

**P4TE**

# **Analyzing P4 Code to RMT Hardware Mapping**

---

Debobroto Das Robin  
Kent State University

<b>Ingress stage header analysis</b>	<b>3</b>
The total length of the header fields in ingress stage is 1216	3
<b>Egress stage header analysis</b>	<b>4</b>
The total length of the header fields in egress stage is 1488	4
<b>Header Analysis Summary:</b>	<b>4</b>
<b>Stagewise table and action mapping for PipelineID.INGRESS_PIPELINE is following :</b>	<b>5</b>
<b>Stagewise table and action mapping (with the stateful memory usage also) for PipelineID.EGRESS_PIPELINE is following :</b>	<b>32</b>
<b>Summary of resource usage in each stage is following</b>	<b>55</b>
<b>Result</b>	<b>78</b>



## Introduction

This document discusses the feasibility of P4TE into existing PISA hardware. Commonly available PISA contains 32 match-actions stages. Each stage having 16 2Kx40n wide TCAM. And these 32 stages are shared by both ingress and egress stage of a program. Therefore for both ingress and egress stage of the program can use 32 stages. The most unoptimized and trivial intermediate representation for the V1model.p4 is generated by the open source P4 compiler for V1model.p4. These intermediate representations can be found at

<https://github.com/drobinkent/P4TE/tree/main/p4src/Build>.

Please remember that, in this intermediate representation, the open source compiler represents a action as a table. Which is a very much unoptimized way . in real life a table and corresponding action (at least one independent action ) can be mapped in a single stage.

Our discussion is divided into 2 sections:

1. The P4 program for leaf switch
2. The P4 program for non-leaf switch



# The P4 program for leaf switch

## Header Analysis

### Ingress stage header analysis

- 1) following fields are used as match fields in the Ingress stage
  - a) 'ethernet.dst\_addr',
  - b) 'ipv6.src\_addr',
  - c) 'ipv6.traffic\_class',
  - d) 'scalars.userMetadata.egr\_port\_rate\_value\_range',
  - e) 'scalars.userMetadata.minimum\_group\_members\_requirement',
  - f) 'scalars.userMetadata.egr\_queue\_depth\_value\_range',
  - g) 'standard\_metadata.ingress\_port',
  - h) 'ipv6.dst\_addr'
- 2) Besides them 59 header fields are used as action fields
- 3) 12 extra header fields were used to carry result of conditional matching results.

The total length of the header fields in ingress stage is 1216

Bitwidth wise header count is {8: 30, 48: 7, 16: 8, 32: 8, 128: 2}



## Egress stage header analysis

- 4) Following fields are used as match fields in the Ingress stage
  - a) 'standard\_metadata.egress\_port'
- 5) Besides them, 51 header fields are used as action fields
- 6) 16 extra header fields were used to carry the result of conditional matching results.

The total length of the header fields in egress stage is 1488

Bitwidth wise header count is {32: 14, 16: 13, 8: 13, 128: 4, 48: 4, 24: 1}

## Header Analysis Summary:

The total number of header fields used by the P4 program for leaf switches is 1216+1488.

This is well within the 4K bit wide header vector availability in RMT hardwares. Moreover, in production-grade compilers, multiple fields are merged together which reduces the header fields length further. So in real life the resource consumption by the leaf switch of P4 program will be even lower.

## Stage wise Resource Mapping

Stagewise table and action mapping for  
PipelineID.INGRESS\_PIPELINE is following :

```
=====
=====
```

Stage:-----0

This is a dummy stage to handle a dummy node in the TDG. Not really mapped to the hardware.  
So please skip it.

```
=====
=====
```

Stage:-----1

MAT node: node\_2

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 125),  
('column', 8), ('source\_fragment', 'hdr.packet\_out.isValid())])")

```
=====
=====
```

Stage:-----2

MAT node: tbl\_leaf127

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: leaf127

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 127), ('column', 7), ('source\_fragment', 'standard\_metadata.egress\_spec = hdr.packet\_out.egress\_port')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 129), ('column', 7), ('source\_fragment', 'hdr.packet\_out.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 131), ('column', 7), ('source\_fragment', 'exit')])

MAT node: node\_4

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 132), ('column', 14), ('source\_fragment', 'hdr.packet\_in.isValid() && (standard\_metadata.instance\_type == BMV2\_V1MODEL\_INSTANCE\_TYPE\_RECIRC)'])")

=====

Stage:-----3

MAT node: tbl\_init\_pkt

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: IngressPipeImpl.init\_pkt

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 104), ('column', 23), ('source\_fragment', '0xF; ...')])



```

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 104), ('column', 23),
('source_fragment', '0xF; ...')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 62), ('column', 8),
('source_fragment', 'local_metadata.egress_rate_event_hdr.setValid()')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 104), ('column', 23),
('source_fragment', '0xF; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 105), ('column', 21),
('source_fragment', '0x0; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 104), ('column', 23),
('source_fragment', '0xF; ...')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 67), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.setValid()')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 69), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.do_l3_l2 = true')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 70), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_delay_proc = false')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 72), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_ing_queue_rate = false')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 73), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_ing_queue_depth = false')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 74), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_egr_queue_depth = false')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 75), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_egr_queue_rate = false')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 77), ('column', 33),
('source_fragment', '(bit<32>)standard_metadata.ingress_port')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 77), ('column', 8),
('source_fragment', 'ingressPortCounter.count((bit<32>)standard_metadata.ingress_port')])

```

```

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 79), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.downstream_routing_table_hit = false')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 80), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_dp_only_multipath_algo_processing_required =
false')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 81), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_fake_ack_for_rate_ctrl_required = false')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 83), ('column', 8),
('source_fragment', 'local_metadata.minimum_group_members_requirement=1')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 84), ('column', 8),
('source_fragment', 'local_metadata.egr_queue_depth_value_range = 1')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 85), ('column', 8),
('source_fragment', 'local_metadata.egr_port_rate_value_range = 1')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 136), ('column', 53),
('source_fragment', '0; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 138), ('column', 50),
('source_fragment', '0; ...')])

```

```

=====
=====

```

Stage:-----4

MAT node:

IngressPipeImpl.ingress\_rate\_monitor\_control\_block.flow\_type\_based\_ingress\_stats\_table

Match Keys are:

- \*) standard\_metadata.ingress\_port
- \*) ipv6.traffic\_class

Actions are:

1 Action Name:

IngressPipeImpl.ingress\_rate\_monitor\_control\_block.monitor\_incoming\_flow\_based\_on\_flow\_type\_for\_pkts\_rcvd\_from\_upstream

Primitives used in action are:

```
*) OrderedDict([('filename', 'p4src/src/ingress_rate_monitor.p4'), ('line', 73), ('column', 8), ('source_fragment', 'flow_type_based_ingress_meter_for_upstream.execute_meter((bit<32>)flow_type_based_meter_idx, local_metadata.ingress_rate_event_hdr.ingress_traffic_color)'))]
```

```
*) OrderedDict([('filename', 'p4src/src/ingress_rate_monitor.p4'), ('line', 74), ('column', 8), ('source_fragment', 'local_metadata.is_pkt_rcvd_from_downstream = false'))]
```

```
*) OrderedDict([('filename', 'p4src/src/ingress_rate_monitor.p4'), ('line', 76), ('column', 8), ('source_fragment', 'local_metadata.flag_hdr.is_packet_from_downstream_port = false'))]
```

```
*) OrderedDict([('filename', 'p4src/src/ingress_rate_monitor.p4'), ('line', 77), ('column', 8), ('source_fragment', 'local_metadata.flag_hdr.is_packet_from_upstream_port = true'))]
```

1 Action Name:

IngressPipeImpl.ingress\_rate\_monitor\_control\_block.monitor\_incoming\_flow\_based\_on\_flow\_type\_for\_pkts\_rcvd\_from\_downstream

Primitives used in action are:

```
*) OrderedDict([('filename', 'p4src/src/ingress_rate_monitor.p4'), ('line', 80), ('column', 8), ('source_fragment', 'flow_type_based_ingress_meter_for_downstream.execute_meter((bit<32>)flow_type_based_meter_idx, local_metadata.ingress_rate_event_hdr.ingress_traffic_color)'))]
```

```
*) OrderedDict([('filename', 'p4src/src/ingress_rate_monitor.p4'), ('line', 81), ('column', 8), ('source_fragment', 'local_metadata.is_pkt_rcvd_from_downstream = true'))]
```

```
*) OrderedDict([('filename', 'p4src/src/ingress_rate_monitor.p4'), ('line', 83), ('column', 8), ('source_fragment', 'local_metadata.flag_hdr.is_packet_from_downstream_port = true'))]
```

```
*) OrderedDict([('filename', 'p4src/src/ingress_rate_monitor.p4'), ('line', 84), ('column', 8), ('source_fragment', 'local_metadata.flag_hdr.is_packet_from_upstream_port = false'))]
```

1 Action Name: NoAction

Primitives used in action are:

=====

Stage:-----5

MAT node: node\_8

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 146), ('column', 8), ('source\_fragment', '(hdr.icmpv6.type == ICMP6\_TYPE\_NS ) && (hdr.icmpv6.type == ICMP6\_TYPE\_NS)'))")

=====

Stage:-----6

MAT node: IngressPipeImpl.ndp\_processing\_control\_block.ndp\_reply\_table

Match Keys are:

\*) ipv6.src\_addr

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: IngressPipeImpl.ndp\_processing\_control\_block.ndp\_ns\_to\_na

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 15), ('column', 7), ('source\_fragment', 'hdr.ethernet.src\_addr = target\_mac')])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 53), ('column', 33), ('source\_fragment', '0x33\_33\_00\_00\_00\_01; ...')])

\*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 21), ('column', 7), ('source\_fragment', 'hdr.ipv6.src\_addr = hdr.ndp.target\_ipv6\_addr')])

\*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 21), ('column', 27), ('source\_fragment', 'hdr.ndp.target\_ipv6\_addr; ...')])

```

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 49), ('column', 31),
('source_fragment', '58; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 56), ('column', 29),
('source_fragment', '136; ...')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 25), ('column', 7),
('source_fragment', 'hdr.ndp.flags = NDP_FLAG_ROUTER | NDP_FLAG_OVERRIDE')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 57), ('column', 38),
('source_fragment', '2; ...')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 27), ('column', 7),
('source_fragment', 'hdr.ndp.length = 1')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 28), ('column', 7),
('source_fragment', 'hdr.ndp.target_mac_addr = target_mac')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 29), ('column', 7),
('source_fragment', 'standard_metadata.egress_spec = standard_metadata.ingress_port')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 30), ('column', 7),
('source_fragment', 'local_metadata.flag_hdr.do_l3_l2 = false')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 31), ('column', 7),
('source_fragment', 'local_metadata.flag_hdr.is_pkt_toward_host = true')])

```

1 Action Name: IngressPipeImpl.ndp\_processing\_control\_block.ndp\_default\_action

Primitives used in action are:

```

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 34), ('column', 10),
('source_fragment', 'local_metadata.flag_hdr.do_l3_l2 = true')])

```

```

=====
=====

```

Stage:-----7

MAT node: tbl\_leaf150

Match Keys are:

Actions are:

1 Action Name: leaf150

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 150), ('column', 7), ('source\_fragment', 'exit')])

=====

Stage:-----8

MAT node: node\_11

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 154), ('column', 8), ('source\_fragment', 'local\_metadata.flag\_hdr.do\_I3\_I2')])")

=====

Stage:-----9

MAT node: IngressPipeImpl.I2\_ternary\_processing\_control\_block.I2\_ternary\_table

Match Keys are:

\*) ethernet.dst\_addr

\*) 8 bit key for handling conditional of previous stage


Actions are:

1 Action Name:

IngressPipeImpl.I2\_ternary\_processing\_control\_block.set\_multicast\_group

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/I2\_ternary.p4'), ('line', 26), ('column', 7), ('source\_fragment', 'standard\_metadata.mcast\_grp = gid')])

  
\*) OrderedDict([('filename', 'p4src/src/l2\_ternary.p4'), ('line', 27), ('column', 7), ('source\_fragment', 'local\_metadata.is\_multicast = true')])

1 Action Name: IngressPipeImpl.l2\_ternary\_processing\_control\_block.drop

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/l2\_ternary.p4'), ('line', 18), ('column', 7), ('source\_fragment', 'mark\_to\_drop(standard\_metadata)')])

=====

Stage:-----10

MAT node: tbl\_l2\_ternary44

Match Keys are:

Actions are:

1 Action Name: l2\_ternary44

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/l2\_ternary.p4'), ('line', 44), ('column', 12), ('source\_fragment', 'exit')])

=====

Stage:-----11

MAT node: IngressPipeImpl.my\_station\_processing\_control\_block.my\_station\_table

Match Keys are:

\*) ethernet.dst\_addr

Actions are:

1 Action Name: NoAction

Primitives used in action are:

=====

Stage:-----12

MAT node: tbl\_my\_station24

Match Keys are:

Actions are:

1 Action Name: my\_station24

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/my\_station.p4'), ('line', 24), ('column', 41), ('source\_fragment', 'local\_metadata.flag\_hdr.my\_station\_table\_hit = true')])

MAT node: tbl\_my\_station25

Match Keys are:

Actions are:

1 Action Name: my\_station25

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/my\_station.p4'), ('line', 25), ('column', 13), ('source\_fragment', 'local\_metadata.flag\_hdr.my\_station\_table\_hit = false')])

=====

Stage:-----13

MAT node: node\_17

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 159), ('column', 12), ('source\_fragment', 'hdr.ipv6.isValid() && local\_metadata.flag\_hdr.my\_station\_table\_hit')])")



=====

Stage:-----14

MAT node: IngressPipImpl.downstream\_routing\_control\_clock.downstream\_routing\_table

Match Keys are:

\*) ipv6.dst\_addr

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipImpl.downstream\_routing\_control\_clock.set\_downstream\_egress\_port

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf\_downstream\_routing.p4'), ('line', 17), ('column', 11), ('source\_fragment', 'standard\_metadata.egress\_spec = port\_num')])

\*) OrderedDict([('filename', 'p4src/src/leaf\_downstream\_routing.p4'), ('line', 18), ('column', 11), ('source\_fragment', 'hdr.ethernet.src\_addr = hdr.ethernet.dst\_addr')])

\*) OrderedDict([('filename', 'p4src/src/leaf\_downstream\_routing.p4'), ('line', 19), ('column', 11), ('source\_fragment', 'hdr.ethernet.dst\_addr = dmac')])

\*) OrderedDict([('filename', 'p4src/src/leaf\_downstream\_routing.p4'), ('line', 21), ('column', 11), ('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1')])

\*) OrderedDict([('filename', 'p4src/src/leaf\_downstream\_routing.p4'), ('line', 23), ('column', 12), ('source\_fragment', 'local\_metadata.flag\_hdr.downstream\_routing\_table\_hit = true')])

1 Action Name: NoAction

Primitives used in action are:

=====

Stage:-----15

MAT node: node\_19

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 162), ('column', 15), ('source\_fragment', 'local\_metadata.flag\_hdr.downstream\_routing\_table\_hit')]))"

=====

Stage:-----16

MAT node: tbl\_leaf163

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: leaf163

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 163), ('column', 16), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_pkt\_toward\_host = true')])

MAT node: tbl\_leaf167

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: leaf167

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 167), ('column', 16), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_pkt\_toward\_host = false')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 168), ('column', 16), ('source\_fragment', 'local\_metadata.flag\_hdr.found\_multi\_criteria\_paths = true')])

=====  
=====  
Stage:-----17

MAT node: node\_21

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 164), ('column', 19), ('source\_fragment', 'hdr.ipv6.hop\_limit == 0')]))"

MAT node:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_queue\_depth\_based\_upstream\_path\_table

Match Keys are:

\*) ipv6.dst\_addr

\*) scalars.userMetadata.egr\_queue\_depth\_value\_range

\*) scalars.userMetadata.minimum\_group\_members\_requirement

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_queue\_depth\_based\_upstream\_path\_selector\_action\_without\_param

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 60), ('column', 8), ('source\_fragment', 'local\_metadata.egr\_queue\_based\_path = 0')]))

\*) OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 61), ('column', 8), ('source\_fragment', 'local\_metadata.flag\_hdr.found\_egr\_queue\_depth\_based\_path = false')]))

1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_queue\_depth\_based\_upstream\_path\_selector\_action\_with\_param

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 65), ('column', 8),  
(('source\_fragment', 'local\_metadata.egr\_queue\_based\_path = port\_num'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 66), ('column', 8),  
(('source\_fragment', 'local\_metadata.flag\_hdr\_found\_egr\_queue\_depth\_based\_path = true'))])

=====

Stage:-----18

MAT node: tbl\_leaf164

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: leaf164

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 164), ('column', 46),  
(('source\_fragment', 'mark\_to\_drop(standard\_metadata)'))])

MAT node:

IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_port\_rate\_based\_upstream\_path\_table

Match Keys are:

\*) ipv6.dst\_addr

\*) scalars.userMetadata.egr\_port\_rate\_value\_range

\*) scalars.userMetadata.minimum\_group\_members\_requirement

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_port\_rate\_based\_upstream\_path\_selector\_action\_without\_param

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 106), ('column', 8),  
('source\_fragment', 'local\_metadata.egr\_rate\_based\_path = 0')])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 107), ('column', 8),  
('source\_fragment', 'local\_metadata.flag\_hdr.found\_egr\_queue\_rate\_based\_path = false')])

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_port\_rate\_based\_upstream\_path\_selector\_action\_with\_param

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 111), ('column', 8),  
('source\_fragment', 'local\_metadata.egr\_rate\_based\_path = port\_num')])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 112), ('column', 8),  
('source\_fragment', 'local\_metadata.flag\_hdr.found\_egr\_queue\_rate\_based\_path = true')])

=====

Stage:-----19

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_lookup\_flowlet\_id\_map

Match Keys are:

Actions are:

1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.lookup\_flowlet\_id\_map

Primitives used in action are:

```
*) OrderedDict([('filename',  
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 21), ('column', 8),  
('source_fragment', 'hash(local_metadata.flowlet_map_index, HashAlgorithm.crc16, (bit<13>)0, {  
hdr.ipv6.src_addr, hdr.ipv6.dst_addr, hdr.ipv6.flow_label, hdr.tcp.src_port, hdr.tcp.dst_port },  
(bit<13>)8191)'))]
```

```
*) OrderedDict([('filename',  
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 23), ('column', 8),  
('source_fragment', 'local_metadata.flow_inter_packet_gap =  
(bit<48>)standard_metadata.ingress_global_timestamp'))]
```

```
*) OrderedDict([('filename',  
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 24), ('column', 8),  
('source_fragment', 'flowlet_lasttime_map.read(local_metadata.flowlet_last_pkt_seen_time,  
(bit<32>)local_metadata.flowlet_map_index)'))]
```

```
*) OrderedDict([('filename',  
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 25), ('column', 8),  
('source_fragment', 'local_metadata.flow_inter_packet_gap =  
local_metadata.flow_inter_packet_gap - local_metadata.flowlet_last_pkt_seen_time'))]
```

```
*) OrderedDict([('filename',  
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 26), ('column', 8),  
('source_fragment', 'flowlet_lasttime_map.write((bit<32>)local_metadata.flowlet_map_index,  
standard_metadata.ingress_global_timestamp)'))]
```

=====

Stage:-----20

MAT node: tbl\_leaf135

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: leaf135

Primitives used in action are:

```
*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 135), ('column', 42),
('source_fragment', '(bit<32>)hdr.packet_in.egress_rate_event_port'))

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 135), ('column', 89),
('source_fragment', '(bit<4>)local_metadata.egress_rate_event_hdr.egress_traffic_color'))

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 135), ('column', 8),
('source_fragment',
'egress_queue_rate_value_map.write((bit<32>)hdr.packet_in.egress_rate_event_port,
(bit<4>)local_metadata.egress_rate_event_hdr.egress_traffic_color )'))

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 136), ('column', 8),
('source_fragment', 'mark_to_drop(standard_metadata)'))
```

MAT node: node\_27

Match Keys are:

Actions are:

```
*) SourceInfo(srcinfo="OrderedDict([('filename',
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 80), ('column', 12),
('source_fragment', '(hdr.ipv6.traffic_class == TRAFFIC_CLASS_HIGH_THROUGHPUT) ||
(hdr.ipv6.traffic_class == TRAFFIC_CLASS_HIGH_THROUGHPUT2)'))")
```

=====

Stage:-----21

MAT node: node\_28

Match Keys are:

```
*) 8 bit key for handling conditional of previous stage
```

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 82), ('column', 16),  
(('source\_fragment', 'local\_metadata.flow\_inter\_packet\_gap >  
FLOWLET\_INTER\_PACKET\_GAP\_THRESHOLD'))]))")

MAT node: node\_39

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 99), ('column', 17),  
(('source\_fragment', 'local\_metadata.flow\_inter\_packet\_gap >  
FLOWLET\_INTER\_PACKET\_GAP\_THRESHOLD'))]))")

=====

Stage:-----22

MAT node: tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing84

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: cp\_assisted\_multicriteria\_upstream\_policy\_routing84

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 84), ('column', 69),  
(('source\_fragment', '(bit<32>)local\_metadata.egr\_rate\_based\_path'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 84), ('column', 18),  
(('source\_fragment', 'egress\_queue\_rate\_value\_map.read(path\_rate\_status,  
(bit<32>)local\_metadata.egr\_rate\_based\_path'))]))



MAT node: tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_old\_port

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_old\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 35), ('column', 6),  
(('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.flowlet\_last\_used\_path'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 37), ('column', 8),  
(('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1'))])

MAT node: tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing101

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: cp\_assisted\_multicriteria\_upstream\_policy\_routing101

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 101), ('column', 70),  
(('source\_fragment', '(bit<32>)local\_metadata.egr\_queue\_based\_path'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 101), ('column', 19),  
(('source\_fragment', 'egress\_queue\_rate\_value\_map.read(path\_rate\_status,  
(bit<32>)local\_metadata.egr\_queue\_based\_path'))])

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_old\_port\_0

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_old\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 35), ('column', 6),  
('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.flowlet\_last\_used\_path']))

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 37), ('column', 8),  
('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1']))

=====  
=====  
Stage:-----23

MAT node: node\_30

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 85), ('column', 20),  
('source\_fragment', 'path\_rate\_status == (bit<4>)GREEN']))")

MAT node: node\_41

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 102), ('column', 21),  
('source\_fragment', 'path\_rate\_status == (bit<4>)GREEN']))")

Stage:-----24

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_rate\_port

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_rate\_port

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 50), ('column', 10), ('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_rate\_based\_path')])

\*) OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 52), ('column', 10), ('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1')])

MAT node: node\_32

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 87), ('column', 26), ('source\_fragment', 'path\_rate\_status == (bit<4>)YELLOW')])")

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_depth\_port\_1

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_depth\_port

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 45), ('column', 9), ('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_queue\_based\_path')])

\*) OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 47), ('column', 9), ('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1')])

MAT node: node\_43

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 106), ('column', 26), ('source\_fragment', 'path\_rate\_status == (bit<4>YELLOW')])")

=====

Stage:-----25

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_depth\_port

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_depth\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 45), ('column', 9),  
('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_queue\_based\_path')])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 47), ('column', 9),  
('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1')])

MAT node: node\_34

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 89), ('column', 27),  
('source\_fragment', 'path\_rate\_status == (bit<4>)RED')])")

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_rate\_port\_0

Match Keys are:


\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_rate\_port

Primitives used in action are:

  
\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 50), ('column', 10),  
('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_rate\_based\_path')])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 52), ('column', 10),  
('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1')])

MAT node: node\_45

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 108), ('column', 28),  
('source\_fragment', 'path\_rate\_status == (bit<4>RED')])")

=====  
=====

Stage:-----26

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_  
depth\_port\_0

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egr  
ess\_queue\_depth\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 45), ('column', 9),  
('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_queue\_based\_path')])



\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 47), ('column', 9),  
(('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1'))])

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_rate\_port\_1

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_rate\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 50), ('column', 10),  
(('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_rate\_based\_path'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 52), ('column', 10),  
(('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1'))])

=====

Stage:-----27

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_update\_flowlet\_id

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.update\_flowlet\_id

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 32), ('column', 8),  
(('source\_fragment',  
'flowlet\_last\_used\_port.write((bit<32>)local\_metadata.flowlet\_map\_index,(bit<9>)standard\_metadata.egress\_spec ))')])

MAT node:  
tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_update\_flowlet\_id\_0

Match Keys are:

Actions are:

1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.update\_flowlet\_id

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 32), ('column', 8),  
(('source\_fragment',  
'flowlet\_last\_used\_port.write((bit<32>)local\_metadata.flowlet\_map\_index,(bit<9>)standard\_metadata.egress\_spec ))')])

MAT node:  
tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_update\_flowlet\_id\_1

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.update\_flowlet\_id





Primitives used in action are:

```
*) OrderedDict([('filename',
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 32), ('column', 8),
('source_fragment',
'flowlet_last_used_port.write((bit<32>)local_metadata.flowlet_map_index,(bit<9>)standard_metad
ata.egress_spec )'))]
```

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_update\_flowlet\_id\_2

Match Keys are:

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.update\_flowle  
t\_id

Primitives used in action are:

```
*) OrderedDict([('filename',
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 32), ('column', 8),
('source_fragment',
'flowlet_last_used_port.write((bit<32>)local_metadata.flowlet_map_index,(bit<9>)standard_metad
ata.egress_spec )'))]
```

```
=====
=====
```

Stage:-----28

total nodes in graph are 48

total nodes in stageWiseMatMap are 50

Longest path legnth for Egress pipeline is 26

Stagewise table and action mapping (with the stateful memory usage also) for PipelineID.EGRESS\_PIPELINE is following :

Stagewise table and action mapping for PipelineID.EGRESS\_PIPELINE is following :

```
=====
=====
```

Stage:-----0

This is a dummy stage to handle a dummy node in the TDG. Not really mapped to the hardware.  
So please skip it.

```
=====
=====
```

Stage:-----1

MAT node: node\_52

Match Keys are:

Actions are:

```
*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 285),
('column', 12), ('source_fragment', 'standard_metadata.instance_type ==
BMV2_V1MODEL_INSTANCE_TYPE_NORMAL')]))")
```

```
=====
=====
```

Stage:-----2

Stage:-----3

MAT node: node\_54

Match Keys are:

Actions are:

```
*) SourceInfo(srcinfo="OrderedDict([('filename',
'p4src/src/egress_queue_depth_monitor.p4'), ('line', 37), ('column', 15), ('source_fragment',
'standard_metadata.deq_qdepth >= (last_updated_deq_depth +
EGRESS_QUEUE_DEPTH_THRESHOLD)'))")
```

Stage:-----4

MAT node: node\_56

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

```
*) SourceInfo(srcinfo="OrderedDict([('filename',
'p4src/src/egress_queue_depth_monitor.p4'), ('line', 47), ('column', 21), ('source_fragment',
'standard_metadata.deq_qdepth < (last_updated_deq_depth -
EGRESS_QUEUE_DEPTH_THRESHOLD)'))")
```

Stage:-----5

MAT node: tbl\_egress\_queue\_depth\_monitor35

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: egress\_queue\_depth\_monitor35

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 35), ('column', 82), ('source\_fragment', '(bit<32>)standard\_metadata.egress\_spec')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 35), ('column', 12), ('source\_fragment', 'port\_last\_updated\_egress\_queue\_avg\_depth.read(last\_updated\_deq\_depth, (bit<32>)standard\_metadata.egress\_spec')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 36), ('column', 12), ('source\_fragment', 'local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_port = standard\_metadata.egress\_port')])

MAT node: tbl\_egress\_queue\_depth\_monitor40

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: egress\_queue\_depth\_monitor40

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 110), ('column', 45), ('source\_fragment', '1; //Indicates the event is being reported by this switch itself ...')])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 129), ('column', 41), ('source\_fragment', '11; //These 3 events notifies if a packet has seen change in delay by threshold ...')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 42), ('column', 16), ('source\_fragment', 'local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_data = (bit<48>)standard\_metadata.deq\_qdepth')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 43), ('column', 63), ('source\_fragment', '(bit<32>)standard\_metadata.egress\_spec')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 43), ('column', 16), ('source\_fragment', 'port\_last\_updated\_egress\_queue\_avg\_depth.write((bit<32>)standard\_metadata.egress\_spec, standard\_metadata.deq\_qdepth)')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 44), ('column', 16), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_depth = true')])

MAT node: tbl\_egress\_queue\_depth\_monitor48

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: egress\_queue\_depth\_monitor48

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 110), ('column', 45), ('source\_fragment', '1; //Indicates the event is being reported by this switch itself ...')])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 130), ('column', 41), ('source\_fragment', '12; ...')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 50), ('column', 16), ('source\_fragment', 'local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_data = (bit<48>)standard\_metadata.deq\_qdepth')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 51), ('column', 63), ('source\_fragment', '(bit<32>)standard\_metadata.egress\_spec')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 51), ('column', 16), ('source\_fragment', 'port\_last\_updated\_egress\_queue\_avg\_depth.write((bit<32>)standard\_metadata.egress\_spec, standard\_metadata.deq\_qdepth)')])

`*) OrderedDict([('filename', 'p4src/src/egress_queue_depth_monitor.p4'), ('line', 52), ('column', 16), ('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_egr_queue_depth = true')])`

MAT node: tbl\_egress\_queue\_depth\_monitor56

Match Keys are:

`*) 8 bit key for handling conditional of previous stage`

Actions are:

1 Action Name: egress\_queue\_depth\_monitor56

Primitives used in action are:

`*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 128), ('column', 41), ('source_fragment', '10; ...')])`

`*) OrderedDict([('filename', 'p4src/src/egress_queue_depth_monitor.p4'), ('line', 61), ('column', 16), ('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_egr_queue_depth = false')])`

=====

Stage:-----6

MAT node: EgressPipeImpl.egress\_rate\_monitor\_control\_block.egress\_rate\_monitor\_table

Match Keys are:

`*) standard_metadata.egress_port`

Actions are:

1 Action Name:

EgressPipeImpl.egress\_rate\_monitor\_control\_block.monitor\_outgoing\_flow

Primitives used in action are:

`*) OrderedDict([('filename', 'p4src/src/egress_rate_monitor.p4'), ('line', 20), ('column', 8), ('source_fragment', 'local_metadata.egress_rate_event_hdr.egress_rate_event_port = standard_metadata.egress_port')])`

=====

Stage:-----7

MAT node: node\_60

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/egress\_rate\_monitor.p4'), ('line', 34), ('column', 12), ('source\_fragment',  
'local\_metadata.egress\_rate\_event\_hdr.egress\_traffic\_color != local\_metadata.temp')]))"

=====

Stage:-----8

MAT node: tbl\_egress\_rate\_monitor35

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: egress\_rate\_monitor35


Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/egress\_rate\_monitor.p4'), ('line', 35),  
(('column', 12), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_rate =  
true'))])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 111), ('column', 49),  
(('source\_fragment', '2; //Indicates the event is being reported by a switch connected to switch  
who is reporting ...'))])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 133), ('column', 38),  
(('source\_fragment', '21; ...'))])

=====



Stage:-----9

MAT node: node\_62

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 28), ('column', 11), ('source\_fragment', 'hdr.tcp.syn\_control\_flag == FLAG\_1')]))"

=====

Stage:-----10

MAT node: node\_64

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 32), ('column', 18), ('source\_fragment', 'hdr.tcp.fin\_control\_flag == FLAG\_1')]))"

=====

Stage:-----11

MAT node: node\_66

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 37), ('column', 17), ('source\_fragment', 'hdr.ipv6.rate\_control\_applicable\_flag == 0')]))"





=====

Stage:-----12

MAT node: node\_67

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 38), ('column', 16), ('source\_fragment', '(local\_metadata.egress\_rate\_event\_hdr.egress\_traffic\_color >= (bit<32>) RED ) && (local\_metadata.ingress\_rate\_event\_hdr.ingress\_traffic\_color > (bit<32>) GREEN)'))")

MAT node: node\_77

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 59), ('column', 19), ('source\_fragment', '(hdr.tcp.ack\_control\_flag == FLAG\_1 ) && (hdr.ipv6.rate\_control\_applicable\_flag == 1)'))")

=====

Stage:-----13

MAT node: node\_71

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 45), ('column', 22), ('source\_fragment',

'(local\_metadata.egress\_rate\_event\_hdr.egress\_traffic\_color <= (bit<32>) YELLOW ) &&  
(local\_metadata.ingress\_rate\_event\_hdr.ingress\_traffic\_color<= (bit<32>) GREEN))']")

=====

Stage:-----14

MAT node: node\_69

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 40), ('column', 19), ('source\_fragment', 'local\_metadata.last\_seq\_no\_with\_rate\_control + SEQ\_NUMBER\_THRESHOLD\_FOR\_RATE\_CONTROL < hdr.tcp.seq\_no')]))")

MAT node: node\_72

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 46), ('column', 19), ('source\_fragment', 'local\_metadata.last\_seq\_no\_with\_rate\_control + SEQ\_NUMBER\_THRESHOLD\_FOR\_RATE\_CONTROL < hdr.tcp.seq\_no')]))")

MAT node: node\_74

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 52), ('column', 19), ('source\_fragment', 'local\_metadata.last\_seq\_no\_with\_rate\_control + SEQ\_NUMBER\_THRESHOLD\_FOR\_RATE\_CONTROL < hdr.tcp.seq\_no')]))")

=====

Stage:-----15

MAT node: tbl\_rate\_control41

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: rate\_control41

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 41), ('column', 20), ('source\_fragment', 'flowlet\_id\_to\_seq\_number\_of\_last\_rate\_control\_action\_map.write((bit<32>)local\_metadata.flowlet\_map\_index, hdr.tcp.seq\_no)'))]

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 140), ('column', 69), ('source\_fragment', '2; ...')])

\*) OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 43), ('column', 20), ('source\_fragment', 'hdr.ipv6.rate\_control\_applicable\_flag = 0')])

MAT node: tbl\_rate\_control47

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: rate\_control47

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 141), ('column', 69), ('source\_fragment', '3; ...')])

\*) OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 48), ('column', 20), ('source\_fragment', 'flowlet\_id\_to\_seq\_number\_of\_last\_rate\_control\_action\_map.write((bit<32>)local\_metadata.flowlet\_map\_index, hdr.tcp.seq\_no)'))]

\*) OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 49), ('column', 20), ('source\_fragment', 'hdr.ipv6.rate\_control\_applicable\_flag = 0')])

MAT node: tbl\_rate\_control53

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: rate\_control53

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 53), ('column', 20), ('source\_fragment', 'hdr.ipv6.rate\_control\_applicable\_flag = 0')])

MAT node: tbl\_rate\_control55

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: rate\_control55

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 55), ('column', 20), ('source\_fragment', 'hdr.ipv6.rate\_control\_applicable\_flag = 0')])

MAT node: tbl\_rate\_control61

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: rate\_control61

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 61), ('column', 12), ('source\_fragment', 'flowlet\_id\_to\_seq\_number\_of\_last\_rate\_control\_action\_map.write((bit<32>)local\_metadata.flowlet\_map\_index, hdr.tcp.seq\_no)')])

MAT node: tbl\_rate\_control29

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: rate\_control29

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 29), ('column', 12), ('source\_fragment', 'flowlet\_id\_to\_seq\_number\_of\_last\_rate\_control\_action\_map.write((bit<32>)(local\_metadata.flowlet\_map\_index), hdr.tcp.seq\_no)')])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 139), ('column', 50), ('source\_fragment', '1; ...')])

MAT node: tbl\_rate\_control33

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: rate\_control33

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 33), ('column', 12), ('source\_fragment', 'flowlet\_id\_to\_seq\_number\_of\_last\_rate\_control\_action\_map.write((bit<32>)local\_metadata.flowlet\_map\_index, 0)')])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 139), ('column', 50), ('source\_fragment', '1; ...')])

MAT node: tbl\_rate\_control39

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: rate\_control39

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 39), ('column', 16), ('source\_fragment', 'flowlet\_id\_to\_seq\_number\_of\_last\_rate\_control\_action\_map.read(local\_metadata.last\_seq\_no\_with\_rate\_control, (bit<32>)local\_metadata.flowlet\_map\_index'))])

Stage:-----16

MAT node: node\_79

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 295), ('column', 16), ('source\_fragment', 'local\_metadata.is\_multicast')])")

Stage:-----17

MAT node: tbl\_leaf296

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: leaf296

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 296), ('column', 16), ('source\_fragment', 'exit')])

=====

Stage:-----18

MAT node: node\_81

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 305), ('column', 15), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_depth || local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_rate')]))")

=====

Stage:-----19

MAT node: tbl\_leaf307

Match Keys are:

\*) 8 bit key for handling conditional of previous stage


Actions are:

1 Action Name: leaf307

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 307), ('column', 38), ('source\_fragment', '(bit<32>)(standard\_metadata.ingress\_port)+((bit<32>)MAX\_PORTS\_IN\_SWITCH \* 2)'))

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 307), ('column', 16), ('source\_fragment', 'clone3(CloneType.E2E, (bit<32>)(standard\_metadata.ingress\_port)+((bit<32>)MAX\_PORTS\_IN\_SWITCH \* 2), {standard\_metadata, local\_metadata}'))]

  
MAT node: node\_83

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 309), ('column', 21), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_depth')]))"

=====  
=====  
Stage:-----20

MAT node: tbl\_leaf311

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: leaf311

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 311), ('column', 16), ('source\_fragment', 'clone3(CloneType.E2E, 255, {standard\_metadata, local\_metadata})')])

=====  
=====  
Stage:-----21

MAT node: node\_85

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:



\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 74), ('column', 50), ('source\_fragment', '0; ...')]))"

=====

Stage:-----22

MAT node: node\_86

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 326), ('column', 12), ('source\_fragment', 'standard\_metadata.egress\_port == 0')]))"

=====

Stage:-----23

MAT node: tbl\_set\_all\_header\_invalid

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:


1 Action Name: EgressPipeImpl.set\_all\_header\_invalid

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 220), ('column', 8), ('source\_fragment', 'hdr.packet\_out.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 223), ('column', 8), ('source\_fragment', 'hdr.ipv4.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 224), ('column', 8), ('source\_fragment', 'hdr.ipv6.setInvalid()')])

  
\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 225), ('column', 8), ('source\_fragment', 'hdr.tcp.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 226), ('column', 8), ('source\_fragment', 'hdr.udp.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 227), ('column', 8), ('source\_fragment', 'hdr.icmpv6.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 228), ('column', 8), ('source\_fragment', 'hdr.ndp.setInvalid()')])

MAT node: node\_89

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 347), ('column', 18), ('source\_fragment', 'standard\_metadata.egress\_port == 255')])")

=====

Stage:-----24

MAT node: tbl\_leaf329

Match Keys are:


Actions are:

1 Action Name: leaf329

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 329), ('column', 12), ('source\_fragment', 'hdr.ethernet.ether\_type = 0')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 330), ('column', 12), ('source\_fragment', 'hdr.packet\_in.setValid()')])

  
\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 332), ('column', 12), ('source\_fragment', 'hdr.packet\_in.ingress\_port = standard\_metadata.ingress\_port')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 336), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 337), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event\_data = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_data')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 338), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event\_port = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_port')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 340), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_traffic\_color = local\_metadata.egress\_rate\_event\_hdr.egress\_traffic\_color')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 341), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_rate\_event\_data = local\_metadata.egress\_rate\_event\_hdr.egress\_rate\_event\_data')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 342), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_rate\_event\_port = local\_metadata.egress\_rate\_event\_hdr.egress\_rate\_event\_port')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 345), ('column', 12), ('source\_fragment', 'recirculate<parsed\_headers\_t>(hdr)')])

MAT node: tbl\_set\_all\_header\_invalid\_0

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: EgressPipeImpl.set\_all\_header\_invalid

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 220), ('column', 8), ('source\_fragment', 'hdr.packet\_out.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 223), ('column', 8), ('source\_fragment', 'hdr.ipv4.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 224), ('column', 8), ('source\_fragment', 'hdr.ipv6.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 225), ('column', 8), ('source\_fragment', 'hdr.tcp.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 226), ('column', 8), ('source\_fragment', 'hdr.udp.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 227), ('column', 8), ('source\_fragment', 'hdr.icmpv6.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 228), ('column', 8), ('source\_fragment', 'hdr.ndp.setInvalid()')])

MAT node: node\_92

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 373), ('column', 16), ('source\_fragment', 'local\_metadata.rate\_control\_event == RATE\_INCREASE\_EVENT\_NEED\_TO\_BE\_APPLIED\_IN\_THIS\_SWITCH')])")

=====

Stage:-----25

MAT node: tbl\_leaf353

Match Keys are:

Actions are:

1 Action Name: leaf353

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 353), ('column', 12), ('source\_fragment', 'hdr.packet\_in.ingress\_port = standard\_metadata.ingress\_port')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 357), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 358), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event\_data = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_data')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 359), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event\_port = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_port')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 361), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_traffic\_color = local\_metadata.egress\_rate\_event\_hdr.egress\_traffic\_color')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 362), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_rate\_event\_data = local\_metadata.egress\_rate\_event\_hdr.egress\_rate\_event\_data')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 363), ('column', 12), ('source\_fragment', 'hdr.packet\_in.egress\_rate\_event\_port = local\_metadata.egress\_rate\_event\_hdr.egress\_rate\_event\_port')])

MAT node: tbl\_build\_fake\_ack\_only\_for\_increase

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: EgressPipeImpl.build\_fake\_ack\_only\_for\_increase

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 46), ('column', 22), ('source\_fragment', '1; ...')])

```

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 252), ('column', 8),
('source_fragment', 'bit<128> temp_src_addr = hdr.ipv6.src_addr;')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 253), ('column', 8),
('source_fragment', 'hdr.ipv6.src_addr = hdr.ipv6.dst_addr')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 254), ('column', 8),
('source_fragment', 'hdr.ipv6.dst_addr = temp_src_addr')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 255), ('column', 8),
('source_fragment', 'hdr.ipv6.payload_len = 20')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 257), ('column', 8),
('source_fragment', 'bit<16> temp_src_port = hdr.tcp.src_port;')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 258), ('column', 8),
('source_fragment', 'hdr.tcp.src_port = hdr.tcp.dst_port')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 259), ('column', 8),
('source_fragment', 'hdr.tcp.dst_port = temp_src_port')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 260), ('column', 8),
('source_fragment', 'bit<32> temp_ack_no = hdr.tcp.ack_no;')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 261), ('column', 8),
('source_fragment', 'hdr.tcp.ack_no = hdr.tcp.seq_no + 1')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 262), ('column', 8),
('source_fragment', 'hdr.tcp.seq_no = temp_ack_no')])

*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 265), ('column', 8),
('source_fragment', 'hdr.tcp.window = hdr.tcp.window + new_window')])

```

MAT node: node\_94

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

```

*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 375),
('column', 22), ('source_fragment', 'local_metadata.rate_control_event ==
RATE_DECREASE_EVENT_NEED_TO_BE_APPLIED_IN_THIS_SWITCH')])")

```

Stage:-----26

MAT node: tbl\_build\_fake\_ack\_only

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: EgressPipeImpl.build\_fake\_ack\_only

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 46), ('column', 22), ('source\_fragment', '1; ...')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 234), ('column', 8), ('source\_fragment', 'bit<128> temp\_src\_addr = hdr.ipv6.src\_addr;')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 235), ('column', 8), ('source\_fragment', 'hdr.ipv6.src\_addr = hdr.ipv6.dst\_addr')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 236), ('column', 8), ('source\_fragment', 'hdr.ipv6.dst\_addr = temp\_src\_addr')])


\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 237), ('column', 8), ('source\_fragment', 'hdr.ipv6.payload\_len = 20')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 239), ('column', 8), ('source\_fragment', 'bit<16> temp\_src\_port = hdr.tcp.src\_port;')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 240), ('column', 8), ('source\_fragment', 'hdr.tcp.src\_port = hdr.tcp.dst\_port')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 241), ('column', 8), ('source\_fragment', 'hdr.tcp.dst\_port = temp\_src\_port')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 242), ('column', 8), ('source\_fragment', 'bit<32> temp\_ack\_no = hdr.tcp.ack\_no;')])

  
\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 243), ('column', 8), ('source\_fragment', 'hdr.tcp.ack\_no = hdr.tcp.seq\_no + 1')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 244), ('column', 8), ('source\_fragment', 'hdr.tcp.seq\_no = temp\_ack\_no')])

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 247), ('column', 8), ('source\_fragment', 'hdr.tcp.window = hdr.tcp.window - new\_window')])

MAT node: node\_96

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 378), ('column', 19), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_packet\_from\_downstream\_port')])")

=====  
=====

Stage:-----27

MAT node: tbl\_leaf379

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: leaf379

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/leaf.p4'), ('line', 379), ('column', 20), ('source\_fragment', 'mark\_to\_drop(standard\_metadata)')])

=====  
=====

Stage:-----28



## Summary of resource usage in each stage is following

Total Resource usage in stage -- 1 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 1 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 2 totalBitWidthOfFieldsUsedAsParameter = 80

Maximum bitwidth of the actions used for Ingress Stage = 16

Maximum bitwidth of the actions used for Egress Stage = 64

Total unused action key bitwidth = 1200

Action Key field statistics for graph drawing 1 16 64 1200

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 2 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 16

Total bit width of the MAT Keys for egress stage = 0



Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 2 16 0 1264

totalNumberOfFieldsBeingModified = 2 headerBitWidthOfFieldsBeingModified = 24

totalNumberOfFieldsUsedAsParameter = 3 totalBitWidthOfFieldsUsedAsParameter = 96

Maximum bitwidth of the actions used for Ingress Stage = 96

Maximum bitwidth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1184

Action Key field statistics for graph drawing 2 96 0 1184

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is  
['standard\_metadata.instance\_type']

Total Resource usage in stage -- 3 is following

Total number of fields used as key for MAT = 1

Total bit width of the MAT Keys for ingress stage = 8

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1272


Mat key statistics Values for graph drawing 3 8 0 1272

totalNumberOfFieldsBeingModified = 22 headerBitWidthOfFieldsBeingModified = 328

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 48

Maximum bitwidth of the actions used for Ingress Stage = 328

Maximum bitwidth of the actions used for Egress Stage = 48



Total unused action key bitwidth = 904

Action Key field statistics for graph drawing 3 328 48 904

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 4 is following

Total number of fields used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 24

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1248

Mat key statistics Values for graph drawing 4 24 8 1248

totalNumberOfFieldsBeingModified = 6 headerBitWidthOfFieldsBeingModified = 48

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 48

Maximum bitwidth of the actions used for Ingress Stage = 48

Maximum bitwidth of the actions used for Egress Stage = 48

Total unused action key bitwidth = 1184

Action Key field statistics for graph drawing 4 48 48 1184

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

■

Total Resource usage in stage -- 5 is following

Total number of fields used as key for MAT = 4

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 32

Total unused MAT Key bitwidth = 1248

Mat key statistics Values for graph drawing 5 0 32 1248

totalNumberOfFieldsBeingModified = 17 headerBitWidthOfFieldsBeingModified = 344

totalNumberOfFieldsUsedAsParameter = 8 totalBitWidthOfFieldsUsedAsParameter = 192

Maximum bitwidth of the actions used for Ingress Stage = 32

Maximum bitwidth of the actions used for Egress Stage = 184

Total unused action key bitwidth = 1064

Action Key field statistics for graph drawing 5 32 184 1064

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Following Register Array from egress portion of pipeline is mapped to this stage

Register Array

name:EgressPipeImpl.egress\_queue\_depth\_monitor\_control\_block.port\_last\_updated\_egress\_queue\_avg\_depth

Total Resource usage in stage -- 6 is following

Total number of fields used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 136

Total bit width of the MAT Keys for egress stage = 16



Total unused MAT Key bitwidth = 1128

Mat key statistics Values for graph drawing 6 136 16 1128

totalNumberOfFieldsBeingModified = 15 headerBitWidthOfFieldsBeingModified = 520

totalNumberOfFieldsUsedAsParameter = 4 totalBitWidthOfFieldsUsedAsParameter = 288

Maximum bitwidth of the actions used for Ingress Stage = 776

Maximum bitwidth of the actions used for Egress Stage = 32

Total unused action key bitwidth = 472

Action Key field statistics for graph drawing 6 776 32 472

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 7 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 7 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 2 totalBitWidthOfFieldsUsedAsParameter = 128

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 128

Total unused action key bitwidth = 1152

Action Key field statistics for graph drawing 7 0 128 1152

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 8 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 8

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 8 8 8 1264

totalNumberOfFieldsBeingModified = 3 headerBitWidthOfFieldsBeingModified = 64

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 16

Maximum bitwdth of the actions used for Ingress Stage = 16

Maximum bitwdth of the actions used for Egress Stage = 64

Total unused action key bitwidth = 1200

Action Key field statistics for graph drawing 8 16 64 1200

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 9 is following



Total number of fields used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 56

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1216

Mat key statistics Values for graph drawing 9 56 8 1216

totalNumberOfFieldsBeingModified = 3 headerBitWidthOfFieldsBeingModified = 32

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 16

Maximum bitwidth of the actions used for Ingress Stage = 32

Maximum bitwidth of the actions used for Egress Stage = 16

Total unused action key bitwidth = 1232

Action Key field statistics for graph drawing 9 32 16 1232

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 10 is following

Total number of fields used as key for MAT = 1

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1272

Mat key statistics Values for graph drawing 10 0 8 1272

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 16

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 16

Total unused action key bitwidth = 1264

Action Key field statistics for graph drawing 10 0 16 1264

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 11 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 48

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1224

Mat key statistics Values for graph drawing 11 48 8 1224

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 16

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 16

Total unused action key bitwidth = 1264

Action Key field statistics for graph drawing 11 0 16 1264

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []



Total Resource usage in stage -- 12 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 12 0 16 1264

totalNumberOfFieldsBeingModified = 2 headerBitWidthOfFieldsBeingModified = 16

totalNumberOfFieldsUsedAsParameter = 4 totalBitWidthOfFieldsUsedAsParameter = 160

Maximum bitwidth of the actions used for Ingress Stage = 8

Maximum bitwidth of the actions used for Egress Stage = 160

Total unused action key bitwidth = 1112

Action Key field statistics for graph drawing 12 8 160 1112

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []


Total Resource usage in stage -- 13 is following

Total number of fields used as key for MAT = 1

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1272



Mat key statistics Values for graph drawing 13 0 8 1272

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 4 totalBitWidthOfFieldsUsedAsParameter = 160

Maximum bitwdth of the actions used for Ingress Stage = 32

Maximum bitwdth of the actions used for Egress Stage = 128

Total unused action key bitwidth = 1120

Action Key field statistics for graph drawing 13 32 128 1120

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 14 is following

Total number of fields used as key for MAT = 4

Total bit width of the MAT Keys for ingress stage = 136

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1128

Mat key statistics Values for graph drawing 14 136 16 1128

totalNumberOfFieldsBeingModified = 5 headerBitWidthOfFieldsBeingModified = 128

totalNumberOfFieldsUsedAsParameter = 4 totalBitWidthOfFieldsUsedAsParameter = 240

Maximum bitwdth of the actions used for Ingress Stage = 176

Maximum bitwdth of the actions used for Egress Stage = 192

Total unused action key bitwidth = 912

Action Key field statistics for graph drawing 14 176 192 912

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 15 is following

Total number of fields used as key for MAT = 8

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 64

Total unused MAT Key bitwidth = 1216

Mat key statistics Values for graph drawing 15 0 64 1216

totalNumberOfFieldsBeingModified = 14 headerBitWidthOfFieldsBeingModified = 256

totalNumberOfFieldsUsedAsParameter = 11 totalBitWidthOfFieldsUsedAsParameter = 336

Maximum bitwidth of the actions used for Ingress Stage = 16

Maximum bitwidth of the actions used for Egress Stage = 112

Total unused action key bitwidth = 1152

Action Key field statistics for graph drawing 15 16 112 1152

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Following Register Array from egress portion of pipeline is mapped to this stage

Register Array

name:EgressPipeImpl.leaf\_rate\_control\_processor\_control\_block.flowlet\_id\_to\_seq\_number\_of  
\_last\_rate\_control\_action\_map

Total Resource usage in stage -- 16 is following

Total number of fields used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 16

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1256

Mat key statistics Values for graph drawing 16 16 8 1256

totalNumberOfFieldsBeingModified = 3 headerBitWidthOfFieldsBeingModified = 24

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 16

Maximum bitwidth of the actions used for Ingress Stage = 16

Maximum bitwidth of the actions used for Egress Stage = 16

Total unused action key bitwidth = 1248

Action Key field statistics for graph drawing 16 16 16 1248

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 17 is following

Total number of fields used as key for MAT = 4

Total bit width of the MAT Keys for ingress stage = 160

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1112

Mat key statistics Values for graph drawing 17 160 8 1112

totalNumberOfFieldsBeingModified = 4 headerBitWidthOfFieldsBeingModified = 48

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 16

Maximum bitwdth of the actions used for Ingress Stage = 48

Maximum bitwdth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1232

Action Key field statistics for graph drawing 17 48 0 1232

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 18 is following

Total number of fields used as key for MAT = 5

Total bit width of the MAT Keys for ingress stage = 168

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1104

Mat key statistics Values for graph drawing 18 168 8 1104

totalNumberOfFieldsBeingModified = 5 headerBitWidthOfFieldsBeingModified = 56

totalNumberOfFieldsUsedAsParameter = 2 totalBitWidthOfFieldsUsedAsParameter = 32

Maximum bitwdth of the actions used for Ingress Stage = 48

Maximum bitwdth of the actions used for Egress Stage = 32

Total unused action key bitwidth = 1200

Action Key field statistics for graph drawing 18 48 32 1200

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 19 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 19 0 16 1264

totalNumberOfFieldsBeingModified = 6 headerBitWidthOfFieldsBeingModified = 256

totalNumberOfFieldsUsedAsParameter = 5 totalBitWidthOfFieldsUsedAsParameter = 176

Maximum bitwidth of the actions used for Ingress Stage = 384

Maximum bitwidth of the actions used for Egress Stage = 32

Total unused action key bitwidth = 864

Action Key field statistics for graph drawing 19 384 32 864


Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Following Register Array from ingress portion of pipeline is mapped to this stage

Register Array

name:IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.flowlet\_  
lasttime\_map



Total Resource usage in stage -- 20 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 8

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 20 8 8 1264

totalNumberOfFieldsBeingModified = 4 headerBitWidthOfFieldsBeingModified = 56

totalNumberOfFieldsUsedAsParameter = 4 totalBitWidthOfFieldsUsedAsParameter = 72

Maximum bitwdth of the actions used for Ingress Stage = 96

Maximum bitwdth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1184

Action Key field statistics for graph drawing 20 96 0 1184

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 21 is following

Total number of fields used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 16

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1256

Mat key statistics Values for graph drawing 21 16 8 1256

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 3 totalBitWidthOfFieldsUsedAsParameter = 256

Maximum bitwidth of the actions used for Ingress Stage = 192

Maximum bitwidth of the actions used for Egress Stage = 64

Total unused action key bitwidth = 1024

Action Key field statistics for graph drawing 21 192 64 1024

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 22 is following

Total number of fields used as key for MAT = 5

Total bit width of the MAT Keys for ingress stage = 32

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1240

Mat key statistics Values for graph drawing 22 32 8 1240

totalnumberOfFieldsBeingModified = 8 headerBitWidthOfFieldsBeingModified = 128

totalNumberOfFieldsUsedAsParameter = 5 totalBitWidthOfFieldsUsedAsParameter = 128

Maximum bitwidth of the actions used for Ingress Stage = 72

Maximum bitwidth of the actions used for Egress Stage = 32

Total unused action key bitwidth = 1176

Action Key field statistics for graph drawing 22 72 32 1176

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []



Following Register Array from ingress portion of pipeline is mapped to this stage

Register Array name:egress\_queue\_rate\_value\_map

Total Resource usage in stage -- 23 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 23 0 16 1264

totalNumberOfFieldsBeingModified = 7 headerBitWidthOfFieldsBeingModified = 56

totalNumberOfFieldsUsedAsParameter = 3 totalBitWidthOfFieldsUsedAsParameter = 64

Maximum bitwdth of the actions used for Ingress Stage = 32

Maximum bitwdth of the actions used for Egress Stage = 56

Total unused action key bitwidth = 1192

Action Key field statistics for graph drawing 23 32 56 1192

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 24 is following

Total number of fields used as key for MAT = 6

Total bit width of the MAT Keys for ingress stage = 32

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1232

Mat key statistics Values for graph drawing 24 32 16 1232

totalNumberOfFieldsBeingModified = 20 headerBitWidthOfFieldsBeingModified = 312

totalNumberOfFieldsUsedAsParameter = 12 totalBitWidthOfFieldsUsedAsParameter = 264

Maximum bitwidth of the actions used for Ingress Stage = 64

Maximum bitwidth of the actions used for Egress Stage = 392

Total unused action key bitwidth = 824

Action Key field statistics for graph drawing 24 64 392 824

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 25 is following

Total number of fields used as key for MAT = 6

Total bit width of the MAT Keys for ingress stage = 32

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1232

Mat key statistics Values for graph drawing 25 32 16 1232

totalNumberOfFieldsBeingModified = 23 headerBitWidthOfFieldsBeingModified = 672

totalNumberOfFieldsUsedAsParameter = 20 totalBitWidthOfFieldsUsedAsParameter = 632

Maximum bitwidth of the actions used for Ingress Stage = 64

Maximum bitwidth of the actions used for Egress Stage = 808

Total unused action key bitwidth = 408

Action Key field statistics for graph drawing 25 64 808 408

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 26 is following

Total number of fields used as key for MAT = 4

Total bit width of the MAT Keys for ingress stage = 16

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1248

Mat key statistics Values for graph drawing 26 16 16 1248

totalNumberOfFieldsBeingModified = 16 headerBitWidthOfFieldsBeingModified = 488

totalNumberOfFieldsUsedAsParameter = 11 totalBitWidthOfFieldsUsedAsParameter = 416

Maximum bitwidth of the actions used for Ingress Stage = 40

Maximum bitwidth of the actions used for Egress Stage = 808

Total unused action key bitwidth = 432

Action Key field statistics for graph drawing 26 40 808 432

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 27 is following

Total number of fields used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 16

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1256

Mat key statistics Values for graph drawing 27 16 8 1256

totalNumberOfFieldsBeingModified = 5 headerBitWidthOfFieldsBeingModified = 72

totalNumberOfFieldsUsedAsParameter = 8 totalBitWidthOfFieldsUsedAsParameter = 192

Maximum bitwidth of the actions used for Ingress Stage = 64

Maximum bitwidth of the actions used for Egress Stage = 8

Total unused action key bitwidth = 1208

Action Key field statistics for graph drawing 27 64 8 1208

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Following Register Array from ingress portion of pipeline is mapped to this stage

Register Array

name:IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.flowlet\_  
last\_used\_port

Total Resource usage in stage -- 28 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0



Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 28 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 28 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 29 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 29 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 0



Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 29 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 30 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 30 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwdth of the actions used for Ingress Stage = 0


Maximum bitwdth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 30 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []



Total Resource usage in stage -- 31 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 31 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwdth of the actions used for Ingress Stage = 0

Maximum bitwdth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 31 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 32 is following

Total number of fields used as key for MAT = 0


Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 32 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0



totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 32 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

3. The P4 program for non-leaf switch
4. The P4 program for non-leaf switch





# The P4 program for non-leaf switch

## Ingress stage header analysis

- 7) following fields are used as match fields in the Ingress stage
  - a) 'ethernet.dst\_addr',
  - b) 'ipv6.src\_addr',
  - c) 'ipv6.traffic\_class',
  - d) 'scalars.userMetadata.egr\_port\_rate\_value\_range',
  - e) 'scalars.userMetadata.minimum\_group\_members\_requirement',
  - f) 'scalars.userMetadata.egr\_queue\_depth\_value\_range',
  - g) 'standard\_metadata.ingress\_port',
  - h) 'ipv6.dst\_addr'
- 8) Besides them 62 header fields are used as action fields
- 9) 11 extra header fields were used to carry result of conditional matching results.

**The total length of the header fields in ingress stage is 1288**

Bitwidth wise header count is {8: 29, 32: 9, 16: 8, 48: 8, 128: 2}

## Egress stage header analysis

- 10) Following fields are used as match fields in the Ingress stage
  - a) 'standard\_metadata.egress\_port'
- 11) Besides them, 49 header fields are used as action fields
- 12) 12 extra header fields were used to carry the result of conditional matching results.

The total length of the header fields in egress stage is 1464

Bitwidth wise header count is {8: 10, 16: 13, 24: 1, 32: 14, 128: 4, 48: 4}

## Header Analysis Summary:

The total number of header fields used by the P4 program for leaf switches is 1288+1464.

This is well within the 4K bit wide header vector availability in RMT hardwares. Moreover, in production-grade compilers, multiple fields are merged together which reduces the header fields length further. So in real life the resource consumption by the leaf switch of P4 program will be even lower.

## Stagewise table and action mapping for PipelineID.INGRESS\_PIPELINE is follwoing :

```
=====
=====
```

Stage:-----0

This is a dummy stage to handle a dummy node in the TDG. Not really mapped to the hardware.  
So please skip it.

```
=====
=====
```

Stage:-----1

MAT node: node\_2

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 148), ('column', 8), ('source\_fragment', 'hdr.packet\_out.isValid())])")

```
=====
=====
```

Stage:-----2

MAT node: tbl\_spine150


Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Nanme: spine150

Primitives used in action are:

  
\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 150), ('column', 7), ('source\_fragment', 'standard\_metadata.egress\_spec = hdr.packet\_out.egress\_port')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 152), ('column', 7), ('source\_fragment', 'hdr.packet\_out.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 154), ('column', 7), ('source\_fragment', 'exit')])

MAT node: node\_4

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 155), ('column', 14), ('source\_fragment', 'hdr.packet\_in.isValid() && (standard\_metadata.instance\_type == BMV2\_V1MODEL\_INSTANCE\_TYPE\_RECIRC)'])")

=====

Stage:-----3

MAT node: tbl\_init\_pkt

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: IngressPipeImpl.init\_pkt

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 83), ('column', 8), ('source\_fragment', 'local\_metadata.egress\_queue\_event\_hdr.setValid()')])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 104), ('column', 23), ('source\_fragment', '0xF; ...')])

```

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 104), ('column', 23),
('source_fragment', '0xF; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 132), ('column', 40),
('source_fragment', '20; ...')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 88), ('column', 8),
('source_fragment', 'local_metadata.ingress_rate_event_hdr.setValid()')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 104), ('column', 23),
('source_fragment', '0xF; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 105), ('column', 21),
('source_fragment', '0x0; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 104), ('column', 23),
('source_fragment', '0xF; ...')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 93), ('column', 8),
('source_fragment', 'local_metadata.egress_rate_event_hdr.setValid()')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 104), ('column', 23),
('source_fragment', '0xF; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 105), ('column', 21),
('source_fragment', '0x0; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 104), ('column', 23),
('source_fragment', '0xF; ...')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 98), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.setValid()')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 102), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.downstream_routing_table_hit = false')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 103), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.do_l3_l2 = true')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 104), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_delay_proc = false')])

```

```

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 106), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_ing_queue_rate = false')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 107), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_ing_queue_depth = false')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 108), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_egr_queue_depth = false')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 109), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_control_pkt_from_egr_queue_rate = false')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 110), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_dp_only_multipart_algo_processing_required =
false')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 111), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.is_fake_ack_for_rate_ctrl_required = false')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 112), ('column', 8),
('source_fragment', 'local_metadata.minimum_group_members_requirement=1')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 114), ('column', 8),
('source_fragment', 'local_metadata.egr_queue_depth_value_range = 1')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 115), ('column', 8),
('source_fragment', 'local_metadata.egr_port_rate_value_range = 1')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 116), ('column', 33),
('source_fragment', '(bit<32>)standard_metadata.ingress_port')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 116), ('column', 8),
('source_fragment', 'ingressPortCounter.count((bit<32>)standard_metadata.ingress_port')])

```

```

=====
=====

```

Stage:-----4

MAT node:

IngressPipeImpl.ingress\_rate\_monitor\_control\_block.flow\_type\_based\_ingress\_stats\_table

Match Keys are:

\*) standard\_metadata.ingress\_port

\*) ipv6.traffic\_class

Actions are:

1 Action Name:

IngressPipeImpl.ingress\_rate\_monitor\_control\_block.monitor\_incoming\_flow\_based\_on\_flow\_type\_for\_pkts\_rcvd\_from\_upstream

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/ingress\_rate\_monitor.p4'), ('line', 73), ('column', 8), ('source\_fragment', 'flow\_type\_based\_ingress\_meter\_for\_upstream.execute\_meter((bit<32>)flow\_type\_based\_meter\_idx, local\_metadata.ingress\_rate\_event\_hdr.ingress\_traffic\_color)')])

\*) OrderedDict([('filename', 'p4src/src/ingress\_rate\_monitor.p4'), ('line', 74), ('column', 8), ('source\_fragment', 'local\_metadata.is\_pkt\_rcvd\_from\_downstream = false')])

\*) OrderedDict([('filename', 'p4src/src/ingress\_rate\_monitor.p4'), ('line', 76), ('column', 8), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_packet\_from\_downstream\_port = false')])

\*) OrderedDict([('filename', 'p4src/src/ingress\_rate\_monitor.p4'), ('line', 77), ('column', 8), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_packet\_from\_upstream\_port = true')])

1 Action Name:

IngressPipeImpl.ingress\_rate\_monitor\_control\_block.monitor\_incoming\_flow\_based\_on\_flow\_type\_for\_pkts\_rcvd\_from\_downstream

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/ingress\_rate\_monitor.p4'), ('line', 80), ('column', 8), ('source\_fragment', 'flow\_type\_based\_ingress\_meter\_for\_downstream.execute\_meter((bit<32>)flow\_type\_based\_meter\_idx, local\_metadata.ingress\_rate\_event\_hdr.ingress\_traffic\_color)')])

\*) OrderedDict([('filename', 'p4src/src/ingress\_rate\_monitor.p4'), ('line', 81), ('column', 8), ('source\_fragment', 'local\_metadata.is\_pkt\_rcvd\_from\_downstream = true')])



\*) OrderedDict([('filename', 'p4src/src/ingress\_rate\_monitor.p4'), ('line', 83), ('column', 8), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_packet\_from\_downstream\_port = true')])

\*) OrderedDict([('filename', 'p4src/src/ingress\_rate\_monitor.p4'), ('line', 84), ('column', 8), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_packet\_from\_upstream\_port = false')])

1 Action Name: NoAction

Primitives used in action are:

=====

Stage:-----5

MAT node: node\_8

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 166), ('column', 8), ('source\_fragment', '(hdr.icmpv6.type == ICMP6\_TYPE\_NS ) && (hdr.icmpv6.type == ICMP6\_TYPE\_NS)')])")

=====

Stage:-----6

MAT node: IngressPipelImpl.ndp\_processing\_control\_block.ndp\_reply\_table

Match Keys are:

\*) ipv6.src\_addr

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: IngressPipelImpl.ndp\_processing\_control\_block.ndp\_ns\_to\_na

Primitives used in action are:

```

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 15), ('column', 7),
('source_fragment', 'hdr.ethernet.src_addr = target_mac')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 53), ('column', 33),
('source_fragment', '0x33_33_00_00_00_01; ...')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 21), ('column', 7),
('source_fragment', 'hdr.ipv6.src_addr = hdr.ndp.target_ipv6_addr')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 21), ('column', 27),
('source_fragment', 'hdr.ndp.target_ipv6_addr; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 49), ('column', 31),
('source_fragment', '58; ...')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 56), ('column', 29),
('source_fragment', '136; ...')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 25), ('column', 7),
('source_fragment', 'hdr.ndp.flags = NDP_FLAG_ROUTER | NDP_FLAG_OVERRIDE')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 57), ('column', 38),
('source_fragment', '2; ...')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 27), ('column', 7),
('source_fragment', 'hdr.ndp.length = 1')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 28), ('column', 7),
('source_fragment', 'hdr.ndp.target_mac_addr = target_mac')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 29), ('column', 7),
('source_fragment', 'standard_metadata.egress_spec = standard_metadata.ingress_port')])


*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 30), ('column', 7),
('source_fragment', 'local_metadata.flag_hdr.do_I3_I2 = false')])

*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 31), ('column', 7),
('source_fragment', 'local_metadata.flag_hdr.is_pkt_toward_host = true')])

```

1 Action Name: IngressPipelImpl.ndp\_processing\_control\_block.ndp\_default\_action

Primitives used in action are:

  
\*) OrderedDict([('filename', 'p4src/src/ndp.p4'), ('line', 34), ('column', 10),  
('source\_fragment', 'local\_metadata.flag\_hdr.do\_l3\_l2 = true')])

=====

Stage:-----7

MAT node: tbl\_spine168

Match Keys are:

Actions are:

1 Action Name: spine168

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 168), ('column', 7),  
('source\_fragment', 'exit')])

=====

Stage:-----8

MAT node: node\_11

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 170),  
('column', 8), ('source\_fragment', 'local\_metadata.flag\_hdr.do\_l3\_l2')])")

=====

Stage:-----9

MAT node: IngressPipeImpl.l2\_ternary\_processing\_control\_block.l2\_ternary\_table

Match Keys are:

\*) ethernet.dst\_addr

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipelImpl.I2\_ternary\_processing\_control\_block.set\_multicast\_group

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/I2\_ternary.p4'), ('line', 26), ('column', 7), ('source\_fragment', 'standard\_metadata.mcast\_grp = gid')])

\*) OrderedDict([('filename', 'p4src/src/I2\_ternary.p4'), ('line', 27), ('column', 7), ('source\_fragment', 'local\_metadata.is\_multicast = true')])

1 Action Name: IngressPipelImpl.I2\_ternary\_processing\_control\_block.drop

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/I2\_ternary.p4'), ('line', 18), ('column', 7), ('source\_fragment', 'mark\_to\_drop(standard\_metadata)')])

=====

Stage:-----10

MAT node: tbl\_I2\_ternary44

Match Keys are:

Actions are:

1 Action Name: I2\_ternary44

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/I2\_ternary.p4'), ('line', 44), ('column', 12), ('source\_fragment', 'exit')])

=====

Stage:-----11

MAT node: node\_14

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 173), ('column', 12), ('source\_fragment', 'hdr.ipv6.isValid())'])")

=====

Stage:-----12

MAT node:

IngressPipeImpl.spine\_downstream\_routing\_control\_block.downstream\_routing\_table

Match Keys are:

\*) ipv6.dst\_addr

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.spine\_downstream\_routing\_control\_block.set\_downstream\_egress\_port

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine\_downstream\_routing.p4'), ('line', 18), ('column', 10), ('source\_fragment', 'standard\_metadata.egress\_spec = port\_num')])

\*) OrderedDict([('filename', 'p4src/src/spine\_downstream\_routing.p4'), ('line', 19), ('column', 10), ('source\_fragment', 'hdr.ethernet.src\_addr = hdr.ethernet.dst\_addr')])

\*) OrderedDict([('filename', 'p4src/src/spine\_downstream\_routing.p4'), ('line', 20), ('column', 10), ('source\_fragment', 'hdr.ethernet.dst\_addr = dmac')])

\*) OrderedDict([('filename', 'p4src/src/spine\_downstream\_routing.p4'), ('line', 22), ('column', 10), ('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1')])

\*) OrderedDict([('filename', 'p4src/src/spine\_downstream\_routing.p4'), ('line', 24), ('column', 10), ('source\_fragment', 'local\_metadata.flag\_hdr.downstream\_routing\_table\_hit = true')])

1 Action Name: NoAction

Primitives used in action are:

=====

Stage:-----13

MAT node: node\_16

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 175), ('column', 15), ('source\_fragment', 'local\_metadata.flag\_hdr.downstream\_routing\_table\_hit')])")

=====

Stage:-----14

MAT node: node\_17

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 177), ('column', 19), ('source\_fragment', 'hdr.ipv6.hop\_limit == 0')])")

MAT node: tbl\_spine180

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: spine180



Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 180), ('column', 12), ('source\_fragment', 'local\_metadata.flag\_hdr.found\_multi\_criteria\_paths = true')])

=====

Stage:-----15

MAT node: tbl\_drop

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: IngressPipelImpl.drop

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 122), ('column', 7), ('source\_fragment', 'mark\_to\_drop(standard\_metadata)')])

MAT node:

IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_queue\_depth\_based\_upstream\_path\_table

Match Keys are:

\*) ipv6.dst\_addr

\*) scalars.userMetadata.egr\_queue\_depth\_value\_range

\*) scalars.userMetadata.minimum\_group\_members\_requirement

Actions are:

1 Action Name:

IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_queue\_depth\_based\_upstream\_path\_selector\_action\_without\_param

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 60), ('column', 8),  
('source\_fragment', 'local\_metadata.egr\_queue\_based\_path = 0')])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 61), ('column', 8),  
('source\_fragment', 'local\_metadata.flag\_hdr.found\_egr\_queue\_depth\_based\_path = false')])

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_queue\_depth\_b  
ased\_upstream\_path\_selector\_action\_with\_param

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 65), ('column', 8),  
('source\_fragment', 'local\_metadata.egr\_queue\_based\_path = port\_num')])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 66), ('column', 8),  
('source\_fragment', 'local\_metadata.flag\_hdr.found\_egr\_queue\_depth\_based\_path = true')])

=====

Stage:-----16

MAT node:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_port\_rate\_based  
\_upstream\_path\_table

Match Keys are:

\*) ipv6.dst\_addr

\*) scalars.userMetadata.egr\_port\_rate\_value\_range

\*) scalars.userMetadata.minimum\_group\_members\_requirement

Actions are:



1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_port\_rate\_based\_upstream\_path\_selector\_action\_without\_param

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 106), ('column', 8),  
(('source\_fragment', 'local\_metadata.egr\_rate\_based\_path = 0'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 107), ('column', 8),  
(('source\_fragment', 'local\_metadata.flag\_hdr.found\_egr\_queue\_rate\_based\_path = false'))])

1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_routing\_control\_block.egr\_port\_rate\_based\_upstream\_path\_selector\_action\_with\_param

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 111), ('column', 8),  
(('source\_fragment', 'local\_metadata.egr\_rate\_based\_path = port\_num'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_routing\_tables.p4'), ('line', 112), ('column', 8),  
(('source\_fragment', 'local\_metadata.flag\_hdr.found\_egr\_queue\_rate\_based\_path = true'))])

=====

Stage:-----17

MAT node:  
tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_lookup\_flowlet\_id\_map

Match Keys are:

Actions are:

1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.lookup\_flowlet\_id\_map

Primitives used in action are:

```
*) OrderedDict([('filename',  
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 21), ('column', 8),  
('source_fragment', 'hash(local_metadata.flowlet_map_index, HashAlgorithm.crc16, (bit<13>)0, {  
hdr.ipv6.src_addr, hdr.ipv6.dst_addr, hdr.ipv6.flow_label, hdr.tcp.src_port, hdr.tcp.dst_port },  
(bit<13>)8191)'))])
```

```
*) OrderedDict([('filename',  
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 23), ('column', 8),  
('source_fragment', 'local_metadata.flow_inter_packet_gap =  
(bit<48>)standard_metadata.ingress_global_timestamp'))])
```

```
*) OrderedDict([('filename',  
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 24), ('column', 8),  
('source_fragment', 'flowlet_lasttime_map.read(local_metadata.flowlet_last_pkt_seen_time,  
(bit<32>)local_metadata.flowlet_map_index)'))])
```

```
*) OrderedDict([('filename',  
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 25), ('column', 8),  
('source_fragment', 'local_metadata.flow_inter_packet_gap =  
local_metadata.flow_inter_packet_gap - local_metadata.flowlet_last_pkt_seen_time'))])
```

```
*) OrderedDict([('filename',  
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 26), ('column', 8),  
('source_fragment', 'flowlet_lasttime_map.write((bit<32>)local_metadata.flowlet_map_index,  
standard_metadata.ingress_global_timestamp)'))])
```

```
=====
```

Stage:-----18

MAT node: tbl\_spine157

Match Keys are:

```
*) 8 bit key for handling conditional of previous stage
```

Actions are:

1 Action Name: spine157



Primitives used in action are:

```
*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 157), ('column', 8),
('source_fragment', 'local_metadata.flag_hdr.do_l3_l2 = false')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 158), ('column', 42),
('source_fragment', '(bit<32>)hdr.packet_in.egress_rate_event_port')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 158), ('column', 89),
('source_fragment', '(bit<4>)local_metadata.egress_rate_event_hdr.egress_traffic_color')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 158), ('column', 8),
('source_fragment',
'egress_queue_rate_value_map.write((bit<32>)hdr.packet_in.egress_rate_event_port,
(bit<4>)local_metadata.egress_rate_event_hdr.egress_traffic_color )')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 159), ('column', 8),
('source_fragment', 'mark_to_drop(standard_metadata)')])
```

MAT node: node\_23

Match Keys are:

Actions are:

```
*) SourceInfo(srcinfo="OrderedDict([('filename',
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 80), ('column', 12),
('source_fragment', '(hdr.ipv6.traffic_class == TRAFFIC_CLASS_HIGH_THROUGHPUT) ||
(hdr.ipv6.traffic_class == TRAFFIC_CLASS_HIGH_THROUGHPUT2)')])")
```

=====

Stage:-----19

MAT node: node\_24

Match Keys are:

```
*) 8 bit key for handling conditional of previous stage
```

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 82), ('column', 16),  
(('source\_fragment', 'local\_metadata.flow\_inter\_packet\_gap >  
FLOWLET\_INTER\_PACKET\_GAP\_THRESHOLD'))]))")

MAT node: node\_35

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 99), ('column', 17),  
(('source\_fragment', 'local\_metadata.flow\_inter\_packet\_gap >  
FLOWLET\_INTER\_PACKET\_GAP\_THRESHOLD'))]))")

=====

Stage:-----20

MAT node: tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing84

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: cp\_assisted\_multicriteria\_upstream\_policy\_routing84

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 84), ('column', 69),  
(('source\_fragment', '(bit<32>)local\_metadata.egr\_rate\_based\_path'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 84), ('column', 18),  
(('source\_fragment', 'egress\_queue\_rate\_value\_map.read(path\_rate\_status,  
(bit<32>)local\_metadata.egr\_rate\_based\_path'))))])

MAT node: tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_old\_port

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_old\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 35), ('column', 6),  
(('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.flowlet\_last\_used\_path'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 37), ('column', 8),  
(('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1'))])

MAT node: tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing101

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: cp\_assisted\_multicriteria\_upstream\_policy\_routing101

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 101), ('column', 70),  
(('source\_fragment', '(bit<32>)local\_metadata.egr\_queue\_based\_path'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 101), ('column', 19),  
(('source\_fragment', 'egress\_queue\_rate\_value\_map.read(path\_rate\_status,  
(bit<32>)local\_metadata.egr\_queue\_based\_path'))])

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_old\_port\_0

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_old\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 35), ('column', 6),  
('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.flowlet\_last\_used\_path']))

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 37), ('column', 8),  
('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1']))

Stage:-----21

MAT node: node\_26

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 85), ('column', 20),  
('source\_fragment', 'path\_rate\_status == (bit<4>)GREEN']))")

MAT node: node\_37

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 102), ('column', 21),  
('source\_fragment', 'path\_rate\_status == (bit<4>)GREEN']))")

Stage:-----22

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_rate\_port

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_rate\_port

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 50), ('column', 10), ('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_rate\_based\_path')])

\*) OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 52), ('column', 10), ('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1')])

MAT node: node\_28

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 87), ('column', 26), ('source\_fragment', 'path\_rate\_status == (bit<4>)YELLOW')])")

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_depth\_port\_1

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_depth\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 45), ('column', 9),  
('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_queue\_based\_path')])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 47), ('column', 9),  
('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1')])

MAT node: node\_39

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 106), ('column', 26),  
('source\_fragment', 'path\_rate\_status == (bit<4>)YELLOW')])")

=====

Stage:-----23

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_depth\_port

Match Keys are:

\*) 8 bit key for handling conditional of previous stage



Actions are:

1 Action Name:

IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_depth\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 45), ('column', 9),  
('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_queue\_based\_path')])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 47), ('column', 9),  
('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1')])

MAT node: node\_30

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 89), ('column', 27),  
('source\_fragment', 'path\_rate\_status == (bit<4>)RED')])")

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_rate\_port\_0

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_rate\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 50), ('column', 10),  
(('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_rate\_based\_path'))])

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 52), ('column', 10),  
(('source\_fragment', 'hdr.ipv6.hop\_limit = hdr.ipv6.hop\_limit - 1'))])

MAT node: node\_41

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 108), ('column', 28),  
(('source\_fragment', 'path\_rate\_status == (bit<4>)RED'))])")

=====

Stage:-----24

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_  
depth\_port\_0

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_depth\_port

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 45), ('column', 9),  
(('source\_fragment', 'standard\_metadata.egress\_spec = local\_metadata.egr\_queue\_based\_path'))])



```
*) OrderedDict([('filename',
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 47), ('column', 9),
('source_fragment', 'hdr.ipv6.hop_limit = hdr.ipv6.hop_limit - 1')])
```

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_use\_low\_egress\_queue\_rate\_port\_1

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:

IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.use\_low\_egress\_queue\_rate\_port

Primitives used in action are:

```
*) OrderedDict([('filename',
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 50), ('column', 10),
('source_fragment', 'standard_metadata.egress_spec = local_metadata.egr_rate_based_path')])
```

```
*) OrderedDict([('filename',
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 52), ('column', 10),
('source_fragment', 'hdr.ipv6.hop_limit = hdr.ipv6.hop_limit - 1')])
```

```
=====
=====
```

Stage:-----25

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_update\_flowlet\_id

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.update\_flowlet\_id

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 32), ('column', 8),  
('source\_fragment',  
'flowlet\_last\_used\_port.write((bit<32>)local\_metadata.flowlet\_map\_index,(bit<9>)standard\_metadata.egress\_spec )')])

MAT node:  
tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_update\_flowlet\_id\_0

Match Keys are:

Actions are:

1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.update\_flowlet\_id

Primitives used in action are:

\*) OrderedDict([('filename',  
'p4src/src/cp\_assisted\_multicriteria\_upstream\_policy\_routing.p4'), ('line', 32), ('column', 8),  
('source\_fragment',  
'flowlet\_last\_used\_port.write((bit<32>)local\_metadata.flowlet\_map\_index,(bit<9>)standard\_metadata.egress\_spec )')])

MAT node:  
tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_update\_flowlet\_id\_1

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name:  
IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.update\_flowlet\_id



Primitives used in action are:

```
*) OrderedDict([('filename',
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 32), ('column', 8),
('source_fragment',
'flowlet_last_used_port.write((bit<32>)local_metadata.flowlet_map_index,(bit<9>)standard_metad
ata.egress_spec )'))]
```

MAT node:

tbl\_cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block\_update\_flowlet\_id\_2

Match Keys are:

Actions are:

1 Action Name:

IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.update\_flowle  
t\_id

Primitives used in action are:

```
*) OrderedDict([('filename',
'p4src/src/cp_assisted_multicriteria_upstream_policy_routing.p4'), ('line', 32), ('column', 8),
('source_fragment',
'flowlet_last_used_port.write((bit<32>)local_metadata.flowlet_map_index,(bit<9>)standard_metad
ata.egress_spec )'))]
```

```
=====
=====
```

Stage:-----26

total nodes in graph are 44

total nodes in stageWiseMatMap are 46

Longest path length for Egress pipeline is 23

Stagewise table and action mapping for  
PipelineID.EGRESS\_PIPELINE is follwoing :

Stage:-----0

This is a dummy stage to handle a dummy node in the TDG. Not really mapped to the hardware.  
So please skip it.

Stage:-----1

MAT node: node\_48

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 298), ('column', 12), ('source\_fragment', 'standard\_metadata.instance\_type == BMV2\_V1MODEL\_INSTANCE\_TYPE\_NORMAL')]))")

Stage:-----2

Stage:-----3

MAT node: node\_50

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 37), ('column', 15), ('source\_fragment',

standard\_metadata.deq\_qdepth >= (last\_updated\_deq\_depth +  
EGRESS\_QUEUE\_DEPTH\_THRESHOLD)))]")

Stage:-----4

MAT node: node\_52

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename',  
'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 47), ('column', 21), ('source\_fragment',  
'standard\_metadata.deq\_qdepth < (last\_updated\_deq\_depth -  
EGRESS\_QUEUE\_DEPTH\_THRESHOLD)))]")

Stage:-----5

MAT node: tbl\_egress\_queue\_depth\_monitor35

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: egress\_queue\_depth\_monitor35

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line',  
35), ('column', 82), ('source\_fragment', '(bit<32>)standard\_metadata.egress\_spec')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line',  
35), ('column', 12), ('source\_fragment',  
'port\_last\_updated\_egress\_queue\_avg\_depth.read(last\_updated\_deq\_depth,  
(bit<32>)standard\_metadata.egress\_spec')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 36), ('column', 12), ('source\_fragment', 'local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_port = standard\_metadata.egress\_port')]))

MAT node: tbl\_egress\_queue\_depth\_monitor40

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: egress\_queue\_depth\_monitor40

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 110), ('column', 45), ('source\_fragment', '1; //Indicates the event is being reported by this switch itself ...')]))

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 129), ('column', 41), ('source\_fragment', '11; //These 3 events notifies if a packet has seen change in delay by threshold ...')]))

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 42), ('column', 16), ('source\_fragment', 'local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_data = (bit<48>)standard\_metadata.deq\_qdepth')]))

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 43), ('column', 63), ('source\_fragment', '(bit<32>)standard\_metadata.egress\_spec')]))

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 43), ('column', 16), ('source\_fragment', 'port\_last\_updated\_egress\_queue\_avg\_depth.write((bit<32>)standard\_metadata.egress\_spec, standard\_metadata.deq\_qdepth')]))

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 44), ('column', 16), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_depth = true')]))

MAT node: tbl\_egress\_queue\_depth\_monitor48



Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: egress\_queue\_depth\_monitor48

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 110), ('column', 45), ('source\_fragment', '1; //Indicates the event is being reported by this switch itself ...')])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 130), ('column', 41), ('source\_fragment', '12; ...')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 50), ('column', 16), ('source\_fragment', 'local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_data = (bit<48> standard\_metadata.deq\_qdepth')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 51), ('column', 63), ('source\_fragment', '(bit<32>)standard\_metadata.egress\_spec')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 51), ('column', 16), ('source\_fragment', 'port\_last\_updated\_egress\_queue\_avg\_depth.write((bit<32>)standard\_metadata.egress\_spec, standard\_metadata.deq\_qdepth')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 52), ('column', 16), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_depth = true')])

MAT node: tbl\_egress\_queue\_depth\_monitor56

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: egress\_queue\_depth\_monitor56

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 128), ('column', 41), ('source\_fragment', '10; ...')])

\*) OrderedDict([('filename', 'p4src/src/egress\_queue\_depth\_monitor.p4'), ('line', 61), ('column', 16), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_depth = false')])

=====

Stage:-----6

MAT node: EgressPipeImpl.egress\_rate\_monitor\_control\_block.egress\_rate\_monitor\_table

Match Keys are:

\*) standard\_metadata.egress\_port

Actions are:

1 Action Name:

EgressPipeImpl.egress\_rate\_monitor\_control\_block.monitor\_outgoing\_flow

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/egress\_rate\_monitor.p4'), ('line', 20), ('column', 8), ('source\_fragment', 'local\_metadata.egress\_rate\_event\_hdr.egress\_rate\_event\_port = standard\_metadata.egress\_port')])

=====

Stage:-----7

MAT node: node\_56

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/egress\_rate\_monitor.p4'), ('line', 34), ('column', 12), ('source\_fragment', 'local\_metadata.egress\_rate\_event\_hdr.egress\_traffic\_color != local\_metadata.temp')]))")

Stage:-----8

MAT node: tbl\_egress\_rate\_monitor35

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: egress\_rate\_monitor35

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/egress\_rate\_monitor.p4'), ('line', 35), ('column', 12), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_rate = true')])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 111), ('column', 49), ('source\_fragment', '2; //Indicates the event is being reported by a switch connected to switch who is reporting ...')])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 133), ('column', 38), ('source\_fragment', '21; ...')])

Stage:-----9

MAT node: node\_58

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 86), ('column', 12), ('source\_fragment', 'hdr.tcp.isValid() && (hdr.ipv6.rate\_control\_applicable\_flag == 0)'))")

=====

Stage:-----10

MAT node: node\_59

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 87), ('column', 16), ('source\_fragment', '(local\_metadata.egress\_rate\_event\_hdr.egress\_traffic\_color >= (bit<32>) YELLOW ) && (local\_metadata.ingress\_rate\_event\_hdr.ingress\_traffic\_color > (bit<32>) GREEN)'))")

=====

Stage:-----11

MAT node: tbl\_rate\_control88

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: rate\_control88

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 140), ('column', 69), ('source\_fragment', '2; ...')])

MAT node: node\_61

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/rate\_control.p4'), ('line', 91), ('column', 22), ('source\_fragment', '(local\_metadata.egress\_rate\_event\_hdr.egress\_traffic\_color < (bit<32>) YELLOW ) && (local\_metadata.ingress\_rate\_event\_hdr.ingress\_traffic\_color <= (bit<32>) GREEN)'))")

=====

Stage:-----12

MAT node: tbl\_rate\_control92

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: rate\_control92

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 141), ('column', 69), ('source\_fragment', '3; ...')])

=====

Stage:-----13

MAT node: node\_63

Match Keys are:


\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 308), ('column', 16), ('source\_fragment', 'local\_metadata.is\_multicast')])")

=====

Stage:-----14

  
MAT node: tbl\_spine309

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: spine309

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 309), ('column', 16), ('source\_fragment', 'exit')])

=====  
=====  
Stage:-----15

MAT node: node\_65

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 311), ('column', 15), ('source\_fragment', 'local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_depth || local\_metadata.flag\_hdr.is\_control\_pkt\_from\_egr\_queue\_rate')]))"

=====  
=====  
Stage:-----16

MAT node: tbl\_spine313

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: spine313

Primitives used in action are:

```
*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 313), ('column', 38),
('source_fragment', '(bit<32>)(standard_metadata.ingress_port)+
((bit<32>)MAX_PORTS_IN_SWITCH * 2)'))]
```

```
*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 313), ('column', 16),
('source_fragment', 'clone3(CloneType.E2E, (bit<32>)(standard_metadata.ingress_port)+
((bit<32>)MAX_PORTS_IN_SWITCH * 2), {standard_metadata, local_metadata}))']]
```

MAT node: node\_67

Match Keys are:

```
*) 8 bit key for handling conditional of previous stage
```

Actions are:

```
*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 315),
('column', 21), ('source_fragment',
'local_metadata.flag_hdr.is_control_pkt_from_egr_queue_depth')]))"
```

```
=====
=====
```

Stage:-----17

MAT node: tbl\_spine317

Match Keys are:

```
*) 8 bit key for handling conditional of previous stage
```

Actions are:

1 Action Name: spine317

Primitives used in action are:

```
*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 317), ('column', 16),
('source_fragment', 'clone3(CloneType.E2E, 255, {standard_metadata, local_metadata}))']]
```



=====

Stage:-----18

MAT node: node\_69

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 74), ('column', 50), ('source\_fragment', '0; ...')]))"

=====

Stage:-----19

MAT node: node\_70

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 325), ('column', 16), ('source\_fragment', 'standard\_metadata.egress\_port == 0')]))"

=====

Stage:-----20

MAT node: tbl\_set\_all\_header\_invalid

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: EgressPipelImpl.set\_all\_header\_invalid



=====

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 232), ('column', 8), ('source\_fragment', 'hdr.packet\_out.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 233), ('column', 8), ('source\_fragment', 'hdr.packet\_in.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 234), ('column', 8), ('source\_fragment', 'hdr.ethernet.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 235), ('column', 8), ('source\_fragment', 'hdr.ipv4.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 236), ('column', 8), ('source\_fragment', 'hdr.ipv6.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 237), ('column', 8), ('source\_fragment', 'hdr.tcp.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 238), ('column', 8), ('source\_fragment', 'hdr.udp.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 239), ('column', 8), ('source\_fragment', 'hdr.icmpv6.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 240), ('column', 8), ('source\_fragment', 'hdr.ndp.setInvalid()')])

MAT node: node\_73

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 349), ('column', 22), ('source\_fragment', 'standard\_metadata.egress\_port == 255')])")

=====

Stage:-----21

MAT node: tbl\_spine327

Match Keys are:

Actions are:

1 Action Name: spine327

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 327), ('column', 16), ('source\_fragment', 'hdr.ethernet.setValid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 328), ('column', 16), ('source\_fragment', 'hdr.ethernet.ether\_type = 0')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 329), ('column', 16), ('source\_fragment', 'hdr.packet\_in.setValid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 331), ('column', 16), ('source\_fragment', 'hdr.packet\_in.ingress\_port = standard\_metadata.ingress\_port')])


\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 334), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 335), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event\_data = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_data')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 336), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event\_port = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_port')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 342), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_traffic\_color = local\_metadata.egress\_rate\_event\_hdr.egress\_traffic\_color')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 344), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_rate\_event\_data = local\_metadata.egress\_rate\_event\_hdr.egress\_rate\_event\_data')])

  
\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 345), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_rate\_event\_port = local\_metadata.egress\_rate\_event\_hdr.egress\_rate\_event\_port')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 347), ('column', 16), ('source\_fragment', 'recirculate<parsed\_headers\_t>(hdr)')])

MAT node: tbl\_set\_all\_header\_invalid\_0

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: EgressPipeImpl.set\_all\_header\_invalid

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 232), ('column', 8), ('source\_fragment', 'hdr.packet\_out.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 233), ('column', 8), ('source\_fragment', 'hdr.packet\_in.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 234), ('column', 8), ('source\_fragment', 'hdr.ethernet.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 235), ('column', 8), ('source\_fragment', 'hdr.ipv4.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 236), ('column', 8), ('source\_fragment', 'hdr.ipv6.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 237), ('column', 8), ('source\_fragment', 'hdr.tcp.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 238), ('column', 8), ('source\_fragment', 'hdr.udp.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 239), ('column', 8), ('source\_fragment', 'hdr.icmpv6.setInvalid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 240), ('column', 8), ('source\_fragment', 'hdr.ndp.setInvalid()')])

MAT node: tbl\_spine372

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: spine372

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 372), ('column', 41), ('source\_fragment', '(bit<32>)standard\_metadata.egress\_port')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 372), ('column', 16), ('source\_fragment', 'p2pFeedbackCounter.count((bit<32>)standard\_metadata.egress\_port')])

=====

Stage:-----22

MAT node: tbl\_spine353

Match Keys are:

Actions are:


1 Action Name: spine353

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 353), ('column', 16), ('source\_fragment', 'hdr.packet\_in.setValid()')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 354), ('column', 16), ('source\_fragment', 'hdr.packet\_in.ingress\_port = standard\_metadata.ingress\_port')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 357), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event')])

  
\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 358), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event\_data = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_data')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 359), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_queue\_event\_port = local\_metadata.egress\_queue\_event\_hdr.egress\_queue\_event\_port')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 365), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_traffic\_color = local\_metadata.egress\_rate\_event\_hdr.egress\_traffic\_color')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 367), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_rate\_event\_data = local\_metadata.egress\_rate\_event\_hdr.egress\_rate\_event\_data')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 368), ('column', 16), ('source\_fragment', 'hdr.packet\_in.egress\_rate\_event\_port = local\_metadata.egress\_rate\_event\_hdr.egress\_rate\_event\_port')])

MAT node: node\_77

Match Keys are:

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 374), ('column', 20), ('source\_fragment', 'local\_metadata.rate\_control\_event == RATE\_DECREASE\_EVENT\_NEED\_TO\_BE\_APPLIED\_IN\_THIS\_SWITCH')])")

=====

Stage:-----23

MAT node: tbl\_build\_p2p\_feedback\_with\_fake\_ack

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: EgressPipelImpl.build\_p2p\_feedback\_with\_fake\_ack

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 46), ('column', 22), ('source\_fragment', '1; ...')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 246), ('column', 8), ('source\_fragment', 'bit<128> temp\_src\_addr = hdr.ipv6.src\_addr;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 247), ('column', 8), ('source\_fragment', 'hdr.ipv6.src\_addr = hdr.ipv6.dst\_addr;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 248), ('column', 8), ('source\_fragment', 'hdr.ipv6.dst\_addr = temp\_src\_addr;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 249), ('column', 8), ('source\_fragment', 'hdr.ipv6.payload\_len = 20;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 251), ('column', 8), ('source\_fragment', 'bit<16> temp\_src\_port = hdr.tcp.src\_port;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 252), ('column', 8), ('source\_fragment', 'hdr.tcp.src\_port = hdr.tcp.dst\_port;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 253), ('column', 8), ('source\_fragment', 'hdr.tcp.dst\_port = temp\_src\_port;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 254), ('column', 8), ('source\_fragment', 'bit<32> temp\_ack\_no = hdr.tcp.ack\_no;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 255), ('column', 8), ('source\_fragment', 'hdr.tcp.ack\_no = hdr.tcp.seq\_no + 1;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 256), ('column', 8), ('source\_fragment', 'hdr.tcp.seq\_no = temp\_ack\_no;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 259), ('column', 8), ('source\_fragment', 'hdr.tcp.window = hdr.tcp.window - new\_window;')])

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 139), ('column', 50), ('source\_fragment', '1; ...')])

MAT node: node\_79

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

\*) SourceInfo(srcinfo="OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 377), ('column', 25), ('source\_fragment', 'local\_metadata.rate\_control\_event == RATE\_INCREASE\_EVENT\_NEED\_TO\_BE\_APPLIED\_IN\_THIS\_SWITCH')]))")

=====

Stage:-----24

MAT node: tbl\_build\_p2p\_feedback\_with\_fake\_ack\_for\_increase

Match Keys are:

\*) 8 bit key for handling conditional of previous stage

Actions are:

1 Action Name: EgressPipeImpl.build\_p2p\_feedback\_with\_fake\_ack\_for\_increase

Primitives used in action are:

\*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 46), ('column', 22), ('source\_fragment', '1; ...')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 265), ('column', 12), ('source\_fragment', 'bit<128> temp\_src\_addr = hdr.ipv6.src\_addr;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 266), ('column', 12), ('source\_fragment', 'hdr.ipv6.src\_addr = hdr.ipv6.dst\_addr;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 267), ('column', 12), ('source\_fragment', 'hdr.ipv6.dst\_addr = temp\_src\_addr;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 268), ('column', 12), ('source\_fragment', 'hdr.ipv6.payload\_len = 20;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 270), ('column', 12), ('source\_fragment', 'bit<16> temp\_src\_port = hdr.tcp.src\_port;')])

\*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 271), ('column', 12), ('source\_fragment', 'hdr.tcp.src\_port = hdr.tcp.dst\_port;')])

```

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 272), ('column', 12),
('source_fragment', 'hdr.tcp.dst_port = temp_src_port')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 273), ('column', 12),
('source_fragment', 'bit<32> temp_ack_no = hdr.tcp.ack_no;')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 274), ('column', 12),
('source_fragment', 'hdr.tcp.ack_no = hdr.tcp.seq_no + 1')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 275), ('column', 12),
('source_fragment', 'hdr.tcp.seq_no = temp_ack_no')])

*) OrderedDict([('filename', 'p4src/src/spine.p4'), ('line', 278), ('column', 12),
('source_fragment', 'hdr.tcp.window = hdr.tcp.window + new_window')])

*) OrderedDict([('filename', 'p4src/src/CONSTANTS.p4'), ('line', 139), ('column', 50),
('source_fragment', '1; ...')])

=====
=====

```

## Summary of resource usage in each stage is following

Total Resource usage in stage -- 1 is follwoing

Total number of fileds used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 1 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 2 totalBitWidthOfFieldsUsedAsParameter = 80



Maximum bitwidth of the actions used for Ingress Stage = 16

Maximum bitwidth of the actions used for Egress Stage = 64

Total unused action key bitwidth = 1200

Action Key field statistics for graph drawing 1 16 64 1200

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 2 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 16

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 2 16 0 1264

totalNumberOfFieldsBeingModified = 2 headerBitWidthOfFieldsBeingModified = 24

totalNumberOfFieldsUsedAsParameter = 3 totalBitWidthOfFieldsUsedAsParameter = 96

Maximum bitwidth of the actions used for Ingress Stage = 96


Maximum bitwidth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1184

Action Key field statistics for graph drawing 2 96 0 1184

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is  
['standard\_metadata.instance\_type']



Total Resource usage in stage -- 3 is follwoing

Total number of fileds used as key for MAT = 1

Total bit width of the MAT Keys for ingress stage = 8

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1272

Mat key statistics Values for graph drawing 3 8 0 1272

totalNumberOfFieldsBeingModified = 26 headerBitWidthOfFieldsBeingModified = 424

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 48

Maximum bitwdth of the actions used for Ingress Stage = 424

Maximum bitwdth of the actions used for Egress Stage = 48

Total unused action key bitwidth = 808

Action Key field statistics for graph drawing 3 424 48 808

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 4 is follwoing

Total number of fileds used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 24

Total bit width of the MAT Keys for egress stage = 8



Total unused MAT Key bitwidth = 1248

Mat key statistics Values for graph drawing 4 24 8 1248

totalNumberOfFieldsBeingModified = 6 headerBitWidthOfFieldsBeingModified = 48

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 48

Maximum bitwidth of the actions used for Ingress Stage = 48

Maximum bitwidth of the actions used for Egress Stage = 48

Total unused action key bitwidth = 1184

Action Key field statistics for graph drawing 4 48 48 1184

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 5 is follwoing

Total number of fileds used as key for MAT = 4

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 32

Total unused MAT Key bitwidth = 1248

Mat key statistics Values for graph drawing 5 0 32 1248

totalNumberOfFieldsBeingModified = 17 headerBitWidthOfFieldsBeingModified = 344

totalNumberOfFieldsUsedAsParameter = 8 totalBitWidthOfFieldsUsedAsParameter = 192

Maximum bitwidth of the actions used for Ingress Stage = 32

Maximum bitwidth of the actions used for Egress Stage = 184

Total unused action key bitwidth = 1064

Action Key field statistics for graph drawing 5 32 184 1064

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Following Register Array from egress portion of pipeline is mapped to this stage

Register Array

name:EgressPipeImpl.egress\_queue\_depth\_monitor\_control\_block.port\_last\_updated\_egress\_q  
ueue\_avg\_depth

Total Resource usage in stage -- 6 is follwoing

Total number of fileds used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 136

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1128

Mat key statistics Values for graph drawing 6 136 16 1128

totalnumberOfFieldsBeingModified = 15 headerBitWidthOfFieldsBeingModified = 520

totalNumberOfFieldsUsedAsParameter = 4 totalBitWidthOfFieldsUsedAsParameter = 288

Maximum bitwdth of the actions used for Ingress Stage = 776

Maximum bitwdth of the actions used for Egress Stage = 32

Total unused action key bitwidth = 472

Action Key field statistics for graph drawing 6 776 32 472

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 7 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 7 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 2 totalBitWidthOfFieldsUsedAsParameter = 128

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 128

Total unused action key bitwidth = 1152

Action Key field statistics for graph drawing 7 0 128 1152

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []


Total Resource usage in stage -- 8 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 8

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1264



Mat key statistics Values for graph drawing 8 8 8 1264

totalNumberOfFieldsBeingModified = 3 headerBitWidthOfFieldsBeingModified = 64

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 16

Maximum bitwdth of the actions used for Ingress Stage = 16

Maximum bitwdth of the actions used for Egress Stage = 64

Total unused action key bitwidth = 1200

Action Key field statistics for graph drawing 8 16 64 1200

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 9 is follwoing

Total number of fileds used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 56

Total bit width of the MAT Keys for egress stage = 8

Total ununsed MAT Key bitwidth = 1216

Mat key statistics Values for graph drawing 9 56 8 1216

totalNumberOfFieldsBeingModified = 3 headerBitWidthOfFieldsBeingModified = 32


totalNumberOfFieldsUsedAsParameter = 2 totalBitWidthOfFieldsUsedAsParameter = 32

Maximum bitwdth of the actions used for Ingress Stage = 32

Maximum bitwdth of the actions used for Egress Stage = 32

Total unused action key bitwidth = 1216

Action Key field statistics for graph drawing 9 32 32 1216



Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 10 is follwoing

Total number of fileds used as key for MAT = 1

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 8

Total ununsed MAT Key bitwidth = 1272

Mat key statistics Values for graph drawing 10 0 8 1272

totalnumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 2 totalBitWidthOfFieldsUsedAsParameter = 128

Maximum bitwdth of the actions used for Ingress Stage = 0

Maximum bitwdth of the actions used for Egress Stage = 128

Total unused action key bitwidth = 1152

Action Key field statistics for graph drawing 10 0 128 1152

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 11 is follwoing

Total number of fileds used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 11 0 16 1264

totalNumberOfFieldsBeingModified = 1 headerBitWidthOfFieldsBeingModified = 8

totalNumberOfFieldsUsedAsParameter = 3 totalBitWidthOfFieldsUsedAsParameter = 144

Maximum bitwidth of the actions used for Ingress Stage = 16

Maximum bitwidth of the actions used for Egress Stage = 128

Total unused action key bitwidth = 1136

Action Key field statistics for graph drawing 11 16 128 1136

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 12 is following

Total number of fields used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 136

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1136

Mat key statistics Values for graph drawing 12 136 8 1136

totalNumberOfFieldsBeingModified = 6 headerBitWidthOfFieldsBeingModified = 136

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 48

Maximum bitwidth of the actions used for Ingress Stage = 176



Maximum bitwidth of the actions used for Egress Stage = 8

Total unused action key bitwidth = 1096

Action Key field statistics for graph drawing 12 176 8 1096

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 13 is following

Total number of fields used as key for MAT = 1

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1272

Mat key statistics Values for graph drawing 13 0 8 1272

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 2 totalBitWidthOfFieldsUsedAsParameter = 32

Maximum bitwidth of the actions used for Ingress Stage = 16

Maximum bitwidth of the actions used for Egress Stage = 16

Total unused action key bitwidth = 1248

Action Key field statistics for graph drawing 13 16 16 1248

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 14 is following

Total number of fields used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 16

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1256

Mat key statistics Values for graph drawing 14 16 8 1256

totalNumberOfFieldsBeingModified = 1 headerBitWidthOfFieldsBeingModified = 8

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 16

Maximum bitwidth of the actions used for Ingress Stage = 16

Maximum bitwidth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1264

Action Key field statistics for graph drawing 14 16 0 1264

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 15 is following

Total number of fields used as key for MAT = 5

Total bit width of the MAT Keys for ingress stage = 168

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1104

Mat key statistics Values for graph drawing 15 168 8 1104

totalNumberOfFieldsBeingModified = 5 headerBitWidthOfFieldsBeingModified = 56

totalNumberOfFieldsUsedAsParameter = 2 totalBitWidthOfFieldsUsedAsParameter = 32

Maximum bitwdth of the actions used for Ingress Stage = 48

Maximum bitwdth of the actions used for Egress Stage = 32

Total unused action key bitwidth = 1200

Action Key field statistics for graph drawing 15 48 32 1200

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 16 is follwoing

Total number of fileds used as key for MAT = 5

Total bit width of the MAT Keys for ingress stage = 160

Total bit width of the MAT Keys for egress stage = 16

Total ununsed MAT Key bitwidth = 1104

Mat key statistics Values for graph drawing 16 160 16 1104

totalNumberOfFieldsBeingModified = 5 headerBitWidthOfFieldsBeingModified = 80

totalNumberOfFieldsUsedAsParameter = 1 totalBitWidthOfFieldsUsedAsParameter = 16

Maximum bitwdth of the actions used for Ingress Stage = 48

Maximum bitwdth of the actions used for Egress Stage = 32

Total unused action key bitwidth = 1200

Action Key field statistics for graph drawing 16 48 32 1200

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 17 is following

Total number of fields used as key for MAT = 1

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1272

Mat key statistics Values for graph drawing 17 0 8 1272

totalNumberOfFieldsBeingModified = 5 headerBitWidthOfFieldsBeingModified = 224

totalNumberOfFieldsUsedAsParameter = 4 totalBitWidthOfFieldsUsedAsParameter = 160

Maximum bitwidth of the actions used for Ingress Stage = 384

Maximum bitwidth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 896

Action Key field statistics for graph drawing 17 384 0 896

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Following Register Array from ingress portion of pipeline is mapped to this stage

Register Array

name:IngressPipeImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.flowlet\_  
lasttime\_map

■

Total Resource usage in stage -- 18 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 8

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 18 8 8 1264

totalNumberOfFieldsBeingModified = 5 headerBitWidthOfFieldsBeingModified = 64

totalNumberOfFieldsUsedAsParameter = 5 totalBitWidthOfFieldsUsedAsParameter = 136

Maximum bitwidth of the actions used for Ingress Stage = 104

Maximum bitwidth of the actions used for Egress Stage = 64

Total unused action key bitwidth = 1112

Action Key field statistics for graph drawing 18 104 64 1112

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 19 is following

Total number of fields used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 16

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1256

Mat key statistics Values for graph drawing 19 16 8 1256

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 3 totalBitWidthOfFieldsUsedAsParameter = 224

Maximum bitwidth of the actions used for Ingress Stage = 192

Maximum bitwidth of the actions used for Egress Stage = 32

Total unused action key bitwidth = 1056

Action Key field statistics for graph drawing 19 192 32 1056

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 20 is following

Total number of fields used as key for MAT = 6

Total bit width of the MAT Keys for ingress stage = 32

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1232

Mat key statistics Values for graph drawing 20 32 16 1232

totalNumberOfFieldsBeingModified = 17 headerBitWidthOfFieldsBeingModified = 200

totalNumberOfFieldsUsedAsParameter = 5 totalBitWidthOfFieldsUsedAsParameter = 128

Maximum bitwidth of the actions used for Ingress Stage = 72

Maximum bitwidth of the actions used for Egress Stage = 72

Total unused action key bitwidth = 1136

Action Key field statistics for graph drawing 20 72 72 1136

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Following Register Array from ingress portion of pipeline is mapped to this stage

Register Array name:egress\_queue\_rate\_value\_map

Total Resource usage in stage -- 21 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 21 0 16 1264

totalNumberOfFieldsBeingModified = 20 headerBitWidthOfFieldsBeingModified = 320

totalNumberOfFieldsUsedAsParameter = 9 totalBitWidthOfFieldsUsedAsParameter = 216

Maximum bitwidth of the actions used for Ingress Stage = 32

Maximum bitwidth of the actions used for Egress Stage = 400

Total unused action key bitwidth = 848

Action Key field statistics for graph drawing 21 32 400 848

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 22 is following

Total number of fields used as key for MAT = 4

Total bit width of the MAT Keys for ingress stage = 32

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1248

Mat key statistics Values for graph drawing 22 32 0 1248

totalNumberOfFieldsBeingModified = 12 headerBitWidthOfFieldsBeingModified = 240

totalNumberOfFieldsUsedAsParameter = 12 totalBitWidthOfFieldsUsedAsParameter = 264

Maximum bitwidth of the actions used for Ingress Stage = 64

Maximum bitwidth of the actions used for Egress Stage = 376

Total unused action key bitwidth = 840

Action Key field statistics for graph drawing 22 64 376 840

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 23 is following

Total number of fields used as key for MAT = 6

Total bit width of the MAT Keys for ingress stage = 32

Total bit width of the MAT Keys for egress stage = 16

Total unused MAT Key bitwidth = 1232

Mat key statistics Values for graph drawing 23 32 16 1232

totalNumberOfFieldsBeingModified = 17 headerBitWidthOfFieldsBeingModified = 496

totalNumberOfFieldsUsedAsParameter = 13 totalBitWidthOfFieldsUsedAsParameter = 448

Maximum bitwidth of the actions used for Ingress Stage = 64



Maximum bitwidth of the actions used for Egress Stage = 816

Total unused action key bitwidth = 400

Action Key field statistics for graph drawing 23 64 816 400

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 24 is following

Total number of fields used as key for MAT = 3

Total bit width of the MAT Keys for ingress stage = 16

Total bit width of the MAT Keys for egress stage = 8

Total unused MAT Key bitwidth = 1256

Mat key statistics Values for graph drawing 24 16 8 1256

totalNumberOfFieldsBeingModified = 17 headerBitWidthOfFieldsBeingModified = 496

totalNumberOfFieldsUsedAsParameter = 10 totalBitWidthOfFieldsUsedAsParameter = 400

Maximum bitwidth of the actions used for Ingress Stage = 40

Maximum bitwidth of the actions used for Egress Stage = 816

Total unused action key bitwidth = 424

Action Key field statistics for graph drawing 24 40 816 424

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 25 is following

Total number of fields used as key for MAT = 2

Total bit width of the MAT Keys for ingress stage = 16

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1264

Mat key statistics Values for graph drawing 25 16 0 1264

totalNumberOfFieldsBeingModified = 4 headerBitWidthOfFieldsBeingModified = 64

totalNumberOfFieldsUsedAsParameter = 8 totalBitWidthOfFieldsUsedAsParameter = 192

Maximum bitwidth of the actions used for Ingress Stage = 64

Maximum bitwidth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1216

Action Key field statistics for graph drawing 25 64 0 1216

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Following Register Array from ingress portion of pipeline is mapped to this stage

Register Array

name:IngressPipelImpl.cp\_assisted\_multicriteria\_upstream\_policy\_routing\_control\_block.flowlet\_  
last\_used\_port

Total Resource usage in stage -- 26 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 26 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 26 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 27 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280


Mat key statistics Values for graph drawing 27 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 0



Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 27 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 28 is follwoing

Total number of fileds used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total ununsed MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 28 0 0 1280

totalnumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwdth of the actions used for Ingress Stage = 0

Maximum bitwdth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 28 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

■

Total Resource usage in stage -- 29 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 29 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 29 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 30 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 30 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwdth of the actions used for Ingress Stage = 0

Maximum bitwdth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 30 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []

Total Resource usage in stage -- 31 is follwoing

Total number of fileds used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total ununsed MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 31 0 0 1280

totalnumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwdth of the actions used for Ingress Stage = 0


Maximum bitwdth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 31 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []



Total Resource usage in stage -- 32 is following

Total number of fields used as key for MAT = 0

Total bit width of the MAT Keys for ingress stage = 0

Total bit width of the MAT Keys for egress stage = 0

Total unused MAT Key bitwidth = 1280

Mat key statistics Values for graph drawing 32 0 0 1280

totalNumberOfFieldsBeingModified = 0 headerBitWidthOfFieldsBeingModified = 0

totalNumberOfFieldsUsedAsParameter = 0 totalBitWidthOfFieldsUsedAsParameter = 0

Maximum bitwidth of the actions used for Ingress Stage = 0

Maximum bitwidth of the actions used for Egress Stage = 0

Total unused action key bitwidth = 1280

Action Key field statistics for graph drawing 32 0 0 1280

Common elements modified by ingress and egress in this stage is []

Common elements used as parameter by ingress and egress in this stage is []



## Conclusion

From The previous discussion, we can see the stage requirement of leaf switches of P4TE along the critical path is 27 stages. Moreover, our code is not optimized. We are working on developing a generalized Traffic engineering framework, hence for flexibility a lot of codes are unoptimized. But still P4TE needs 27 stages where 32 stages at both ingress and egress are available. The non leaf switch is same to leaf with only one exception. It does not require to store per flow state used in rate control. Hence, the non-leaf switch's P4 program is even simpler. Hence we can say P4TE is realizable in currently available P4TE hardware.

