ETH zürich

Networked Systems
ETH Zürich — seit 2015

D-ITET

Prof. Dr. Laurent Vanbever

February 2022

R. Jacob, E. Costa Molero,
A. Dietmüller, A. Gran Alcoz, E. Kirci, R. Meier

Exam: Advanced Topics in Communication Networks

3 February 2022, 09:00–11:00, Room HG E 7

- ▷ You **must wear a face mask at all times** during the exam, except for short eating and drinking breaks. Only medical (IIR) and FFP2 masks are allowed. Contact the assistants in case you need a spare mask.

- ▷ Write your **name** and your **ETH student number** below on this front page and **sign it**.
- ▷ Put your **legitimation card** on the most accessible corner of your desk. Make sure that the side containing your name and **student number is visible**.
- ▷ Verify that you have received **all task sheets** (Pages **1 - 32**).
- ▷ **Do not separate** the task sheets. We will collect the exams **only after you have left** the room.

- ▷ Write your answers directly on the task sheets.
- ▷ All answers fit within the allocated space—often in much less.
- ▷ If you need more space, use the **extra sheets** at the end of the exam. Indicate the **task** in the corresponding field.
- ▷ **Read each task completely before you start solving it.**
- ▷ It is not required to score all points to get the best mark.

- ▷ Answer in **English**.
- ▷ **Write clearly** in blue or black ink (not red) using a **pen**, not a pencil.
- ▷ **Cancel** invalid parts of your solutions **clearly** (e.g., by crossing them out).
- ▷ At the end of the exam, **place the exam face up** on the most accessible corner of your desk. Then collect all your belongings and **exit the room** according to the given instructions.

- ▷ No written material nor calculator are allowed.

Family name: Student legi nr.:

First name: Signature:

Do not write in the table below (used by correctors only):

Task	Points	Sig.
Programmable data planes	/25	
Managing network traffic	/20	
Optimizing network performance	/25	
Design question	/50	
Total	/120	

leave blank

leave blank

Task 1: Programmable data planes — 25 Points

a) General Questions — (8 Points)

For each of the following statements, indicate whether they are *true* or *false*. There is always one correct answer. Each block of questions is awarded up to 4 points: 4 points for 4 correct answers, 2 points for 3 correct answers, and 0 points otherwise.

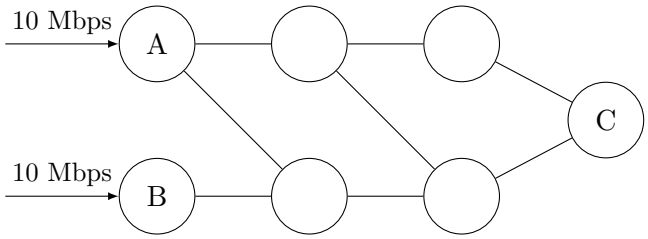
(i) P4 language, architecture, and programs (4 Points)

true	false	
☐	☐	No part of a P4 program can contain loops.
☐	☐	P4 supports variable length headers and a variable number of headers.
☐	☐	The `extern` construct provides an interface to functionalities that are not implemented in the P4 program itself.
☐	☐	Any correct P4 program can be compiled for any P4 device.

(ii) Load balancing (4 Points)

Programmable data planes enable a variety of load balancing decisions. Consider the network below, where all links have a capacity of 10 Mbps, equal delay, and equal IGP weights (e.g. equal OSPF cost).

10 Mbps of TCP traffic are entering at A and B respectively, both destined for C. Assume the traffic to consist of many flows.



true	false	
☐	☐	In the network above, ECMP ensures optimal throughput.
☐	☐	In the network above, randomly assigning flowlets to equal-cost paths leads to better throughput than ECMP on average.
☐	☐	In general, traffic created by any transport protocol leads to flowlets whose size reacts to congestion.
☐	☐	Per-packet load balancing prevents reordering packets in the network.

leave blank

b) Probabilistic data structures **(17 Points)**

You are given a Bloom filter that uses 10 cells and 3 hash functions. The hash functions for this task are defined as follows:

$$\begin{aligned} hash_1(x) &= x && \mod 10 \\ hash_2(x) &= 3 \cdot (x+1) && \mod 10 \\ hash_3(x) &= 5 \cdot (x+1) && \mod 10 \end{aligned}$$

where x is the numerical input value. In this task, we will input words, and use the sum of the letters' ASCII values as x. For example, the word *AdvNet* has value 578. The numerical values you need are provided in Table 1.

	x	$3\ (x+1)$	$5\ (x+1)$
AdvNet	578	1737	2895
exam	427	1284	2140
easy	434	1305	2175
hard	415	1248	2080

Table 1: Numerical word values.

(i) Start with an empty Bloom filter and update it by inserting the words *AdvNet*, *exam*, and *easy* one after another. Use the table below to indicate the state of the Bloom filter after each word. (3 Points)

	(empty)		**AdvNet**		**exam**		**easy**
0	0						
1	0						
2	0						
3	0						
4	0	→		→		→	
5	0						
6	0						
7	0						
8	0						
9	0						

(ii) After inserting all words, query the Bloom filter for the word *hard*, and explain the result: what happened, and why? (3 Points)

Query result for *hard*: true ☐ false ☐

Explain the result:

(iii) You decide to improve the overall performance of the Bloom filter by using a fourth hash function in addition to the current hash functions:

$$hash_4(x) = 7 \cdot (x + 1) \mod 10$$

Again, insert the words *AdvNet*, *exam*, and *easy* into an empty Bloom filter. Use the table below to indicate the state of the Bloom filter after each word. Afterwards, query the Bloom filter for the word *hard*. The numerical values you need are provided in table 2. (2 Points)

	x	$3\ (x+1)$	$5\ (x+1)$	$7\ (x+1)$
AdvNet	578	1737	2895	4053
exam	427	1284	2140	2996
easy	434	1305	2175	3045
hard	415	1248	2080	2912

Table 2: Numerical word values.

	(empty)		**AdvNet**		**exam**		**easy**
0	0						
1	0						
2	0						
3	0						
4	0	→		→		→	
5	0						
6	0						
7	0						
8	0						
9	0						

Query result for *hard*: true ☐ false ☐

(iv) Do additional hash functions *always* improve a Bloom filter? Explain your reasoning. (2 Points)

(v) Consider the following code taken from a P4 program that implements the Bloom filter above. Your final task is to modify this program to turn the Bloom filter into a CountMin sketch. Assume that you will need to count 2^{16} elements at most.

For each of the three sections of the program (metadata and register declaration, `insert`, and `query`), explain: *(i)* which parts of the code can remain the same, and *(ii)* which parts require changes, and how they need to be changed. Motivate your answers.

Only explain. Do not write P4 code. (7 Points)

```
#define BLOOM_FILTER_ENTRIES 10

struct metadata {
  bit<32> hash_one;
  bit<32> hash_two;
  bit<32> hash_three;
  bit<32> hash_four;
  bit<1> out_one;
  bit<1> out_two;
  bit<1> out_three;
  bit<1> out_four;
  bit<1> query_result;
}

register<bit<1>>(BLOOM_FILTER_ENTRIES) bloom_filter;

action insert() {
  // Compute hashes, abbreviated for simplicity.
  hash(meta.hash_one, ...);
  hash(meta.hash_two, ...);
  hash(meta.hash_three, ...);
  hash(meta.hash_four, ...);

  // Update.
  bloom_filter.write(meta.hash_one, 1);
  bloom_filter.write(meta.hash_two, 1);
  bloom_filter.write(meta.hash_three, 1);
  bloom_filter.write(meta.hash_four, 1);
}

action query() {
  // Compute hashes, abbreviated for simplicity.
  hash(meta.hash_one, ...);
  hash(meta.hash_two, ...);
  hash(meta.hash_three, ...);
  hash(meta.hash_four, ...);

  // Read.
  bloom_filter.read(meta.out_one, meta.hash_one);
  bloom_filter.read(meta.out_two, meta.hash_two);
  bloom_filter.read(meta.out_three, meta.hash_three);
  bloom_filter.read(meta.out_four, meta.hash_four);

  bit<1> temp_one = meta.out_one & meta.out_two;
  bit<1> temp_two = meta.out_three & meta.out_four;
  meta.query_result = temp_one & temp_two;
}
```

Metadata and register declaration:

`insert` action:

`query` action:

leave blank

leave blank

Task 2: Managing network traffic **20 Points**

a) Label switching theory **(4 Points)**

For each of the following statements, indicate whether they are *true* or *false*. There is always one correct answer. Each block of questions is awarded up to 4 points: 4 points for 4 correct answers, 2 points for 3 correct answers, and 0 points otherwise.

true	false	
☐	☐	The Resource Reservation Protocol (RSVP) establishes Label Switched Paths (LSPs) bidirectionally.
☐	☐	A Label Switched Router (LSR) may assign multiple labels for a particular address prefix.
☐	☐	MPLS allows for both traffic aggregation and disaggregation, whereas classical, destination-based IP forwarding allows for aggregation only.
☐	☐	Only the Egress Label Edge Routers can remove MPLS labels.

b) Label switching application **(16 Points)**

leave blank

This subtask is based on the topology shown in Figure 1.

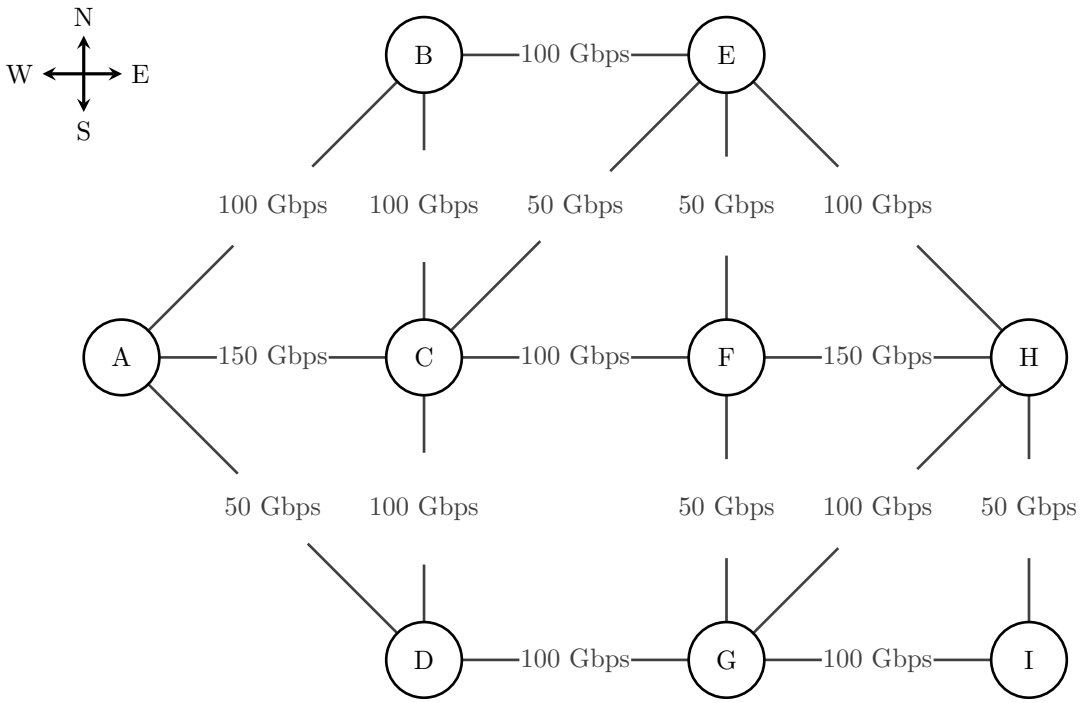


Figure 1: The topology of the network. Link annotations indicate the link's bidirectional capacity, and all links have the same delay.

(i) Consider the network topology shown in Figure 1. Link annotations indicate the link's bidirectional capacity, and all links have the same delay. Find the shortest constrained path(s) for an aggregated traffic demand of 75 Gbps from Router A to Router I. Describe the steps of your reasoning. (3 Points)

Path: _____

Algorithm steps: _____

Router A

Inlabel	Nexthop	Operation
L1	SE	Swap(L7)
L5	NE	Push(L8)
L2	E	Swap(L12)
L3	E	Push(L6)
L7	NE	Push(L8)
L4	E	Push(L6)
L6	NE	Push(L8)

Router B

Inlabel	Nexthop	Operation
L8	E	Pop
L1	S	Push(L3)

Router C

Inlabel	Nexthop	Operation
L1	E	Pop
L12	S	Swap(L1)
L6	E	Pop
L3	NE	Pop

Router D

Inlabel	Nexthop	Operation
L7	E	Push(L15)
L1	E	Push(L15)

Router E

Inlabel	Nexthop	Operation
L5	SE	Swap(L3)
L2	SW	Push(L1)
L7	S	Swap(L6)
L1	SE	Swap(L5)

Router F

Inlabel	Nexthop	Operation
L4	E	Swap(L7)
L2	E	Push(L6)
L6	S	Push(L2)

Router G

Inlabel	Nexthop	Operation
L15	E	Pop
L2	NE	Pop

Router H

Inlabel	Nexthop	Operation
L7	SW	Swap(L2)
L5	Local	Pop
L3	S	Swap(L4)
L1	S	Swap(L5)
L6	NW	Pop

Router I

Inlabel	Nexthop	Operation
L7	Local	Pop
L1	Local	Pop
L4	Local	Pop
L3	Local	Pop
L5	Local	Pop
L2	Local	Pop
L6	Local	Pop

Table 3: Label forwarding tables for each router.

(ii) Consider the label forwarding tables shown in Table 3. In this forwarding state, there exists at least two paths from Router A to Router I which are non-overlapping; i.e., the LSP tunnels do not share any edge. For a packet entering in Router A, find two possible labels (one for each path) that will eventually deliver the packet to Router I. Indicate the path taken for each case with the associated label headers at each hop. (6 Points)

Hint: For example, a packet initially labeled with L1 entering in Router B terminates at Router H with L5. The path taken in this case is as follows:
$L1 \rightarrow (B) \rightarrow L3, L1 \rightarrow (C) \rightarrow L1 \rightarrow (E) \rightarrow L5 \rightarrow (H)$

Path 1:

Path 2:

(iii) Explain one purpose of label stacking. Then, find two links with label stacking based on Table 3. How do the stacked labels on these links help with the indicated purpose? (4 Points)

Purpose:

Links:

How do they help:

(iv) Find one forwarding loop in the network. (3 Points)

Hint: One loop includes Router F.

leave blank

leave blank

Task 3: Optimizing network performance **25 Points**

a) QoS-enabled router **(10 Points)**

Due to the recent remote-working restrictions, you find yourself at home sharing your Internet connection with your parents and your brother. Even though you are subscribed to the maximum capacity that your ISP can provide, you identify congestion in the upstream of your home router (see Figure 2). Being a student of the Adv-Net lecture, you decide to put in practice what you have learned about Quality of Service (QoS) to handle the congestion.

Mother	Download (FTP)
Father	VoIP, Web browsing
Brother	Video game
You	Video streaming

Table 4: Used applications

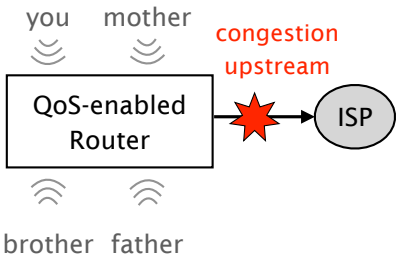


Figure 2: Router setup

(i) First, you verify that your home router is QoS-enabled. What does it mean for a router to be QoS-enabled? Which components do you expect a QoS-enabled router to have? Describe briefly the purpose of each component. (3 Points)

QoS-enabled meaning:

QoS components:

(ii) For the Download and VoIP applications, how important is it to be served with low delay and high bandwidth? Classify as "low/medium/high" and justify your answers. What QoS policy would you apply in order to optimize the QoS across these two applications? (3 Points)

Importance of delay for VoIP: _____

Importance of bandwidth for VoIP: _____

Justification: _____

_____

_____

Importance of delay for Download: _____

Importance of bandwidth for Download: _____

_____

_____

QoS policy: _____

_____

_____

(iii) The first step in implementing QoS at the router is to classify the ingress traffic of each application. What traffic features would you use to classify the traffic from Download and VoIP applications? Justify your answer. (2 Points)

_____

_____

_____

_____

_____

_____

(iv) You want to reduce the quality of your video-streaming service from its default 4K resolution, to Full-HD (FHD). You want to do that *from the router*. Explain what QoS technique would you use, and why. In Table 5, you will find the required bandwidth of each resolution for the codec used by your video-streaming service (H.265). (2 Points)

Resolution	Bandwidth
1280×720 (HD)	3Mbps
1920×1080 (FHD)	6Mbps
3840×2160 (UHD)	25Mbps
4096×2160 (4K)	32Mbps

Table 5: Bandwidth requirements

Technique:

Reason:

leave blank

b) Scheduling algorithms **(5 Points)**

(i) For the Fair-Queuing scheduling algorithm, explain its main objective, detail its design, and name one of its disadvantages. Is it a work-conserving or non-work conserving algorithm? Tick the correct box. (3 Points)

Objective: _____

Design: _____

_____

_____

_____

Disadvantage: _____

_____

☐ Work-conserving ☐ Non work-conserving

(ii) Figure 3 depicts the drop probability as a function of the average queue depth of a queue-management algorithm. What is the name of that algorithm? Explain what the algorithm is trying to achieve and how. What is the purpose of x_1 and x_2? (2 Points)

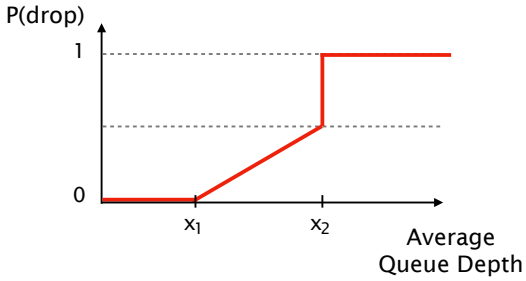


Figure 3: Drop probability of a queue-management algorithm

Algorithm name: _____

Objective and design: _____

_____

_____

_____

Purpose of x_1 and x_2: _____

_____

leave blank

c) Fast convergence **(10 Points)**

Consider the network topology in Figure 4. Each link is annotated with its IGP weight. In the following, we are interested in identifying possible Loop-Free Alternates (LFAs).

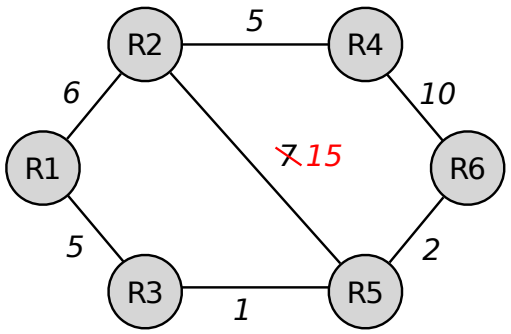


Figure 4: Network topology.

(i) What are LFAs, and why are they important in fast convergence? (3 Points)

(ii) Compute the shortest path from each of the different routers towards R6, as well as their cost. (2 Points)

Path between R1→R6: _____

Cost of R1→R6: _____

Path between R2→R6: _____

Cost of R2→R6: _____

Path between R3→R6: _____

Cost of R3→R6: _____

Path between R4→R6: _____

Cost of R4→R6: _____

Path between R5→R6: _____

Cost of R5→R6: _____

(iii) Which router(s), if any, can be considered a ~~per-link~~ per-prefix LFA of R1, to reach R6? Justify your answer. (3 Points)

(iv) Does R1 have any *remote* LFA (RLFA) towards R6? Justify your answer. (2 Points)

leave blank

Task 4: Design question **50 Points**

In this task, you are hired to deploy a new link-layer protocol in the network shown in Figure 5. The protocol is an alternative to `Ethernet` called `Advnet`. Its packets have a similar structure as `Ethernet`; the `Advnet` header consists of the following fields:

- `dstAddr` (11 bits): the destination address
- `srcAddr` (11 bits): the source address
- `advType` (16 bits): indicates the protocol used in the payload (e.g., IPv4)
- `flags` (10 bits): can be used for custom applications. Packets coming from hosts always have all flags set to `0`.

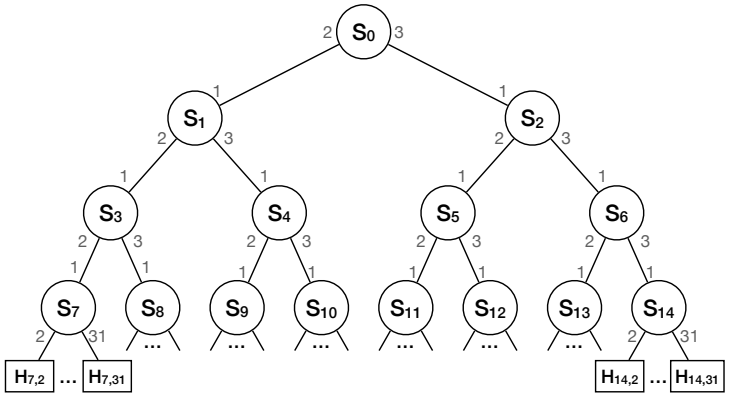


Figure 5: Network topology

`Advnet` is designed specifically to work in the topology shown in Figure 5. Note the following properties of the topology and `Advnet`:

- The topology has a tree structure and hosts are only connected to the leaf switches
- The network is isolated (there is no gateway to other networks or the Internet)
- Each switch has only 3 active ports, except for S_0 (2 ports) and the leaf switches (31 ports)
- Each leaf switch is connected to 30 hosts. A host $H_{i,j}$ is connected to switch S_i on port j

Further note that `Advnet` addresses are constructed in a particular way: The format is `A:B:C:D` where `A`, `B`, `C` and `D` describe the path to the addressed host starting from S_0:

- `A` (2 bits) identifies the port on which the switch at level 0 (S_0) needs to send the packet
- `B` (2 bits) identifies the port on which the switch at level 1 (S_1 or S_2) needs to send the packet
- `C` (2 bits) identifies the port on which the switch at level 2 (S_3 – S_6) needs to send the packet
- `D` (5 bits) identifies the port on which the leaf switch at level 3 (S_7 – S_{14}) needs to send the packet

For example, host $H_{7,2}$ has the `Advnet` address `2:2:2:2`.

During this task, you will develop a P4 program that runs on all switches in this network (the same code runs on all switches, only table entries are different for each switch). To start, Figure 6 contains a code skeleton that you will extend and modify during this task.

```
#include <core.p4>
#include <v1model.p4>

const bit<16> TYPE_IPV4 = 0x0800;

header advnet_t {
    bit<11>  dstAddr;
    bit<11>  srcAddr;
    bit<16>  advType;
    bit<10>  flags;
}

header ipv4_t {
    bit<4>  version;     bit<4>  ihl;            bit<8>    diffserv;
    bit<16> totalLen;    bit<16> identification; bit<3>    flags;
    bit<13> fragOffset;  bit<8>  ttl;            bit<8>    protocol;
    bit<16> hdrChecksum; bit<32> srcAddr;        bit<32>   dstAddr;
}

struct metadata { }

struct headers {
    advnet_t   advnet;
    ipv4_t     ipv4;
}

parser MyParser(packet_in packet, out headers hdr, inout metadata meta,
                inout standard_metadata_t standard_metadata) {

    state start {
        transition parse_advnet;
    }

    // ToDo: parse Advnet and IPv4
}

control MyVerifyChecksum(...) { apply {  } }

control MyIngress(inout headers hdr, inout metadata meta,
                  inout standard_metadata_t standard_metadata) {

    action send_to_port(bit<9> port) {
        standard_metadata.egress_spec = port;
    }

    action send_to_port_and_set_flags(bit<9> port, bit<10> flags) {
        standard_metadata.egress_spec = port;
        hdr.advnet.flags = flags;
    }

    table l2_forward {
        key = {
            hdr.advnet.dstAddr: ternary;
            hdr.advnet.flags:   ternary;
        }
        actions = {
            send_to_port;
            send_to_port_and_set_flags;
            NoAction;
        }
        default_action = NoAction();
        size = 1024;
    }
```

```
64
65      register<bit<32>>(2048) packets_sent;
66      register<bit<32>>(2048) packets_received;
67
68      action update_statistics() {
69          bit<16> packets_sent_before
70          packets_sent.read(packets_sent_before, (bit<32>) hdr.advnet.srcAddr);
71          packets_sent.write(packets_send_before + 1, (bit<32>) hdr.advnet.
        srcAddr);
72
73          bit<16> packets_received_before;
74          packets_received.read(packets_received_before, (bit<32>) hdr.advnet.
        dstAddr);
75          packets_received.write((bit<32>) hdr.advnet.dstAddr,
        packets_sent_before + 1);
76      }
77
78      table monitoring {
79          key = {
80              hdr.srcAddr: exact;
81          }
82          actions = {
83              NoAction;
84          }
85          default_action = NoAction();
86      }
87
88      #include <legacy_1.p4>
89
90      apply {
91          l2_forward.apply();
92          monitoring.apply();
93          #include <legacy_2.p4>
94      }
95  }
96
97
98  control MyEgress(...) { apply {  } }
99
100 control MyComputeChecksum(...) { apply { } }
101
102 control MyDeparser(packet_out packet, in headers hdr) {
103     apply {
104         // todo
105     }
106 }
107
108 V1Switch(MyParser(), MyVerifyChecksum(), MyIngress(), MyEgress(),
        MyComputeChecksum(), MyDeparser()) main;
```

Figure 6: P4 code skeleton

leave blank

a) Getting familiar with the network **(5 Points)**

Before you start implementing `Advnet`, the customer wants to make sure that you understand how the protocol and the topology works. So she asks you some short questions.

(i) What is the `Advnet` address of host $H_{10,11}$ (1 Point)

(ii) How many links does a packet from `3:2:3:5` to `3:3:3:4` traverse if it follows the shortest path? (Include the links between hosts and switches in your count). (1 Point)

(iii) One advantage of `Advnet` compared to `Ethernet` is that its header is shorter (the `Ethernet` header is 112 bits long). How long does it take to send 1'500'000 bytes of IP *payload* over an `Advnet` and an `Ethernet` link with the following properties:

- 100 Gbps line rate (12.5×10^9 bytes per second)
- no packet loss or congestion
- maximum 1500 bytes IP *payload* per packet

Your final result can be a fractional number (there is no need to compute the division). (3 Points)

with `Ethernet`

with `Advnet`

b) Parsing packets **(5 Points)**

leave blank

Now it is time to extend the program such that it parses and deparses `Advnet` packets correctly.

(i) The parser states for `Advnet` and `IPv4` are missing in the skeleton. Write their code below. Note that the skeleton already contains the header definitions and, if a packet is not `IPv4`, it is enough to parse the `Advnet` header. (4 Points)

```
state parse_advnet {
}

state parse_ipv4 {
}
```

(ii) Write the code for the deparser. (1 Point)

```
apply {
}
```

leave blank

c) Basic forwarding **(19 Points)**

The code skeleton already defines and applies a table called `l2_forward`. In this task, you need to fill the table with entries.

Note that the table uses ternary matches. In table entries, you can write ternary matches using "*" for bits/digits that should be ignored (for example, you can write "`1:*:3:4`" or "`01:**:11:00100`" to match on `Advnet` addresses). Since multiple entries can match on the same packet, list the rules with decreasing priority (first entry has the highest priority).

(i) The customer tells you that she wants to forward packets along the shortest paths. Which entries do you need to add to the `l2_forward` table on **switch S_1**? Use the smallest number of entries possible.
Abbreviate the action names as follows: "SP" for `send_to_port`, "SPF" for `send_to_port_and_set_flags` and "NA" for `NoAction`. (5 Points)

Match		Action	
`dstAddr`	`flags`	name	parameters

(ii) Could you reduce the number of entries if you could also change the P4 code? Explain why and how, if it is possible; or why not, if it is not possible. (1 Point)

(iii) The customer changes her mind and wants that all traffic transit via switch S_0 such that it can be inspected there (as opposed to sending it along the shortest path). How can you achieve this through the `l2_forward` table? Only explain your high-level idea here. You will be asked to provide the table entries later. (3 Points)

(iv) Which entries do you need to add to the `l2_forward` table on **switch S_0** to implement your idea from above? Use the smallest number of entries possible.
Abbreviate the action names as follows: "SP" for `send_to_port`, "SPF" for `send_to_port_and_set_flags` and "NA" for `NoAction`. (4 Points)

Match		Action	
`dstAddr`	`flags`	name	parameters

(v) Which entries do you need to add to the `l2_forward` table on **switch S_1** to implement your idea from above? Use the smallest number of entries possible.
Abbreviate the action names as follows: "SP" for `send_to_port`, "SPF" for `send_to_port_and_set_flags` and "NA" for `NoAction`. (4 Points)

Match		Action	
`dstAddr`	`flags`	name	parameters

(vi) Currently, the table reserves space for 1024 entries. But as you have found out above, the number of entries required is much smaller. How many entries need to fit in the table at most in this network? Consider the case where all traffic is sent via S_0 and no other changes to the code are allowed. Explain your answer. (2 Points)

leave blank

d) Computing statistics **(10 Points)**

The customer wants to compute statistics about the traffic in her network. In particular, she wants a way to specify a list of hosts for which the switches count the packets they send.

The code skeleton already contains some code for a table in which the monitored hosts can be specified and some code to compute the statistics using registers. The relevant lines are shown again below (Figure 7).

```
65      register<bit<32>>(2048) packets_sent;
66      register<bit<32>>(2048) packets_received;
67
68      action update_statistics() {
69          bit<16> packets_sent_before
70          packets_sent.read(packets_sent_before, (bit<32>) hdr.advnet.srcAddr);
71          packets_sent.write(packets_send_before + 1, (bit<32>) hdr.advnet.
        srcAddr);
72
73          bit<16> packets_received_before;
74          packets_received.read(packets_received_before, (bit<32>) hdr.advnet.
        dstAddr);
75          packets_received.write((bit<32>) hdr.advnet.dstAddr,
        packets_sent_before + 1);
76      }
77
78      table monitoring {
79          key = {
80              hdr.srcAddr: exact;
81          }
82          actions = {
83              NoAction;
84          }
85          default_action = NoAction();
86      }
```

Figure 7: Section of the code skeleton with many errors

(i) Unfortunately, the programmer who wrote this code made some mistakes (logical and syntactical ones), which you must find. Use the table below to list the lines that contain errors and write down the fixed version of them (rewrite the entire line). (6 Points)

Hint: The first error is in line 69 and there are 6 lines which contain at least one error.

Line	Correct code
69	

(ii) The code uses registers to count the packets sent, but not all P4 devices support registers. How would you need to change the code for it to run on a device that supports counters but not registers. Explain the required changes when using indirect and direct counters. You do not need to write the code. (4 Points)

Changes when using indirect counters:

Changes when using direct counters:

leave blank

e) Legacy code **(11 Points)**

Your customer also wants to have some legacy code included. This code is in the files `legacy_1.p4` and `legacy_2.p4` (Figures 8 and 9). The code compiles without errors but the customer does not remember what it does exactly. So she asks you to explain it to her. Unfortunately, the programmer did not document the code properly.

```
register<bit<48>>(2048) X;
register<bit<32>>(2048) Y;

table A {
    key = {
        hdr.advnet.srcAddr: exact;
    }
    actions = {
        NoAction;
    }
    default_action = NoAction();
}
```

Figure 8: `legacy_1.p4`

```
if (A.apply().hit) {
    bit<48> M;
    bit<32> N;
    bit<48> T = standard_metadata.ingress_global_timestamp; // microseconds

    X.read(M, (bit<32>) hdr.advnet.srcAddr);

    if (T > M+10000000) {
        X.write((bit<32>) hdr.advnet.srcAddr, T);
        Y.write((bit<32>) hdr.advnet.srcAddr, 1);
    }
    else {
        Y.read(N, (bit<32>) hdr.advnet.srcAddr);
        Y.write((bit<32>) hdr.advnet.srcAddr, N + 1);

        if (N > 3) {
            mark_to_drop(standard_metadata);
        }
    }
}
```

Figure 9: `legacy_2.p4`

(i) The code defines two register arrays (`X` and `Y`). What kind of information is stored in these registers? (2 Points)

Register `X`: _____

_____

Register `Y`: _____

_____

(ii) The code defines a table (`A`) but the only available action in this table is `NoAction`, which does not do anything to a packet. What is the purpose of this table? (2 Points)

(iii) How many entries can be added to table `A`? Explain your answer. (1 Point)

(iv) What does the legacy code do? (6 Points)

Hint: investigate what happens if a host's `Advnet` *address is in table* `A`.

Extra Sheet 1

In case you need more space, use the following pages. Make sure to always indicate the task to which the answer belongs (e.g., *3 d) (ii)*).

Task: _____

Extra Sheet 2

Task: _____

Task: _____