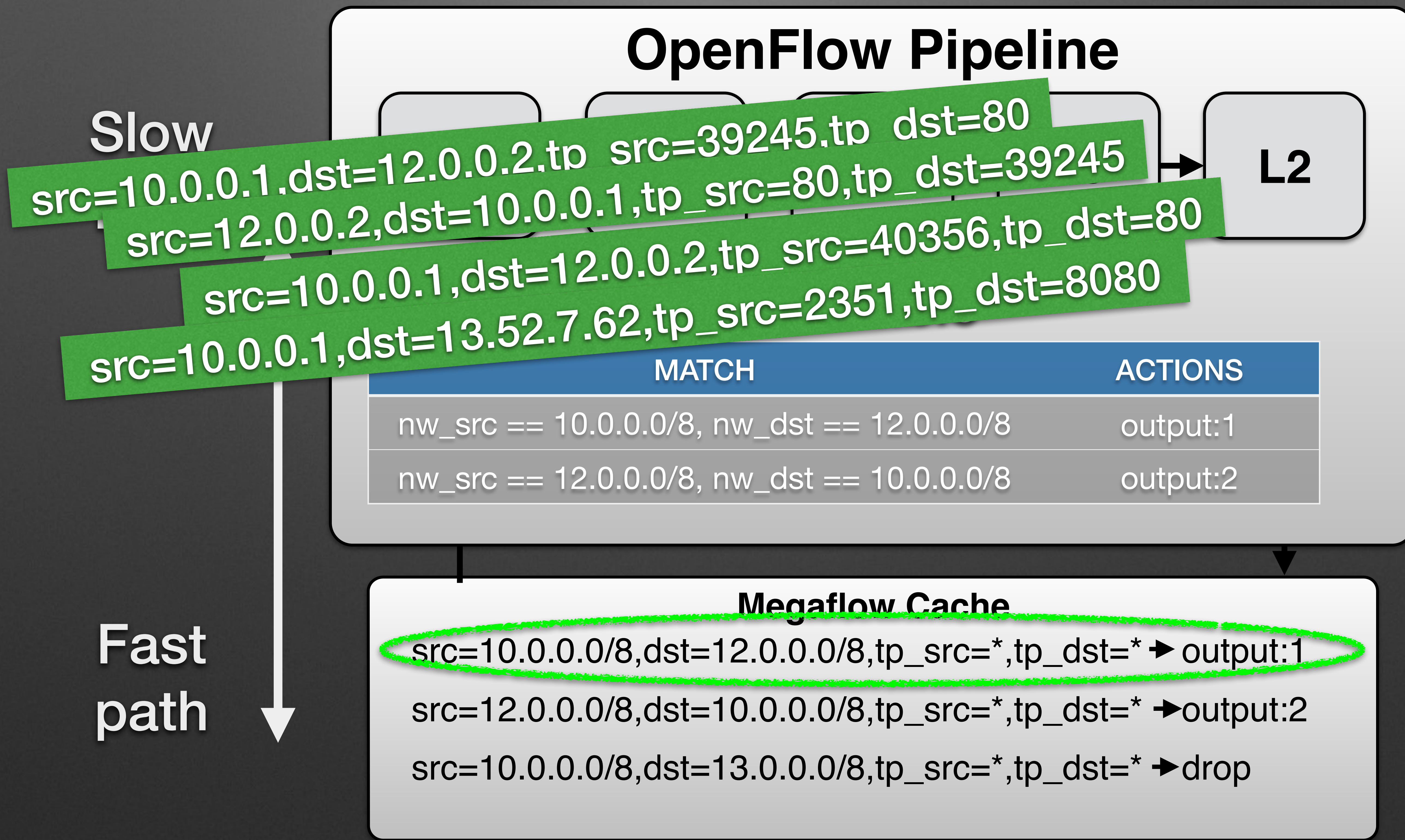
Packet 2

	L2		L3		L4	
Port	MAC src	MAC dst	IP src	IP dst	TP src	TP dst
2	00:11:22:33:44:55	00:11:22:33:44:55	69.171.248.18	96.30.52.192	52125	80

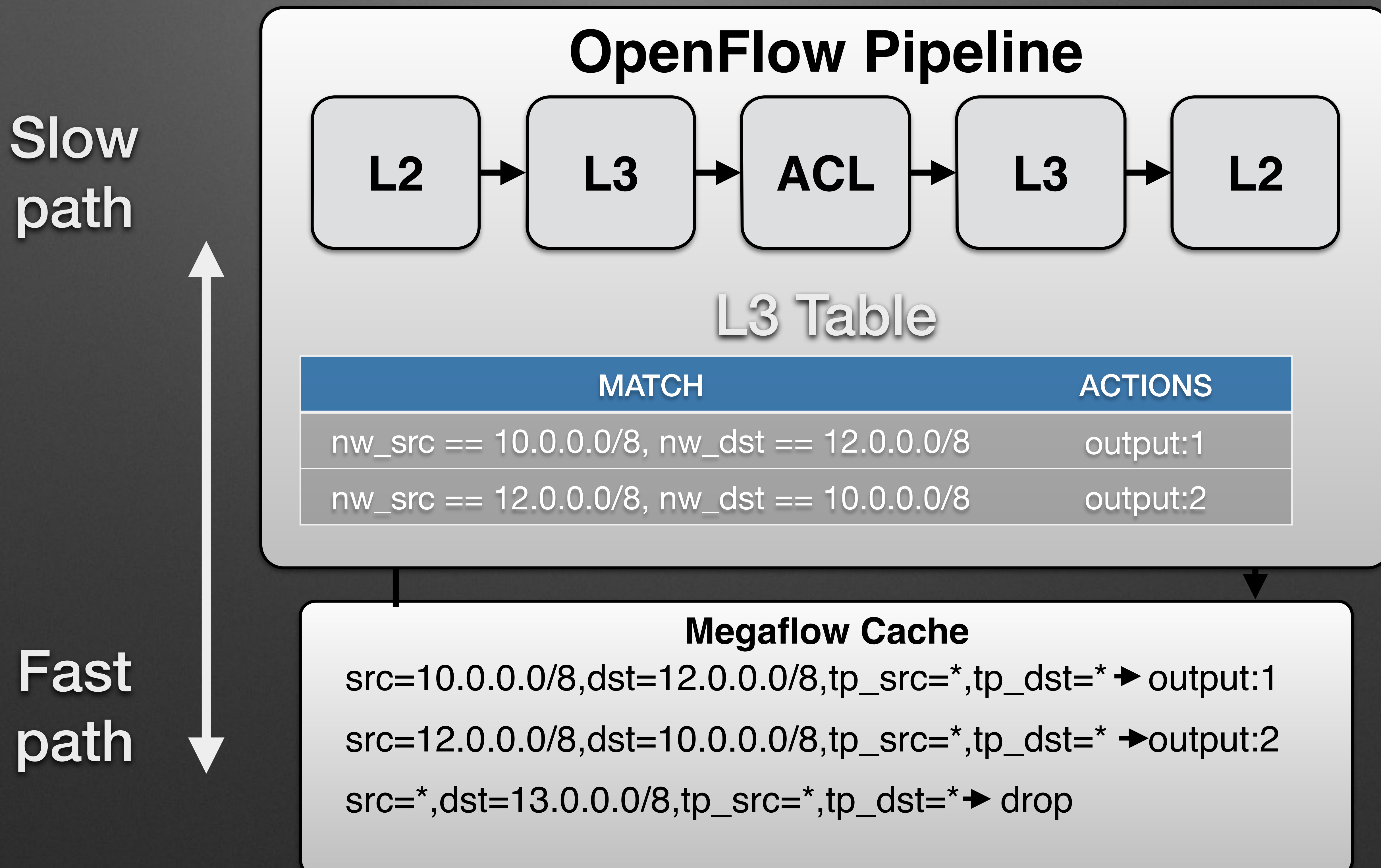
Flow classification in OVS



“Megaflow” Cache



“Megaflow” Cache



MATCH	ACTIONS
nw_src == 10.0.0.0/8, nw_dst == 12.0.0.0/8	output:1
nw_src == 12.0.0.0/8, nw_dst == 10.0.0.0/8	output:2

OVS slow path

- Relies on OpenFlow table pipeline
- Must handle highly complicated lookups— arbitrary packet metadata, recursion
- Computationally expensive
- Lots of repeat lookup sequences for packets that only differ by 1 or 2 bits

Ideal fast path

- Constant time lookup
- Easy to compute entries (translate from OpenFlow tables)
- Flexible for any number of OpenFlow fields
- Reasonable memory (we cannot cache every single packet)
- Easy to implement and maintain (in software)

Proposal 1. *Pro-active* processing

Table 1

RULE	PRIORITY	MATCH	ACTIONS
1	10	tp_dst == 80	drop
2	5	tp_dst == 443	drop
3	5	tp_dst == xx	reg4 <= 1, resubmit: table 2

→

Table 2

RULE	PRIORITY	MATCH	ACTIONS
1	1000	reg4 == 1	output:2

Cross-product Table

MATCH	ACTIONS	
tp_dst == 80	drop	Table 1, Rule 1
tp_dst == 443	drop	Table 1, Rule 2
tp_dst == 0-79, 81-442, 443 - 65535	reg4 <= 1, output: 2	Table 1, Rule 3 then Table 2, Rule 1

Header space notation:
tp_dst == xx - {80 U 443 }

Proposal 1. *Pro-active* processing

- But, how do we translate a header space, with a union, to wildcarded rules?

RULE	PRIORITY	MATCH	ACTIONS
A	10	tp_dst == 0101	drop
B	5	tp_dst == 0110	drop
C	5	tp_dst == xxxx	output: 2

$$\begin{aligned} h_c &= C - A - B \\ &= C - (A \cup B) \\ &= C \cap (A \cup B)' \end{aligned}$$

- Union minimization is an NP problem, so good heuristics is key

Proposal 2. *Re-active* processing

- Goal: For each new packet, install one wildcarded flow in the cache to match as many packets as possible

RULE	PRIORITY	MATCH	ACTIONS
A	10	tp_dst == 0101	drop
B	5	tp_dst == 0110	drop
C	5	tp_dst == xxxx	output: 2

New packet, p:
p = 1010

→

New cached flow:

MATCH	ACTIONS
tp_dst = 1xxxx	output: 2

- Doesn't cover the entire header space, h_c, but a good heuristic (with 100 ACLs, ~80% of true header space)

Proposal 2. *Re-active* processing

- Problem: Too many flow masks for most rule sets (easier to hash just on a few masks)

Fast path cache

FLOW MASK	RULES			
1000 0000	1xxx xxxx	0xxx xxxx		
1100 0000	01xx xxxx	10xx xxxx		
1110 0000	110x xxxx	101x xxxx	111x xxxx	011x xxxx
1111 0000	1010 xxxx	0011 xxxx	1011 xxxx	

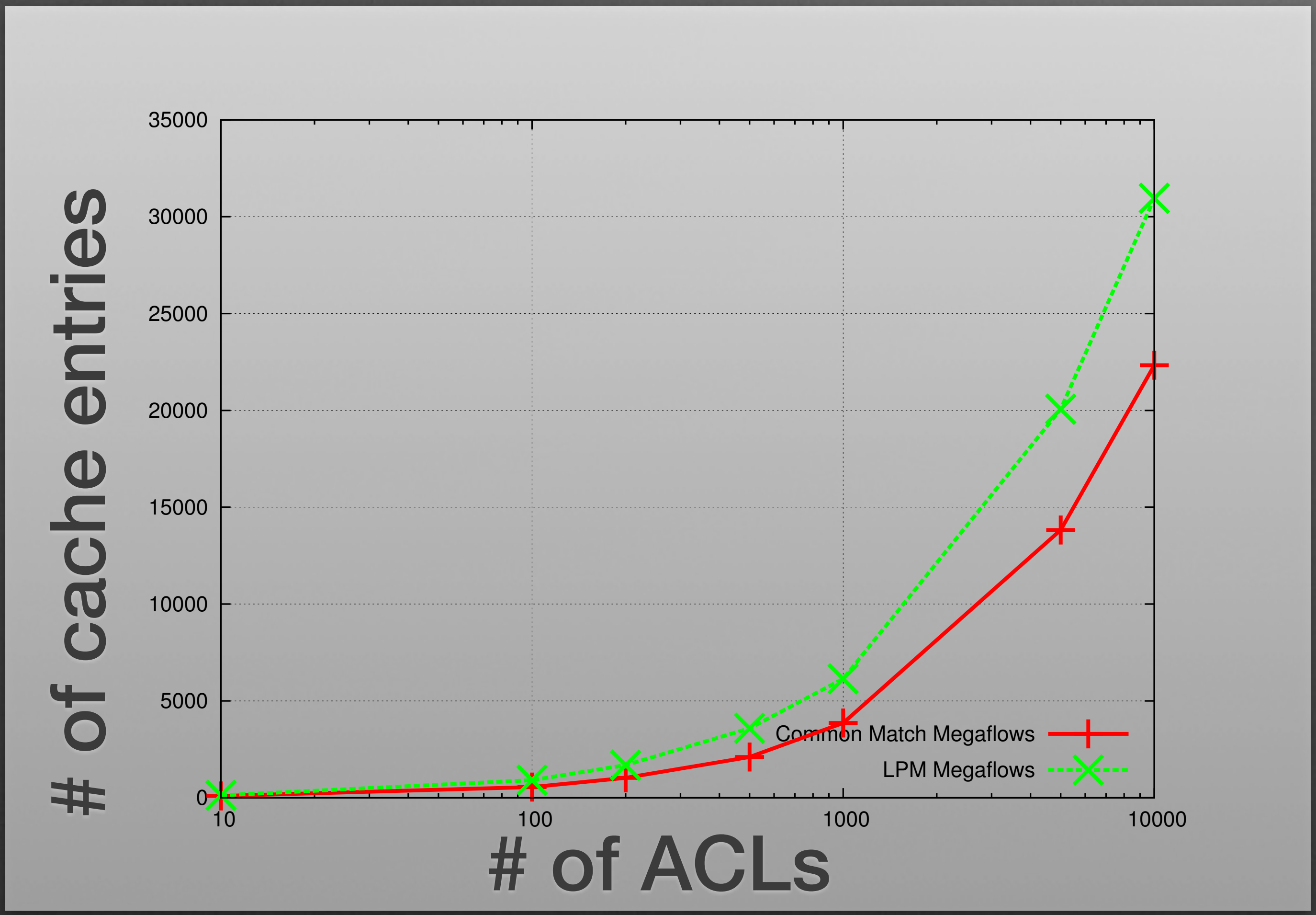
- Use decision tree to un-wildcard flows in order, maximum of N masks (for N-bit high entropy field)

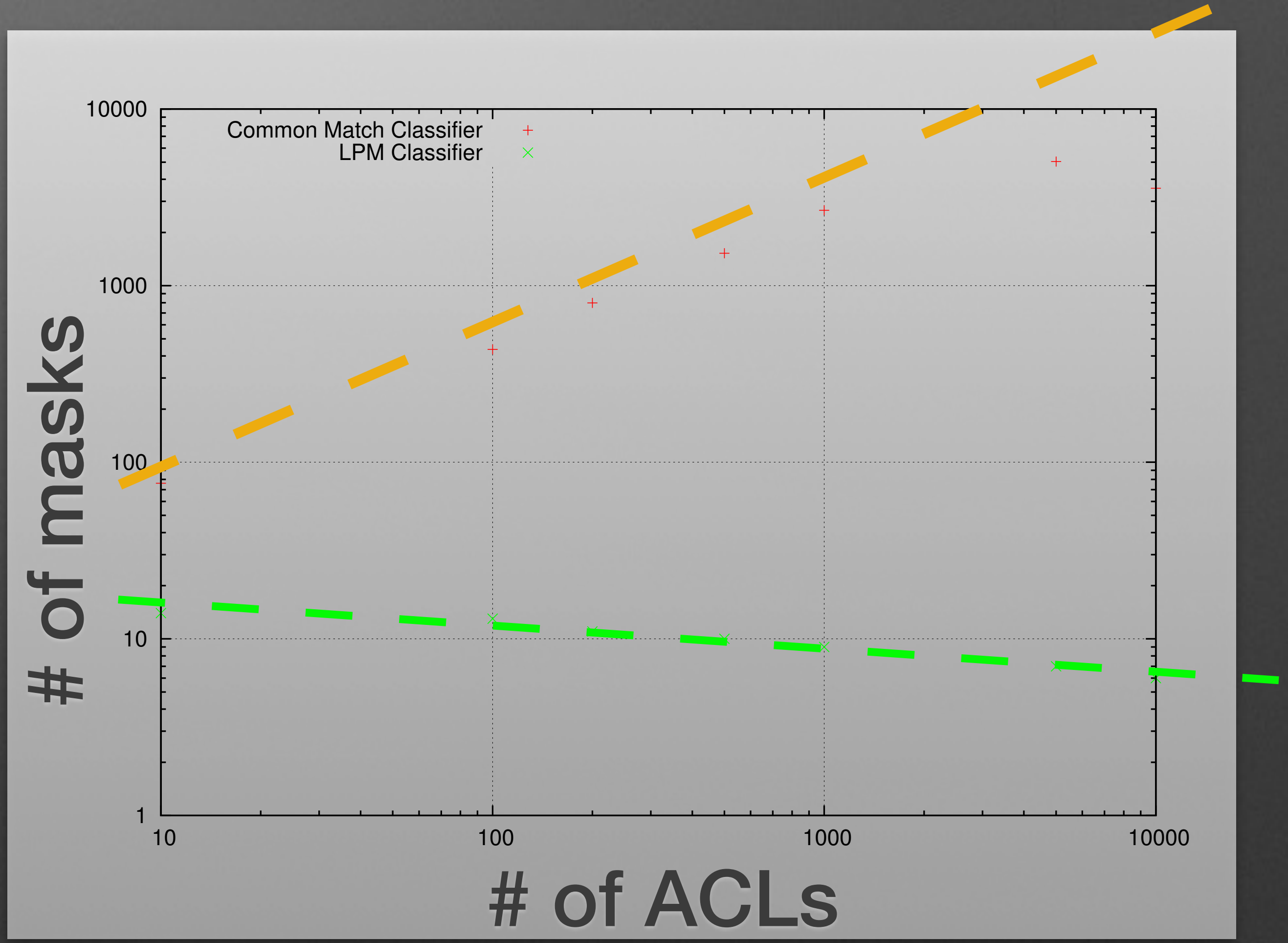
Comparison in performance

	COMMON MATCH			DECISION TREE		
ACLS	FLOWS	MASKS	% OPT	FLOWS	MASKS	% OPT
10	93	76	94%	126	14	16%
100	552	435	80%	913	13	13%
1000	3860	2669	57%	6141	9	-
10000	22333	3563	-	30947	6	-

Comparison in cache size

Can handle up to 1000 higher priority rules (ACLs) with only 5K entries





Longest prefix (decision tree) is better with the number of masks

Future work

- How does the number of hash masks affect run time?
- What is the effect of even more high entropy fields on the number of cache entries and coverage (optimal flow)?
- Can we adapt the classifier to different rule sets, exploiting traffic localities?
- Map one packet to more than one flow in the cache?

Questions?