

Exam: Advanced Topics in Communication Networks

13 February 2026, 11:30–14:00, HG F 1 / HG F 5

Sample Solution

- ▷ Write **legibly** your ETH student number (legi number) below on this front page.
- ▷ **Do not write your name** or use a stamp with your name on it.
- ▷ **TRIPLE-check that your legi number is correct!**
You will not be graded if you make a mistake when writing your number.
- ▷ Put your legitimization card (legi) 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 after you 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, and add a “**see Extra Sheet X**” note in the original task space.
- ▷ **Read** each task **completely** before you start solving it.
- ▷ 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, devices or calculator are allowed.**
- ▷ It is not required to score all points to get the best mark.

Student legi nr.:

--

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

Task	Points
Advanced routing	/25
Sustainable networking	/17
Programmable data planes	/15
Network verification	/20
Transport	/18
Networking for ML	/15
ML for Networking	/10
Total	/120

Task 1: Advanced Routing**25 Points****a) Hierarchical routing****(6 Points)**

Consider the network topology composed of six BGP routers depicted in Figure 1 and configured either as:

1. An iBGP full-mesh; or as
2. A route reflection topology, as shown in Figure 2, with R1 and R2 acting as route reflectors (RRs) and R3, R4, R5 and R6 acting as clients. The dashed lines indicate iBGP sessions, with the single-headed arrows indicating client sessions (they point towards the route reflector) and the double-headed arrow indicating normal iBGP sessions.

The network relies upon an IGP whose weights are depicted next to each link.

Router R4 receives an eBGP route **r1** for a prefix *p*. At the same time, router R6 receives an eBGP route **r2** for the same prefix *p*. Route **r1** has **local-pref 50** and an **AS-PATH length of 2**, while route **r2** has **local-pref 40** and an **AS-PATH length of 2**.

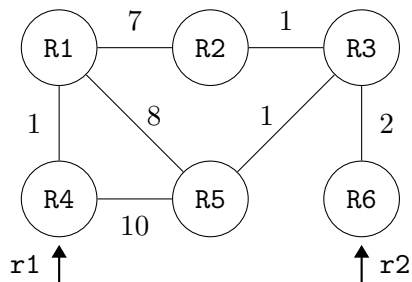


Figure 1: The base topology.

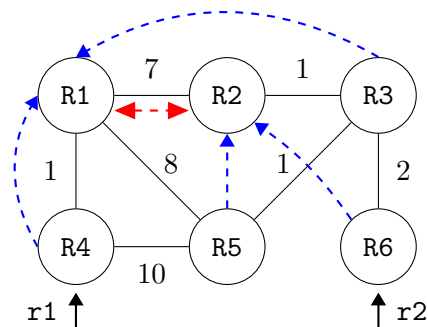


Figure 2: BGP sessions with route reflectors.

- (i) For every router, identify its selected egress router and the weight of the IGP path towards it (write 0 if the router and the egress router are the same). Do this for both the full mesh scenario (left table) and the route reflection one (right table). (3 Points)

Solution: No issues.

Router	Egress Router	IGP Weight
R1	R4	1
R2	R4	8
R3	R4	9
R4	R4	0
R5	R4	9
R6	R4	11

Router	Egress Router	IGP Weight
R1	R4	1
R2	R4	8
R3	R4	9
R4	R4	0
R5	R4	9
R6	R4	11

- (ii) Due to a new commercial agreement, the route r2 now also has **local-pref 50**. Fill in the tables for both scenarios as before. Use the left table for the full mesh scenario and the right table for the route reflection one. (2 Points)

Note: We show the topologies again below for convenience.

Solution: Forwarding loop between R2 and R3.

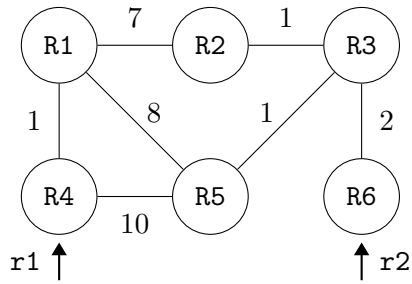


Figure 3: The base topology.

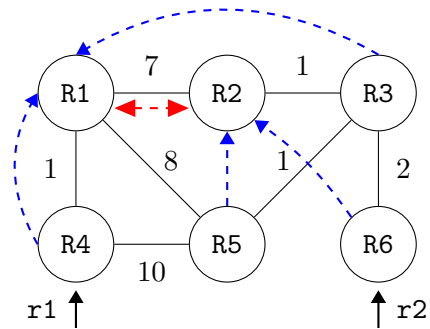


Figure 4: BGP sessions with route reflectors.

Router	Egress Router	IGP Weight
R1	R4	1
R2	R6	3
R3	R6	2
R4	R4	0
R5	R6	3
R6	R6	0

Router	Egress Router	IGP Weight
R1	R4	1
R2	R6	3
R3	R4	9
R4	R4	0
R5	R6	3
R6	R6	0

The route reflector scenario now results in a forwarding anomaly. What type of forwarding anomaly occurs, and where? (1 Point)

Solution: there is a forwarding loop between R2 and R3.

b) Loop-Free Alternates on Gao-Rexford policies

(8 Points)

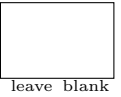
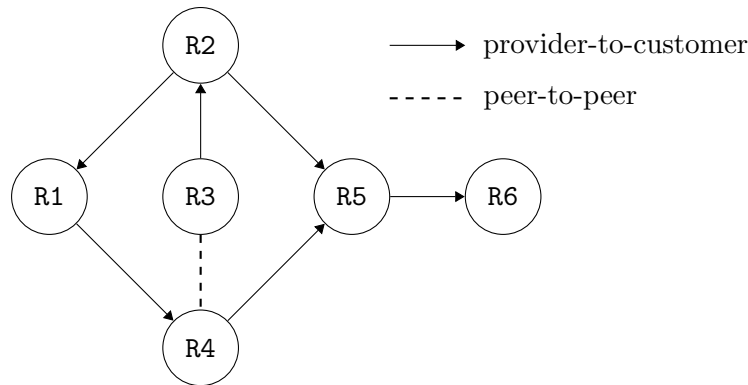


Figure 5: A simple topology following Gao-Rexford policies.

In this task paths are selected according to Gao-Rexford policies: besides preferring customers over peers over providers, ASes break ties by shortest AS path length.

- (i) Consider the topology in Figure 5, with Gao-Rexford policies as defined in the legend (e.g. R3 is a provider of R2, and R2 is a provider of R1 and R5). Find the paths used by R1 and by R4 to reach every other node and fill in the table below accordingly. We provide an example for R2. (2 Points)

Node	Path from R2	Path from R1	Path from R4
R1	R2 - R1	R1	R4 - R3 - R2 - R1
R2	R2	R1 - R2	R4 - R3 - R2
R3	R2 - R3	R1 - R2 - R3	R4 - R3
R4	R2 - R1 - R4	R1 - R4	R4
R5	R2 - R5	R1 - R4 - R5	R4 - R5
R6	R2 - R5 - R6	R1 - R4 - R5 - R6	R4 - R5 - R6

- (ii) Is R4 a per-link Loop-Free Alternate (LFA) for R1 considering the failure of the (R1, R2) link? Explain why or why not. (2 Points)

Solution: Yes, it is an LFA. There are two destinations that R1 reaches via the link R1-R2: R2 and R3. R4 reaches both of these without traversing said link, making it a LFA.

- (iii) Consider the network in Figure 6. You must assign a Gao-Rexford relationship to each of the 4 links such that:
- the best path from R3 to R2 *must not* traverse R1-R2, **and**
 - R3 *must not* be a valid per-link LFA for R1 given the failure of link R1-R2.

Draw the 4 links in Figure 7 with the corresponding relationships according to the legend. (4 Points)

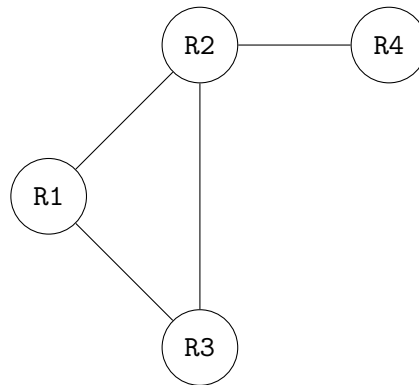


Figure 6: A simple topology.

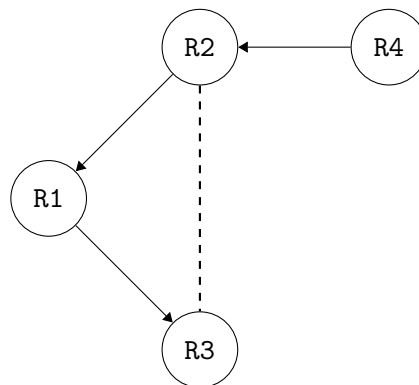
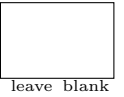


Figure 7: Draw the 4 links again, with the right relationships.

c) Aggregating Prefixes

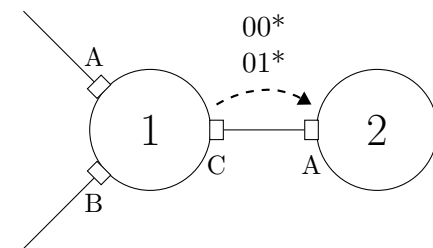
(11 Points)



In this task, we want to apply the ORTC algorithm for prefix filtering; however, we will do so **at the routing table level**. This will therefore affect not only the forwarding tables, but also the propagation of prefixes to other routers. (Similarly to DRAGON.)

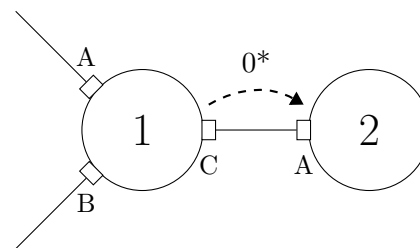
Figures 8 and 9 provide an example: In both cases, assume that node 1 has learned of prefixes 00^* and 01^* , and selected port B for both in its Routing Information Base (RIB).

- In Figure 8, it propagates both prefixes to node 2 (e.g. via BGP).
- In Figure 9, node 1 executes the ORTC algorithm to find the optimal aggregation of the two prefixes: a single prefix 0^* with selected port B. However, unlike the standard ORTC seen in the lecture, node 1 also stops propagating 00^* and 01^* to node 2, meaning node 2 only learns 0^* .



RIB of Node 1	
Prefix	Port
00^*	B
01^*	B

RIB of Node 2	
Prefix	Port
00^*	A
01^*	A



RIB of Node 1	
Prefix	Port
0^*	B

RIB of Node 2	
Prefix	Port
0^*	A

Figure 8: Without aggregation.

Figure 9: With aggregation. Node 2 only learns of prefix 0^* .

- (i) In Figure 10 we show a 3-node topology. Nodes 1 and 2 have received prefix advertisements from the rest of the network, and propagate all of them to node 3. Node 3 does not further propagate any prefixes it receives. We show the routing tables for each of the three nodes after convergence.

Consider now that both node 1 **and** 2 apply ORTC but that node 3 does **not**. Fill in the routing tables of the 3 nodes after convergence in Figure 11. (4 Points)

Note: You do not need all the rows.

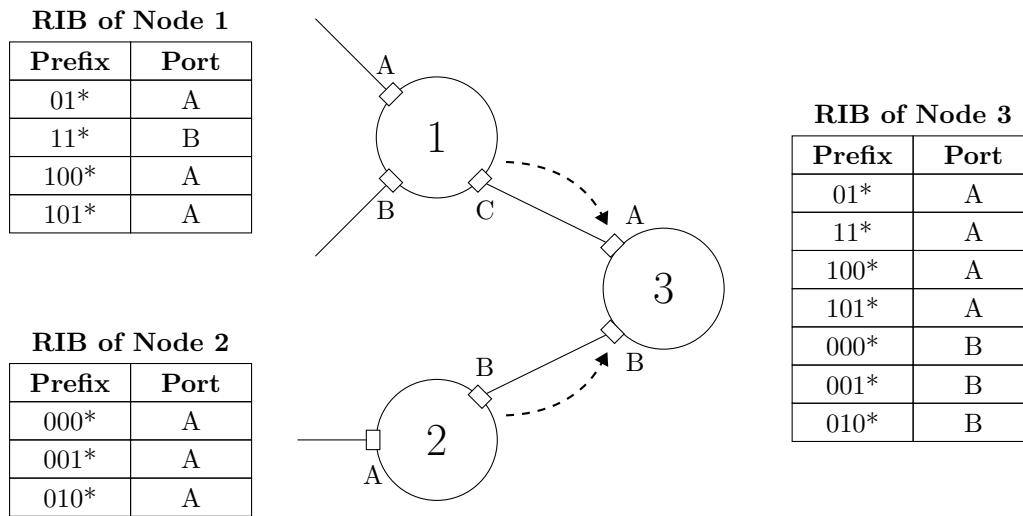


Figure 10: Without aggregation.

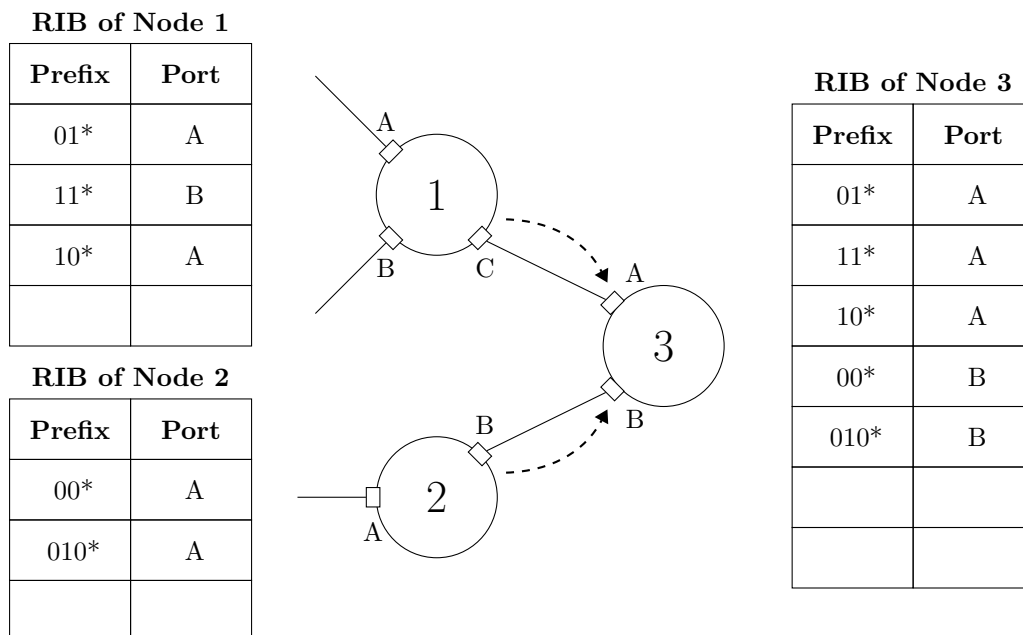


Figure 11: With aggregation at nodes 1 and 2.

- (ii) Weak forwarding consistency is defined as *all nodes forward IP packets to the same port after aggregation as they would if there had been no aggregation*. We want to prove that applying ORTC at the RIB level can break weak forwarding consistency.

Let us consider the same 3-node topology as in Figure 10. We ask you to add a single entry to the routing table of either node 1 or 2 such that applying ORTC leads to node 3 breaking weak forwarding consistency. Explain why this happens, and give an example IP which node 3 would forward on a different port after aggregation. (5 Points)

Hint: Think of what metric makes an IP match one forwarding entry over another, and how aggregation can change its value.

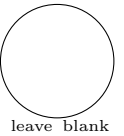
Add entry	Prefix	Port	to RIB of Node 2
	011*	A	

Example IP: anything in 00*.

Grading: no explanation → 0pts. Wrong entry → -2pt. No mention of longest prefix matching or explanation of the concept → -1pt. Wrong IP → -1pt.

- (iii) Strong forwarding consistency specifies that no IP becomes routable after aggregation if it had not been routable before. Can there be a case where RIB-level ORTC breaks strong forwarding consistency *without* breaking the weak one? That is, can an IP become routable after the aggregation while all previously routable IPs are forwarded to the same ports? If yes, give an example. If no, explain why not. (2 Points)

Solution: No. The ORTC algorithm does not let a non-routable destination become routable. This means that no prefixes will be advertised containing previously non-routable sub-prefixes, meaning that if weak consistency is broken, so too is strong consistency.

Task 2: Sustainable networking**17 Points****a) Analysing sustainability numbers****(10 Points)**

Electric cars are commonly advertized with a carbon footprint of 0 gCO₂e/km, alike in the example shown in Figure 12. This is not factually wrong. However, it is only correct under some hypothesis on the scope and type of analysis considered.

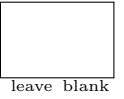


Figure 12: An electric car ad (cropped)

- (i) What type of analysis leads to an electric car's carbon footprint of 0 gCO₂e/km? If there are several correct answers possible, tick them all. (3 Points)

Solution:*Dimensions considered*

- | | | |
|---|-----------------------------------|----------------------------------|
| ☒ Operational | ☐ Embodied | |
| ☒ Scope 1 | ☐ Scope 2 | ☐ Scope 3 |

Type of analysis

- | | |
|---|---|
| ☒ Attributional | ☒ Consequential |
|---|---|

Grading: 1pt per line entirely correct. 0pt otherwise.

If the first two lines are correct, then “Attributional” must be ticked as well for a full mark. If the Embodied, Scope 2 or Scope 3 are ticked, then ticking “Attributional” is also wrong.

- (ii) One other important methodological decision in sustainability studies is to use either a top-down or a bottom-up approach. Consider a network switch and the task of estimating the average energy cost of forwarding 100 GB of data from an input port A to an output port B. How would you measure that energy cost using a... (4 Points)

Solution: A top-down approach would be measuring the entire power draw of the switch and divide it by the amount of traffic forwarded to derive a rate in W/Gbps, which is equivalent to Ws/Gb, or J/Gb. Multiplying that rate by 8*100 will give the energy consumed for forwarding 100GB.

A bottom-up approach would measure the power draw of the basic components involved in receiving the bits, processing the header, copying over the bits in the output buffer, etc.

Grading:

2 points for each approach. 1 point if the explanation is half-baked. 0+ for suggesting to measuring the energy cost of a single packet (without more info) for bottom-up.

- (iii) Imagine you study the potential energy savings of streaming video in HD instead of 4k. Which approach is most appropriate here? Tick the corresponding box. (1 Point)

☐ Top-down ☒ Bottom-up

Grading: 1 point if correct.

- (iv) Explain what kind of error you would be making by selecting the wrong approach. (2 Points)

Solution: The top-down approach includes all the idle power costs (cooling, power conversion, etc) which do not scale with the amount of traffic forwarded. Thus, it will overestimate the energy savings resulting from reducing the traffic volume by banning 4k.

Grading: 2 points for a valid explanation. 1 point if the explanation is half-baked.

1 point if it mentions that bottom-up underestimates the total energy.

2 minus points if arguing it needs consequential and top-down would be attributional.

b) Power proportionality in wired networks (4 Points)

Power proportionality refers to the concept of power draw scaling linearly with the utilization level of a device. As seen in class, most wired networking equipment shows little proportionality; that is, the idle power (at 0% utilization) is a large fraction of the maximal power (at 100% utilization), as illustrated in Figure 13.

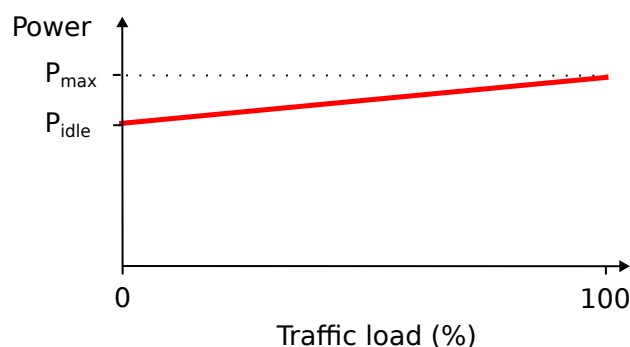
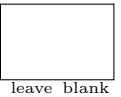


Figure 13: The typical power profile of a wired network device displays little proportionality.

- (i) Give one example of an existing feature in consumer electronics (i.e., laptops, smart-phones, ...) which fosters power proportionality. (1 Point)

Solution: Turning off screen, CPU power modes, ...

Grading: 1 point for any valid example.

0.5 pt for "sleeping to save power" or "turning off parts" without concrete examples.

0 pt for just "sleeping."

- (ii) Give two different features we could envision in wired networks to foster power proportionality. (2 Points)

Solution: Link sleeping, Turning off redundant PSUs

Grading: 1 point for each valid example, whether it was covered in class or not.

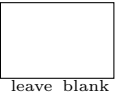
- (iii) Why is having routers with good power proportionality particularly important for ISP networks? (1 Point)

Solution: Because ISP network routers are often running at low utilization, with poor power proportionality, most of the energy is spend on powering the infrastructure without doing much useful work. That is, the energy efficiency is poor.

Grading: 1 point for a correct explanation.

c) Promise and pitfalls of energy efficiency

(3 Points)



- (i) In the lecture, we argued that, from a sustainability perspective, one should rather aim to reduce absolute energy usage than improving energy efficiency. Why? (2 Points)

Solution: Energy efficiency can be improved by pulling more power to reach proportionally higher speed. This was typically the case of the Ethernet standard speeds. Ultimately, the important point for sustainability is to cap the total amount of resources used. Efficiency may be a mean to achieve that, but that is not the goal.

NOTA BENE Many students understood the question at the systemic level, and thus talked about the Jevon's paradox. I should have made it more clear that I meant here a discussion at the device level.

Grading:

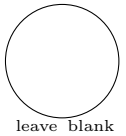
- 1 point for mentioning that higher efficiency may come at the cost of higher power.
- 1 point for mentioning that reducing/capping the absolute resource usage is the goal for sustainability.
- 1 point for mentioning Jevons' paradox and a minimal explanation.
- 1 point for any other valid point I cannot think of right now.

Maximum 2 points.

- (ii) In certain cases, an improvement in energy efficiency may lead to an increase in energy usage. How is this effect called? (1 Point)

Solution: The rebound effect. Jevon's paradox is also accepted.

Grading: 1 point for a correct answer.

**Task 3: Programmable data planes****15 Points**

In modern AI training clusters, large model parameters are exchanged between GPUs across a leaf-spine network fabric. These *elephant flows* can cause congestion when ECMP hashing maps multiple large flows to the same path.

You are tasked with analyzing a P4 program that implements **load-aware packet spreading** using the *power-of-two-choices* technique: for each packet, the switch samples two candidate paths, compares their current load, and forwards the packet on the less loaded path.

To avoid excessive path switching when loads are similar, the design uses a **minimum delta threshold**: path 2 is chosen only if its load is sufficiently lower than path 1.

Assume the switch has 8 uplink ports (numbered 0–7) to spine switches, and these numbers are the actual egress port identifiers.

The following P4 snippet implements this logic; in the questions below you will analyze it.

```
1  struct metadata_t {
2      bit<3>  path1;
3      bit<3>  path2;
4      bit<3>  selected_path;
5      bit<32> load1;
6      bit<32> load2;
7  }
8
9  control MyIngress(inout headers hdr,
10                  inout metadata_t meta,
11                  inout standard_metadata_t standard_metadata) {
12
13      register<bit<32>>(8) path_load;
14
15      const bit<32> MIN_DELTA = 5;
16
17      action sample_two_paths() {
18          hash(meta.path1, HashAlgorithm.crc16, (bit<32>)0,
19              { hdr.ipv4.srcAddr, hdr.ipv4.dstAddr,
20                hdr.tcp.srcPort, hdr.tcp.dstPort }, (bit<32>)8);
21          hash(meta.path2, HashAlgorithm.crc32, (bit<32>)0,
22              { hdr.ipv4.srcAddr, hdr.ipv4.dstAddr,
23                hdr.tcp.srcPort, hdr.tcp.dstPort }, (bit<32>)8);
24      }
25
26      action read_path_loads() {
27          path_load.read(meta.load1, (bit<32>)meta.path1);
28          path_load.read(meta.load2, (bit<32>)meta.path2);
29      }
30  }
```

```
31     action select_path() {
32         if (meta.path1 == meta.path2) {
33             meta.selected_path = meta.path1;
34         } else if (meta.load2 + MIN_DELTA < meta.load1) {
35             meta.selected_path = meta.path2;
36         } else {
37             meta.selected_path = meta.path1;
38         }
39     }
40
41     action update_path_load() {
42         bit<32> current_load;
43         path_load.read(current_load, (bit<32>)meta.selected_path);
44         path_load.write((bit<32>)meta.selected_path, current_load + 1);
45     }
46
47     action set_egress_port() {
48         standard_metadata.egress_spec = (bit<9>)meta.selected_path;
49     }
50
51     apply {
52         if (hdr.ipv4.isValid() && hdr.tcp.isValid()) {
53             sample_two_paths();
54             read_path_loads();
55             select_path();
56             update_path_load();
57             set_egress_port();
58         }
59     }
60 }
```

- (a) Suppose the `path_load` register currently holds the packet counts shown below.

Index	0	1	2	3	4	5	6	7
Load	40	25	62	30	48	53	61	35

A valid TCP/IPv4 packet arrives with hash outputs `path1 = 2` and `path2 = 5`.

Trace the packet through the ingress pipeline: list the actions in order, the key values read or computed, and the decision taken. State the final egress port and the updated register state. (6 Points)

Solution:

- **Validity Check**
`hdr.ipv4.isValid()` and `hdr.tcp.isValid()` are true, so processing proceeds.
- **Sample Two Paths**
`meta.path1 = 2` and `meta.path2 = 5`.
- **Read Path Loads**
`meta.load1 = path_load[2] = 62`
`meta.load2 = path_load[5] = 53`
- **Select Path**
`path1 == path2` is false.
`load2 + MIN_DELTA < load1`: $53 + 5 = 58 < 62$ is true, so `selected_path = 5`.
- **Update Path Load**
`path_load[5]` updates from 53 to 54. Other entries are unchanged.
- **Set Egress Port**
`egress_spec = 5`.

Final egress port: 5 Updated register state:

Index	0	1	2	3	4	5	6	7
Load	40	25	62	30	48	54	61	35

Grading: 1 point each for: correct action sequence, correct load reads, collision check, strict inequality test, correct selection/update, correct egress port and final table. Partial credit for minor mistakes.

- (b) The current implementation counts *packets* sent on each path. However, in AI training workloads, packet sizes vary significantly—small ACKs and control messages vs. large gradient updates. Additionally, the uplinks have different capacities: ports 0–3 are 200 Gbps and ports 4–7 are 100 Gbps.

Explain why packet counting can lead to poor load balancing. Then, propose a modification that makes the tracking *byte-aware* and *link-capacity-aware*. Assume `hdr.ipv4.totalLen` is available and in bytes. Provide a short pseudo-code snippet for the modified logic and an integration note. (5 Points)

Solution: Problem with packet counting: Two paths can have the same packet count but very different byte loads. The counters would look equal even if one path carries far more traffic, so the selection can keep sending large packets to a saturated link.

Proposed modification (byte-aware + link-capacity-aware): Increment by the size of the packet in bytes, i.e. `hdr.ipv4.totalLen` instead of 1. Then compare *weighted* loads to account for link capacity (e.g., treat 100 Gbps ports as having double the cost per byte compared to 200 Gbps ports, which can be implemented with a left shift).

```

action update_path_load() {
    bit<32> current_load;
    path_load.read(current_load, (bit<32>)meta.selected_path);
    path_load.write((bit<32>)meta.selected_path,
        current_load + (bit<32>)hdr.ipv4.totalLen);
}

```

Example link-capacity-aware comparison:

```

// Per-port weight: 200G -> 1, 100G -> 2
// Assume port_weight is initialized by the control plane.
register<bit<16>>(8) port_weight;

action select_path() {
    bit<16> w1;
    bit<16> w2;
    port_weight.read(w1, (bit<32>)meta.path1);
    port_weight.read(w2, (bit<32>)meta.path2);
    bit<48> cost1 = (w1 == 2) ? ((bit<48>)meta.load1 << 1) : (bit<48>)meta.load1;
    bit<48> cost2 = (w2 == 2) ? ((bit<48>)meta.load2 << 1) : (bit<48>)meta.load2;
    if (meta.path1 == meta.path2) {
        meta.selected_path = meta.path1;
    } else if (cost2 + MIN_DELTA < cost1) {
        meta.selected_path = meta.path2;
    } else {
        meta.selected_path = meta.path1;
    }
}

```

Integration: The byte-aware change is localized to `update_path_load()`. Link-capacity awareness is handled in `select_path()` via weighted comparisons (or load/capacity normalization). `MIN_DELTA` must be recalibrated to match the new units (bytes or weighted bytes).

Grading: 2 points for the packet-vs-byte explanation, 1 point for a correct byte-aware and link-capacity-aware idea, 1 point for matching pseudo-code, and 1 point for the integration note (including `MIN_DELTA` reinterpretation).

- (c) Consider the *byte-aware, link-capacity-aware* design from (b). Identify **two** limitations or potential problems. One must relate to packet reordering or flow consistency; the other must relate to stale/incorrect load signals (e.g., no decay or no congestion feedback). For each, propose a mitigation strategy and discuss trade-offs. (4 Points)

Solution:

Expected limitations (one from each category; any valid stale/incorrect-load issue is acceptable):

1. Out-of-order delivery

Problem: Packets from the same flow may take different paths and arrive out of order.

Mitigation: Use flowlet switching (only change paths after an idle gap), or per-flow stickiness.

Trade-off: Reduced load-balancing granularity and more state.

2. No congestion feedback

Problem: Counters track traffic sent, not actual downstream congestion.

Mitigation: Incorporate ECN/INT feedback or queue depth signals.

Trade-off: More complexity and signaling overhead.

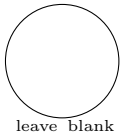
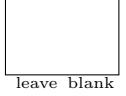
3. Counter growth / stale history

Problem: Counters accumulate indefinitely and can become stale or overflow.

Mitigation: Periodic decay/reset, or sliding-window counters.

Trade-off: Additional control-plane involvement or state.

Grading: 1 point for a valid flow-consistency limitation, 1 point for a valid stale/incorrect-load limitation, and 1 point each for the corresponding mitigation with trade-off. Total: 4 points.

**Task 4: Routing Algebra and Network Verification****20 Points****a) Warm-up****(4 Points)**

The following questions consider a routing algebra $(A, \oplus, \preceq)$.

- (i) Let $x \in A$ be an attribute for which $\forall a \in A: a \preceq x$, and $x = x \oplus a$. Give an interpretation of what x can represent both when it is “selected” as a route by a node and when it is “assigned” to an edge. (1 Point)

Solution:

- **Node:** The absence of a route, a black hole, an invalid route; or the worst-possible route.
- **Edge:** Link being down; or deactivated; or an access-list / control-list denying paths on that edge.

Grading: Half a point for each part of the answer. Examples from specific routing algebras are also accepted.

- (ii) Let $x \in A$ be an attribute for which $\forall a \in A: x \preceq a$, and $a = x \oplus a$. Give an interpretation of what x can represent both when it is “selected” as a route by a node and when it is “assigned” to an edge. (1 Point)

Solution:

- **Node:** A router selects a local route; or one that is always preferred.
- **Edge:** A transparent link; a logical link; a possible cycle with zero length; a neutral attribute.

Grading: Half a point for each part of the answer. Examples from specific routing algebras are also accepted.

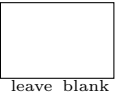
- (iii) Recall that an algebra $(A, \oplus, \preceq)$ is inflationary if $\forall a, b \in A, a \preceq b \oplus a$. Similarly, the algebra is strictly inflationary if $\forall (a, b) \in A, a \prec b \oplus a$. Say you use an SMT solver to find a solution to the stable state problem. What issue can arise for a non-strictly inflationary algebra that cannot occur for strictly inflationary algebras? (2 Points)

Solution: Possible zero-length cycle: The solver might find solutions (i.e. return SAT) that are unreachable in practice.

Grading: 2 points for mention of zero-length cycles. 1 point for mentioning zero lengths paths or for hinting at paths being able to grow indefinitely. 1/2 points for a mention of solutions to the stable state equations that are unreachable in practice. -1 point for saying strictly inflationary algebras might not have a stable state.

b) Modeling RIP

(5 Points)

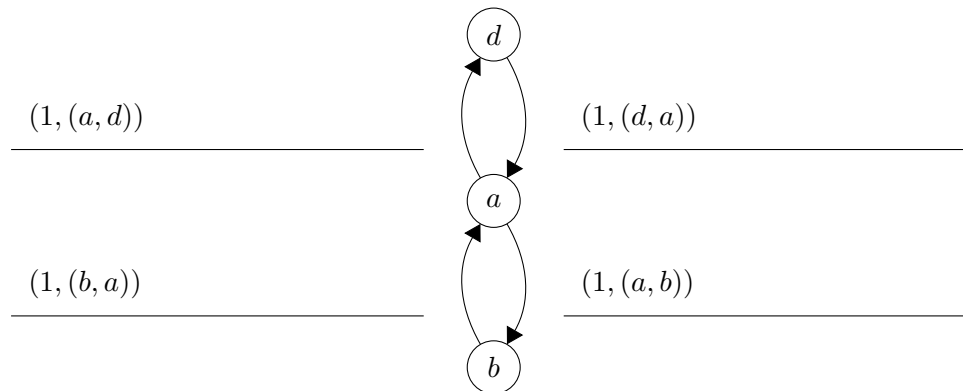


The *Routing Information Protocol (RIP)* is a distance-vector routing protocol with the following properties:

- *Minimum-hop routing*: It employs hop count as a routing metric.
- *Route poisoning*: It limits the hop count to a maximum of 15. A hop-count of 16 indicates that the destination is unreachable.
- *Split horizon*: Routers never advertise their path to their next-hop (i.e., if router A uses B to reach destination d , it does not advertise a route for d to B).

We model RIP as a routing algebra $(A, \oplus, \preceq)$ with attributes $A = \{0, \dots, 16\} \times N^2$. Attribute $(d, (u, v)) \in A$ contains the hop count d and the *next edge* (u, v) on the path (where u is the source and v is the destination of that edge).

- (i) Given the following graph with three nodes and 4 (directional) edges, label all edge with the attribute according to the definition of A stated above. (2 Points)



Grading: 0.5 point per correctly labelled edge. -0.5 if all edges are correct but place on the opposite edge. $+0.5$ if all edges have distance 1.

- (ii) Define the $\preceq$ and $\oplus$ operators to complete the routing algebra for RIP. (3 Points)

Solution: $(d_1, (u_1, v_1)) \preceq (d_2, (u_2, v_2)) \iff d_1 \leq d_2$

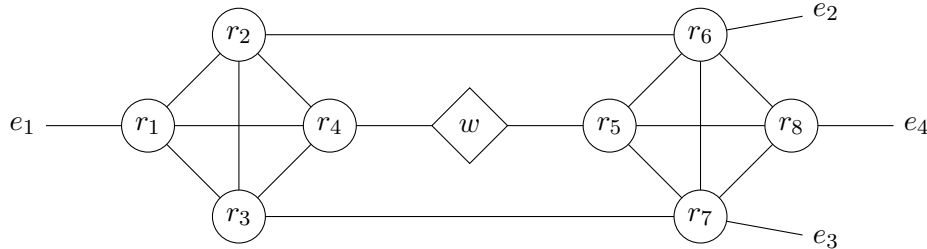
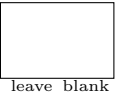
$$(d_1, (u_1, v_1)) \oplus (d_2, (u_2, v_2)) = \begin{cases} (d_1 + d_2, (u_1, v_1)) & \text{if } d_1 + d_2 < 16 \text{ and } u_1 \neq v_2 \\ (16, (-, -)) & \text{otherwise} \end{cases}$$

Grading: 0.5 point for each of: $d_1 + d_2$, (u_1, v_1) , case distinction on $d_1 + d_2 < 16$, case distinction on $u_1 \neq v_2$.

c) Waypoint specifications

(11 Points)

You are given the following network $G = (N, E)$ with internal routers $\{r_1, \dots, r_8, w\} \in \text{int}$ and external routers $\{e_1, \dots, e_4\} \in \text{ext}$, with one firewall w :



You try to verify waypoint properties in your network that uses any routing algebra $(A, \oplus, \preceq)$ to forward traffic along optimal paths. All links are bidirectional. The algebra is such that any optimal path is acyclic. Other than that, you cannot assume anything about the algebra.

The verifier uses the same equations as we have seen in the lecture. As a reminder, they are:

$$\begin{aligned}
 & \bigwedge_{e \in \text{ext}} E_d(e) \in A \\
 & \bigwedge_{u \in V \setminus \text{ext}} \bigwedge_{v \in u_{\text{out}}} \begin{cases} E_d(u) = A(uv) \oplus E_d(v) & \text{if } \text{from}(u) = v \\ E_d(u) \preceq A(uv) \oplus E_d(v) & \text{otherwise} \end{cases} \\
 & \bigwedge_{u \in V \setminus \text{ext}} \text{from}(u) \in u_{\text{out}} \\
 & \wedge \neg \phi
 \end{aligned}$$

- (i) Write a specification ϕ_1 to that ensures w is on r_4 's optimal path. ϕ_1 needs only to work for the given network G . (1 Point)

Solution: $\phi_1 = (\text{from}(r_4) = w)$

Grading: Note, that in the absence of a path at r_4 , it would still force $\text{from}(r_4) = w$, which would no longer be meaningful. We mark full points to answers that ignore this detail. In the case that r_5 is written instead of w , we subtract 0.5 points (only once if the mistake is repeated in the next subtasks).

- (ii) Write a specification ϕ_2 that ensures w is on the optimal path of r_1, r_2, r_3 , and r_4 . ϕ_2 needs only to work for the given network G . (3 Points)

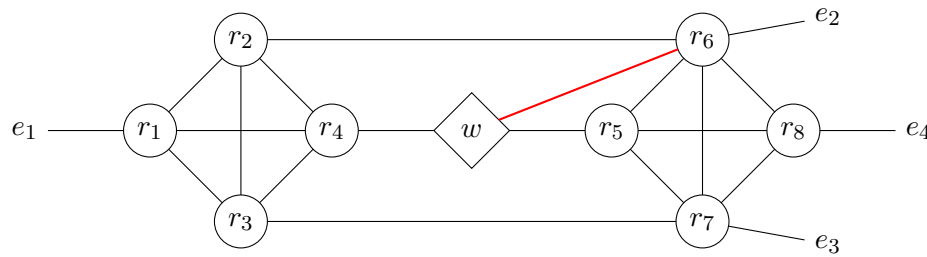
Solution: $\phi_2 = (\text{from}(r_1) \neq e_1 \wedge \text{from}(r_2) \neq r_6 \wedge \text{from}(r_3) \neq r_7)$

An other valid solution is:

$\phi_2 = (\text{from}(r_1) \in \{r_2, r_3, r_4\} \wedge \text{from}(r_2) \in \{r_1, r_3, r_4\} \wedge \text{from}(r_3) \in \{r_1, r_2, r_4\})$

Grading: 1 point for the specification being satisfied for each of r_1, r_2, r_3 . -1 point per extra egress point the traffic can leave from (e_1, r_6, r_7). Since we did not mention that the specification should not more than what it is required to do, we also accept over-constraining solutions.

From now on, assume there exists an additional link between w and r_6 . As a reference, here is the same network $G' = (N, E')$, including the additional link:



- (iii) Explain why it is now much more difficult to write a specification that ensures w to be on r_4 's optimal path. You do *not* need to write such a specification. (2 Points)

Solution: There now exist a large number of valid paths that satisfy the intent. The specification would need to enumerate them all.

Long version (not necessary to get full points): Before, the only acyclic path from r_4 via w was when link (r_4, w) was on its optimal path. If instead (r_5, w) was on the path, then the next link is either (w, r_4) or (w, r_5) , both would create a cycle. Yet, with the addition of link (w, r_6) , a path like $[r_4, r_2, r_6, w, r_5, r_8, e_4]$ is acyclic and complies with the intent.

Grading: 1.5 point for mentioning an example with which the old spec does not hold or explaining why such examples exist. 0.5 points for mentioning that there are many such counter examples.

- (iv) You still want to verify such waypoint properties, and be able to do so in an arbitrary network. Design a routing algebra $(A', \oplus', \preceq')$ that will always select the same paths as $(A, \oplus, \preceq)$, but enables us to verify waypoint specifications in arbitrary networks.

Hint: Reuse $(A, \oplus, \preceq)$ and add additional information to the attributes. (4 Points)

Solution: We explicitly encode the set of all nodes traversed on a path using the powerset $P(N)$. (Recall, that the powerset $P(N)$ is the set of all possible subsets of N .)

$$\begin{aligned} A' &= A \times P(N) \\ (a, p_a) \oplus' (b, p_b) &= (a \oplus b, p_a \cup p_b) \\ (a, p_a) \preceq' (b, p_b) &\iff a \preceq b \end{aligned}$$

An alternative solution is to use a flag $\{\top, \perp\}$ to represent whether the path has already traversed w :

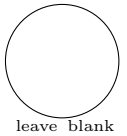
$$\begin{aligned} A' &= A \times \{\top, \perp\} \\ (a, f_a) \oplus' (b, f_b) &= (a \oplus b, f_a \vee f_b) \\ (a, f_a) \preceq' (b, f_b) &\iff a \preceq b \end{aligned}$$

Grading: 1 point for correct idea, 1 point per each correct definition of $A, \oplus, \preceq$

- (v) With your new algebra $(A', \oplus', \preceq')$, write a specification ϕ_3 that checks if w is on r_4 's optimal path, without assuming anything about the network. (1 Point)

Solution: $\phi_3 = (w \in p_{r_4})$, where $(-, p_{r_4}) = E_d(r_4)$.

If using a flag f to encode whether the path traversed w , the solution is: $\phi_3 = f_{r_4}$ where $(-, f_{r_4}) = E_d(r_4)$.

**Task 5: Transport Protocols****18 Points****Transport for Satellite networks**

In class, we looked at how we can optimize transport on the Internet and in the Datacenter. In this question, you will have to rely on similar principles in picking a suitable transport protocol, loss recovery scheme, and congestion control algorithm for a satellite network.

Consider a Low Earth Orbit (LEO) satellite constellation with the following characteristics:

1. End-to-end RTT: **40–80 ms**, depending on satellite position.
2. Downlink bottleneck capacity: **100 Mbps**.
3. Packet losses occur due to:
 - (a) Handovers between satellites.
 - (b) Temporary link outages (blackout periods) lasting 200–800 ms.
4. Buffers: satellite networks have **shallow/small** buffers to minimize queueing delay.

Assume that this network is going to support regular Internet traffic, which has a typical flow size distribution (expressed in number of packets) as follows:

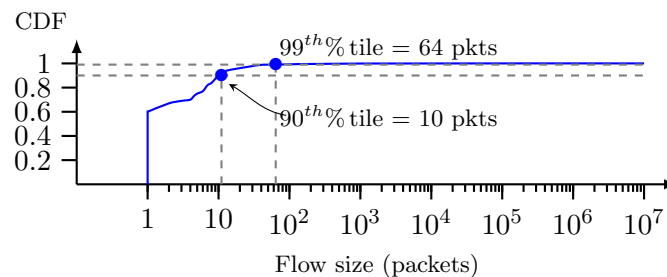
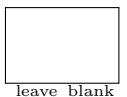


Figure 14: CDF of flow lengths for 60mins of CAIDA Internet traces

a) Picking a Transport Stack**(6 Points)**

- (i) Which transport protocol between TCP and QUIC would you recommend to use in this network? Justify your choice by explicitly discussing:
 1. The connection setup overhead; and
 2. The sensitivity to RTT and RTT variability.

(3 Points)**Solution:**

Given the long RTTs and the typical short flow size on the Internet, QUIC would a natural protocol of choice for this network. Short flows only last a few RTTs, which makes optimizing the connection setup time critical. Further, since the RTTs in these networks are already long, saving a few round trips during connection set up (compared to TCP) can greatly improve performance.

Grading:

- +1pt for picking QUIC;
- +1pt for reasoning about the connection setup overhead;
- +1pt for reasoning about the long RTTs;

- (ii) Would you consider enabling TSO and GRO on this satellite network? Discuss how it would interact with:

1. Higher than usual loss rates seen in satellite networks; and
2. Shallow/small buffers.

(3 Points)

Solution:

While TSO and GRO can improve throughput in theory, large segments mean that if a packet is lost, the entire large segment (with multiple regular-sized packets worth of data) needs to be retransmitted. Therefore, in a lossy network, TSO and GRO can be costly. Moreover, TSO and GRO can produce bursty traffic, that is something that might not play well with the shallow buffers typically seen in satellite networks. For these reasons, TSO and GRO are better disabled in this network.

Grading:

- +1pt for reasoning about TSO and GRO;
- +1pt for identifying larger segments incur higher retransmission constellations;
- +1pt for identifying bursty TSO traffic doesn't go well with shallow buffers;

b) Loss-recovery with *probe packets*

(6 Points)

Throughput in satellite networks often suffers due to frequent packet losses. This makes minimizing the cost of loss detection and recovery critical, especially because of the large RTTs.

Some proposals suggest that pairs of endpoints connected over a satellite link continuously send low-priority probe packets, even when no application is actively sending traffic. These packets are acknowledged and carry sequence numbers as well, just like normal data packets but without the payload. The purpose of this passive, long-lived, and low-priority connection is to provide regular timely feedback about the network conditions, especially during the first and last few RTTs of a flow.

- (i) Describe how this stream of low priority probe packets could be leveraged to design a better loss recovery scheme compared to standard fast retransmits and RTOs. Consider its impact on tail-loss detection and reordered packets. (3 Points)

Solution:

The regular stream of probe packets allow the transport layer to have regular feedback, even when it is application limited during the first and last few RTTs of a flow. It can therefore help with tail loss detection, for example, similar to how RACK-TLP uses a tail-loss probe to detect loss. Since they have sequence numbers, they can also signal reordering in the network to the transport layer.

Grading:

- +1pt quicker feedback of congestion or blackout periods;
- +1pt better tail loss detection;
- +1pt better reordering detection;

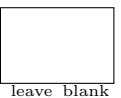
- (ii) Given the steady stream of feedback from the probe packets, does this protocol with probe packets still need to rely on RTOs (Retransmission Time Out)? Why or why not? (3 Points)

Solution:

Essentially, the probe packets can help remove all the 'false' alarms that can cause unwarranted ReTX and RTOs. However, an RTO will still be needed to detect blackout periods, link failures, and handovers.

Grading:

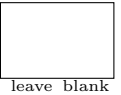
- +1.5pt no for tail loss and reordering;
- +1.5pt yes for blackout period and handovers;



c) Congestion Control

(6 Points)

Since most of the flows in our network are short flows, we are going to focus on optimizing start up behavior of our congestion control algorithm - ie the Slow Start. To begin with, consider the standard standard slow start algorithm which sends two packets for every ACK it receives for its data packets.



- (i) Consider a start window of 10 packets and a packet size of 1,500 bytes. How many RTTs will it take for the sender to saturate a 96 Mbps bottleneck link on a 50 ms RTT link (Bandwidth Delay Product (BDP) = 400 packets)? (2 Points)

Solution:

Since the BDP is 400 packets, we need to calculate how many RTTs it will take for the `cwnd` to grow larger than 400 packets.

During slow start, the `cwnd` grows exponentially:

$10 \rightarrow 20 \rightarrow 40 \rightarrow 80 \rightarrow 160 \rightarrow 320 \rightarrow 640$

It takes 7 RTTs to saturate the link.

Grading:

+1pt for modeling exponential growth;

+1pt for getting the numbers right;

- (ii) Large RTTs mean that we can't waste a lot of time ramping up to the bottleneck link rate. Consider a modified slow start mechanism called *Sudden Start*.

The sender will send two packets for every ACK it sees for both its own data packets, **as well as** the probe packets being sent in the background. Assume that there are no losses and the probe packets are being sent (and their ACKs are being received) every 5 ms. You can also assume that the probe packets consume a negligible amount of bandwidth at the bottleneck. How many RTTs does the sender now take to saturate the same 96 Mbps bottleneck link on a 50 ms RTT path? (4 Points)

Solution:

This time, the sender sends two packets for every ACK it sees, as well the probe packets whose ACKs arrive every 5 ms (10 times per RTT).

Therefore, the `cwnd` will grow as follows:

$10 \rightarrow 2 \cdot (10 + 10) = 40$

$\rightarrow 2 \cdot (40 + 10) = 100$

$\rightarrow 2 \cdot (100 + 10) = 220$

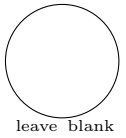
$\rightarrow 2 \cdot (220 + 10) = 460$

It takes 4 RTTs to saturate the link.

Grading:

+1pt for factoring in 10 additional packets per RTT;

+3pt for getting the numbers right;

**Task 6: Networking for ML****15 Points**

In the course we introduced Ring AllReduce, an AllReduce shape executed in 2 phases of $|devices| - 1$ stages each. Phase 1 consists of sending partial sum (our reduction operator in this task) over partitions of the tensor, while Phase 2 progressively gathers the full sums of each partition.

We now consider a *2-dimensional* Ring AllReduce. Let us order the devices into a grid with r rows and c columns. The 2-dimensional Ring AllReduce executes in three steps:

Step 1. Phase 1 of 1-dimensional Ring AllReduce on the rows.

Step 2. Phase 1 *and* 2 of 1-dimensional Ring AllReduce on the columns.

Step 3. Phase 2 of 1-dimensional Ring AllReduce on the rows.

- (i) Figures 15 (corresponding to the initial state) to 19 illustrate the execution of a 2D Ring AllReduce with 4 devices ($r = 2, c = 2$). **We ask you to fill in the dotted gaps with the correct values.** (5 Points)

Note: Each 1D Ring AllReduce phase only lasts 1 stage as there are only two devices involved.

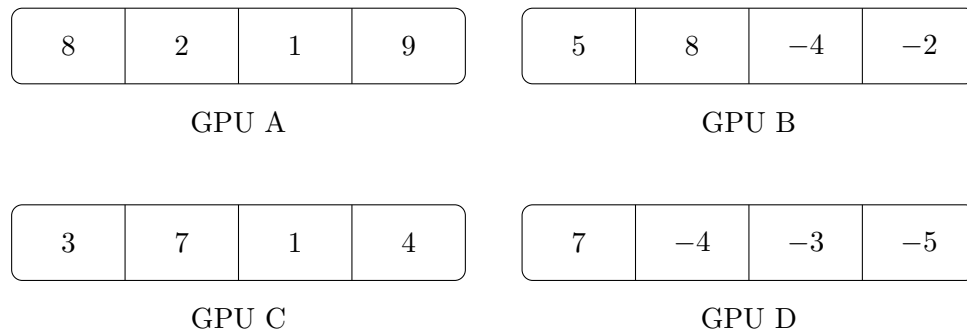


Figure 15: Initial state.

We start with **Step 1**: Phase 1 of r 1-dimensional Rings *in parallel*, with c participants each. The input is the whole tensor, and each flow sends one c -th of the tensor in every stage. At the end, each device knows the row-wise sum of one c -th of the tensor (half in this example).

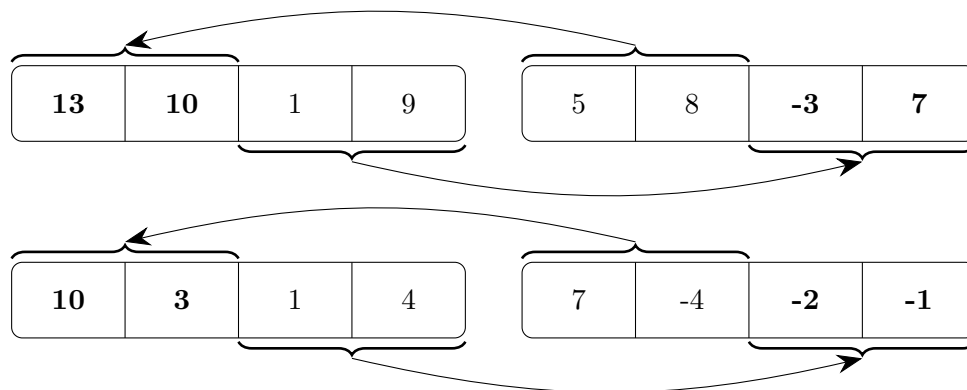


Figure 16: After phase 1 of row-wise Ring AllReduce (stage 1). Fill me in.

We continue with **Step 2**: c 1-dimensional Rings *in parallel*, with r participants each. The input is one c -th of the tensor, corresponding to the column index (half the tensor in this example); each flow sends one r -th of the subtensor at a time, i.e. one $(r \cdot c)$ -th of the full tensor (a fourth in this example). At the end, each device knows the full sum of one c -th of the tensor (half in this case).

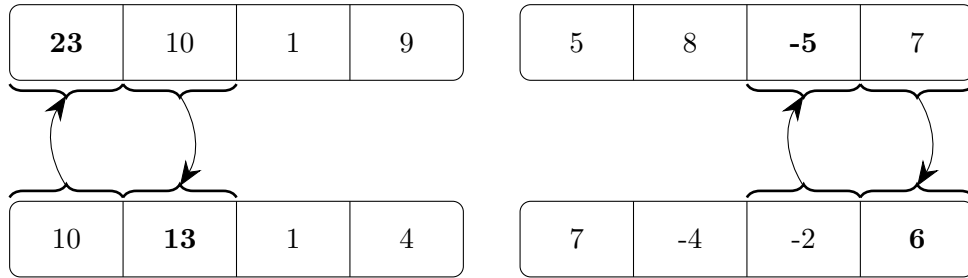


Figure 17: After phase 1 of column-wise Ring AllReduce (stage 2). Fill me in.

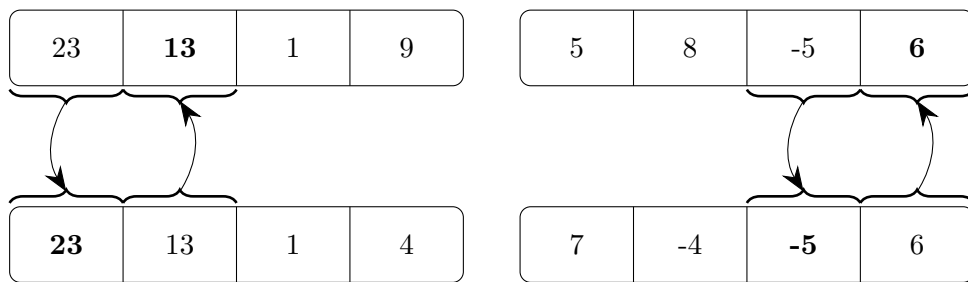


Figure 18: After phase 2 of column-wise Ring AllReduce (stage 3). Fill me in.

Finally, **Step 3**: Identical to Step 1, but we now gather results instead of performing reductions. At the end, each device knows the full sum of the full tensor.

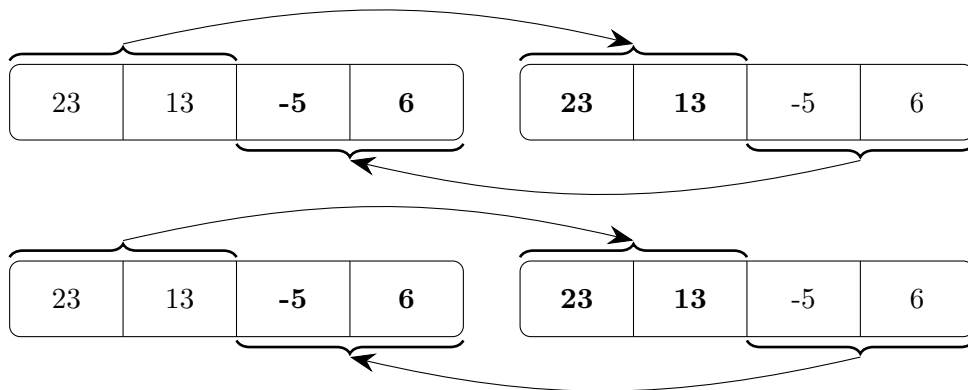


Figure 19: After phase 2 of row-wise Ring AllReduce (stage 4). Fill me in.

Grading: +1.5pt for step 1, +2.5pt for step 2, +1pt for step 3.

- (ii) We now ask you to analyse the 2D Ring AllReduce as done in class for the other AllReduce shapes. For reference, we include the table shown in the lecture: (6 Points)

Pattern	Stages	Total Data Sent	Max Incoming Flows
All-to-All	1	$N(N-1)M$	$N-1$
Parameter Server	2	$2(N-1)M$	$N-1$
Ring	$2(N-1)$	$2(N-1)M$	1
2D Ring	???	???	???

Assume the 2D grid has r rows and c columns. The full tensor's total number of bytes is M , which is a multiple of $r \cdot c$. Note that operations performed in parallel count as a single one for what concerns the total number of stages. In Figures 15 to 19, 4 stages were performed in total.

It is not necessary to explain your reasoning, but you may receive partial points if you do so in case the final answer is incorrect.

Hint: Note that there are effectively two 1D Ring AllReduces being performed, one across rows and one across columns, with different input sizes.

Stages (as a function of r and c): $2(r-1) + 2(c-1)$

Total Data Sent (as a function of r , c and M): $2r(c-1)M + 2(r-1)M$

Max (Simultaneous) Incoming Flows: 1

Grading: +1pt if at least one of the first two questions right. +2pts for each of the first 2 questions right, +1pt for incoming flows right. Forgetting the 'r' in the Total Data Sent question is -1pt.

- (iii) Consider a traffic-agnostic round-robin (RR) optical circuit-switched (OCS) core between the four devices shown in Figure 15. Each device has one transmit port and one receive port. The round-robin (RR) circuit schedule between these four devices is shown in Table 1. The schedule repeats after T3.

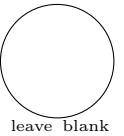
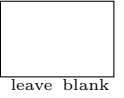
	A	B	C	D
T1	B	C	D	A
T2	C	D	A	B
T3	D	A	B	C

Table 1: Round-robin schedule of the OCS core.

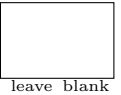
Consider one hop routing between the devices; that is, device A can only send data packets to device B during those time slots when they have a direct circuit according to the RR schedule (there is no Valiant Load Balancing). Assume that time slots are longer than the flow completion times; that is, one time slot is sufficient for a device to send more than a whole tensor's worth of data.

	A	B	C	D	Step	Circuits
T1	B	C	D	A	1	A→B, C→D
T2	C	D	A	B		
T3	D	A	B	C	1	B→A, D→C
T4	B	C	D	A		
T5	C	D	A	B	2	A→C, B→D, C→A, D→B
T6	D	A	B	C	3	B→A, D→C
T7	B	C	D	A	3	A→B, C→D
T8	C	D	A	B		
T9	D	A	B	C		
T10	B	C	D	A		
T11	C	D	A	B		
T12	D	A	B	C		

Table 2: Four cycles of the round-robin schedule of the OCS core.

**Task 7: ML for Networking****10 Points****a) Learning-Based Control in Networking****(5 Points)**

- (i) How does BBA (a classical, ML-free Adaptive Bitrate (ABR) algorithm) infer the current capacity of a network? (2 Points)
- (ii) Name two important challenges that ABR algorithms encounter while trying to optimize Quality of Experience (QoE) over the Internet? How can learning-based approaches help alleviate them? (2 Points)
- (iii) Conversely, name two risks stemming from employing learning-based methods. (1 Point)

b) Sensible Use of LLMs in Network Operations**(5 Points)**

- (i) Batfish (a formal model-based network verification tool) can verify whether a network will behave as expected after a configuration update. However, even when using a verifier, unexpected issues may still arise. Give two possible reasons for the verifier's inaccuracy.

(2 Points)

Solution: Features or protocols may be present in the network that are not supported by the verifier; the verifier might not model device behaviour with perfect fidelity.

- (ii) Imagine that the log parsers currently used to monitor network devices will not be able to handle the new log formats introduced by an OS update. You could replace the existing parsers with a large language model (LLM) to handle each log without expecting a specific format. Propose an improvement to that solution that will greatly improve its scalability.

(1 Point)

Solution: Keep the existing parsers and only use LLMs for unknown parsers or do that and use LLMs to generate new parsers.

- (iii) You could prompt an LLM chatbot with *Show me the first 10 critical BGP errors that occurred in AS 65676 yesterday* and add the logs from that day into the prompt. Give two reasons why the LLM might return inaccurate or no results.

(1 Point)

Solution: The logs from an entire day may not fit in the context window.

The LLM may hallucinate details such as timestamps or fail to find specific entries, especially given the volume of input data.

- (iv) How could you improve accuracy for handling such requests, while still keeping the LLM-based interaction mode?

(1 Point)

Solution: The LLM should instead translate the natural-language request into tool calls (e.g., SQL queries) that deterministically retrieve the requested data.