

P4TE

Analyzing P4 Code to RMT Hardware Mapping

Debobroto Das Robin
Kent State University

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



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 3 sections:

1. The Ingress Control Bock
2. The Egress Control Bock
3. An example of how to convert a dangling If-Else statement to a TCAM based implementation

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: EgressPipelImpl.egress_rate_monitor_control_block.egress_rate_monitor_table

Match Keys are:

`*) standard_metadata.egress_port`

Actions are:

1 Action Name:

EgressPipelImpl.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 bitwdth of the actions used for Ingress Stage = 16

Maximum bitwdth 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 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 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 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 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 []

Result

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. And we are nt including it here. Hence we can say P4TE is realizable in currently available P4TE hardware.

