Programming Language for Switches

ECE/CS598HPN

Radhika Mittal

Conventional SDN

- Very flexible control plane in software.
- Interacts with dataplane through OpenFlow.
- Dataplane flexibility limited by:
    - what OpenFlow supports.
    - what the underlying hardware can support.

OpenFlow Support

Version	Date	# Headers
OF 1.0	Dec 2009	12
OF 1.1	Feb 2011	15
OF 1.2	Dec 2011	36
OF 1.3	Jun 2012	40
OF 1.4	Oct 2013	41

Programmable Switches

PISA: Protocol Independent Switch Architecture

- RMT:
  - Programmable parsers.
  - Reconfigurable match-action tables.

- Intel FlexPipe

- Cavium Xpliant

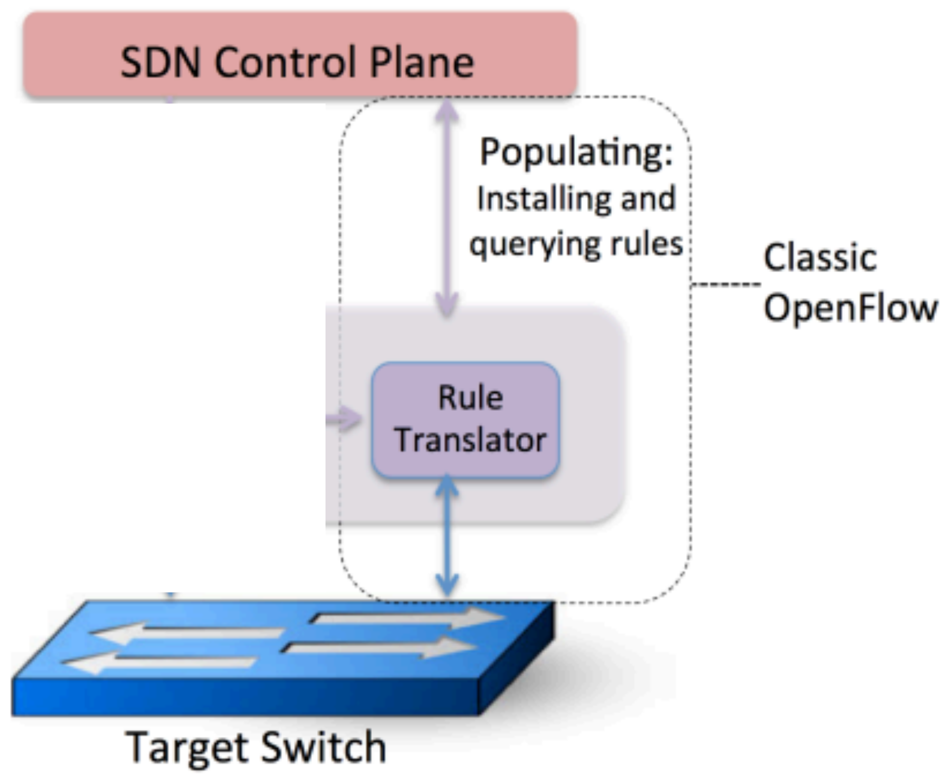
What was missing?

An interface to program such switches.

P4 Goals

- Protocol independence
  - Switches are not tied to specific packet formats.
- Reconfigurability
  - Controller can redefine packet parsing and processing in the field.
- Target Independence
  - User program need not be tied to a specific hardware.
  - Compiler's job to do the mapping.

P4 vs OpenFlow



Components of a P4 program

- Header definitions
- Parser definition
- Tables: what fields to match on, and which action to execute
- Action definition.

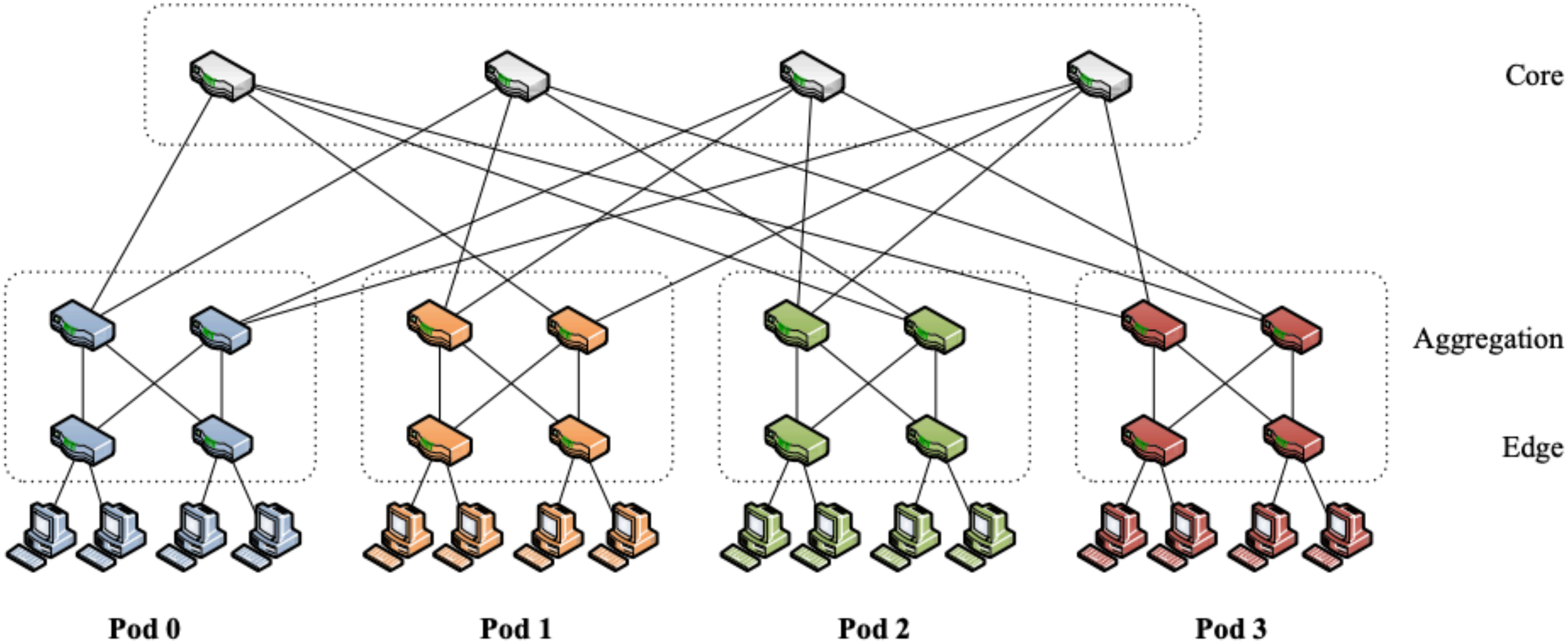
P4 Compiler

- If the target is a fixed-function switch?
  - Check if specified parser and match-action tables are supported.
  - If not, return error.

- If target is a software switch?
  - Full flexibility to execute specified program.
  - May use specific software data structures for optimizations.

- If target is an RMT switch?
  - Figure out table layout
    - mapping logical stages to physical ones.
    - When to use RAM vs TCAM
  - If tables don't fit, an action not supported, etc: return an error.

Example



From PortLand, SIGCOMM'09

Example

```
header ethernet {
    fields {
        dst_addr : 48; // width in bits
        src_addr : 48;
        ethertype : 16;
    }
}
```

```
header vlan {
    fields {
        pcp : 3;
        cfi : 1;
        vid : 12;
        ethertype : 16;
    }
}
```

```
header mTag {
    fields {
        up1 : 8;
        up2 : 8;
        down1 : 8;
        down2 : 8;
        ethertype : 16;
    }
}
```

Example

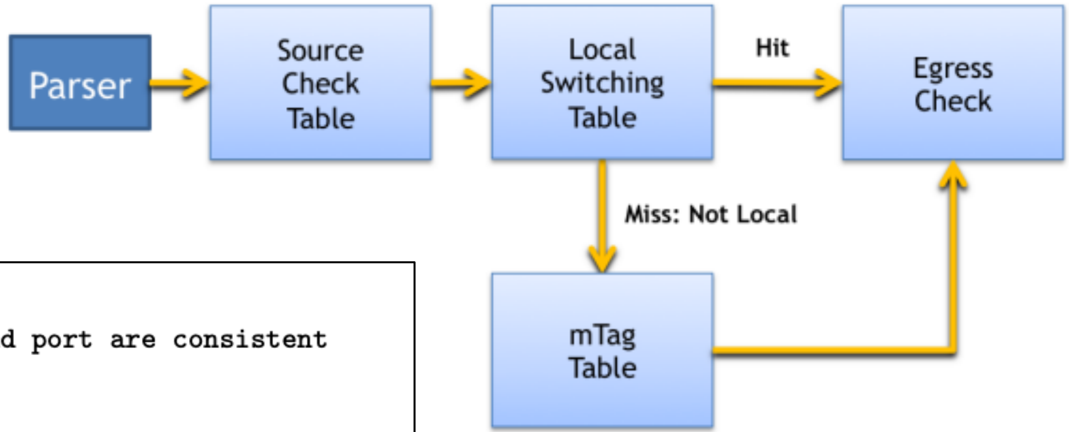
```
parser start {
    ethernet;
}

parser ethernet {
    switch(ethertype) {
        case 0x8100: vlan;
        case 0x9100: vlan;
        case  0x800: ipv4;
        // Other cases
    }
}
```

```
parser vlan {
    switch(ethertype) {
        case 0xaaaa: mTag;
        case  0x800: ipv4;
        // Other cases
    }
}

parser mTag {
    switch(ethertype) {
        case 0x800: ipv4;
        // Other cases
    }
}
```

Example




```
control main() {
    // Verify mTag state and port are consistent
    table(source_check);

    // If no error from source_check, continue
    if (!defined(metadata.ingress_error)) {
        // Attempt to switch to end hosts
        table(local_switching);

        if (!defined(metadata.egress_spec)) {
            // Not a known local host; try mtagging
            table(mTag_table);
        }

        // Check for unknown egress state or
        // bad retagging with mTag.
        table(egress_check);
    }
}
```

Example

```
table mTag_table {
    reads {
        ethernet.dst_addr : exact;
        vlan.vid : exact;
    }
    actions {
        // At runtime, entries are programmed with params
        // for the mTag action.  See below.
        add_mTag;
    }
    max_size : 20000;
}
```

Example

```
action add_mTag(up1, up2, down1, down2, egr_spec) {
    add_header(mTag);
    // Copy VLAN ethertype to mTag
    copy_field(mTag.ethertype, vlan.ethertype);
    // Set VLAN's ethertype to signal mTag
    set_field(vlan.ethertype, 0xaaaa);
    set_field(mTag.up1, up1);
    set_field(mTag.up2, up2);
    set_field(mTag.down1, down1);
    set_field(mTag.down2, down2);

    // Set the destination egress port as well
    set_field(metadata.egress_spec, egr_spec);
}
```

Example

- This was the edge switch's mTag match-action table.
- What will the core do?
  - Table will have ternary match on mTag
  - Action will be mTag_forward
    - Forward on specified port.
    - The rule about which mTag matches to which port is part of the configuration file.

What are some new opportunities enabled by P4?

Limitations of P4 or PISA?

Event-driven packet processing

- Restricted to packet ingress and egress events.
- May want to react to other types of events:
  - Periodic (generate probes, reset counters), link failures.
- Updated switch architecture design that allows reacting on such events.
- Ibanez et. al., HotNets'19.

External memory for switches

- Switches require high memory bandwidth.
  - Use fast, but expensive on-chip SRAM and TCAM.
  - Limited in size.

- Memory size could be a limiting factor for many applications.

- Let's access endhost memory remotely….

- Kim et. al., HotNets'18, SIGCOMM'20

On P4 compilers and programs

- Some "target-awareness" is still needed when writing P4 program (e.g. array sizes etc). How to enable true target-independence?
  - Use elastic data structures
  - P4All, Hogan et. al., HotNets'20, NSDI'22
- Compiler must somehow map a given P4 program to target hardware. How to optimize this?
  - Profile-guided optimizations, Wintermeyer et. al., HotNets'20
- Mapping a P4 program to multiple subprograms on different switches and running them as a distributed system.
  - FlightPlan, Sultana et. al., NSDI'21

Enabling runtime programmability

- In-situ Programmable Switching
  - Feng et. al, HotNets'21, NSDI'22

- Runtime Programmable Switches
  - Xing et. al., HotNets'21, NSDI'22

- ActiveRMT
  - Das et. al., HotNets'20, SIGCOMM'23

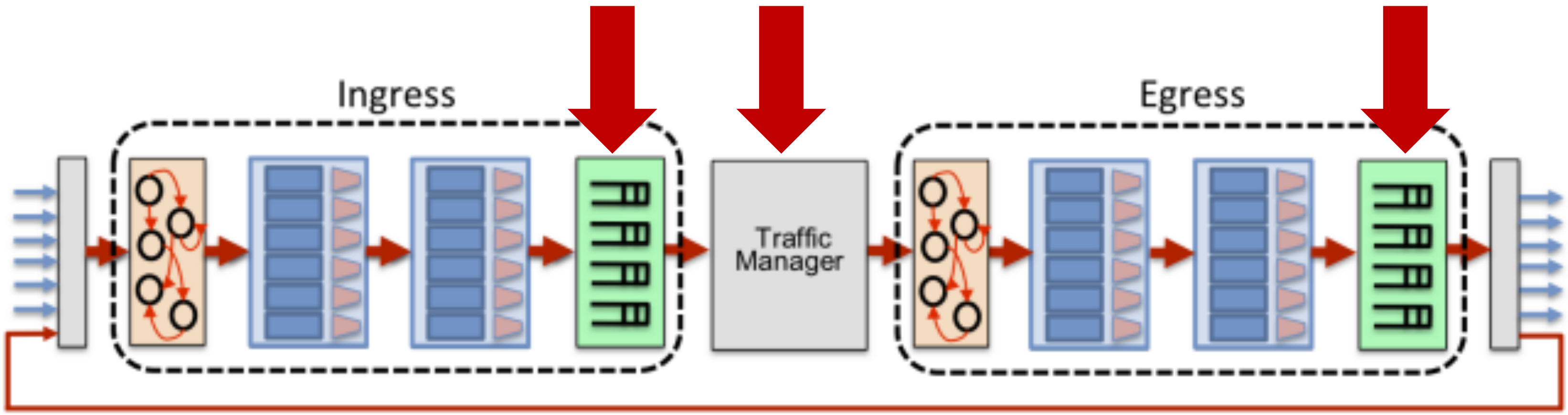
Extending/testing capabilities

- Statistics in P4
  - Gao et. al., HotNets'21
- Floating point operations in P4
  - Yuan et. al., NSDI'22

On state management

- Vision for enabling more stateful packet processing
  - Gebara et. al., HotNets'20
- Distributed state management for a program running on multiple switches
  - Zeno et. al., HotNets'20, NSDI'22
- Fault-tolerance for switch state
  - RedPlane, Kim et. al., SIGCOMM'22

Flexible queuing and scheduling



Later in the course…..

Logistics

- Project proposals:
  - Have you signed up for a slot for tomorrow? Any questions?

- Warm-up assignment 2 has been released! Due next week.