CS 168 Spring 2026 — Introduction to the Internet Midterm

Solutions last updated: May 23, 2026

Print Your Name: _____

Print Your Student ID: _____

You have 110 minutes. There are 7 questions of varying credit. (100 points total)

Question:	1	2	3	4	5	6	7	Total
Points:	9	12	14	16	13	14	22	100

For questions with **circular bubbles**, you may select only one choice.

- Ⓐ Unselected option (Completely unfilled)
- Ⓑ Don't do this (it will be graded as incorrect)
- ● Only one selected option (completely filled)

For questions with **square checkboxes**, you may select one or more choices.

- ■ You can select
- ■ multiple squares
- ☑ Don't do this (it will be graded as incorrect)

Anything you write outside the answer boxes or you ~~cross out~~ will not be graded. If you write multiple answers, your answer is ambiguous, or the bubble/checkbox is not entirely filled in, we will grade the worst interpretation.

Honor Code: Read the honor code below and sign your name.

I understand that I may not collaborate with anyone else on this exam, or cheat in any way. I am aware of the Berkeley Campus Code of Student Conduct and acknowledge that academic misconduct will be reported to the Center for Student Conduct and may further result in, at minimum, negative points on the exam.

Sign your name: _____

Clarifications

Note: We had some newer members of staff working on issuing clarifications for this exam, so the wording and content of these clarifications is not indicative of how we usually issue clarifications. Notably, we usually would not issue clarifications 1, 2, 3, 4, 6, 7, 8, and 10 as they did not break the exam.

- Q3: FIFO stands for First-In First-Out
- Q3.2: The size of the packets that make up the queue of 300 Gb is irrelevant. You can assume any packet size you want, the answer will be the same.
- References to TCP in the reference card. We reused an earlier reference card. There are no questions on TCP in the exam itself.
- Q3.2: The queue of 300 Gb is for packets going from R5->R6. They only use the link from R5->R6 and no other link.
- Q3.3-3.4:

Q3.3: Typo in the answer choices:

$>$ should be $<$

Ⓐ $x < 1$	Ⓒ $x < 2$	Ⓔ $x < 3$	Ⓖ $x > 4$	Ⓘ $x < 5$
Ⓑ $x > 1$	Ⓓ $x > 2$	Ⓕ $x > 3$	Ⓗ $x > 4$	Ⓙ $x > 5$

Q3.4: Typo in the answer choices:

$>$ should be $<$

Ⓐ $y < 5$	Ⓒ $y < 7$	Ⓔ $y > 10$	Ⓖ $y < 14$	Ⓘ $y < 30$
Ⓑ $y > 5$	Ⓓ $y > 7$	Ⓕ $y > 10$	Ⓗ $y > 14$	Ⓙ $y > 30$

- Q3.2: The queue of 300 Gb is for packets going from R5->R6. They only use the link from R5->R6 and no other link. This data gets dropped at R6.
- Q4: The first advertisement gets sent out at t=2, as specified in the question.
- Q4.3: Cost 16 is considered infinite
- Q6.9: The last and second-last options are the same. That is, 729*3^26 and 3^32 are equal. If you think that is the answer, mark either one and you will get credit if it is correct.
- Q6.7: Fixed Bits in the table should be Fixed Trits.

Q1 *Potpourri* **(9 points)**

Q1.1 (2 points) In Project 1 (Traceroute), which is **not** a reason we send 3 probes for each TTL?

(A) Because probes could be dropped.

(B) Because responses to probes could be dropped.

(C) Because there could be multiple paths to the destination.

(D) Because there could be routing loops in the network.

Solution:

(A)-(B): True. Sending multiple probes decreases the chance that all probes/responses get dropped and we miss a router.

(C): True. Sending multiple probes increases the chance that we find all routers that are `i` hops away (where `i` is the TTL of the probe).

(D): False. Routing loops are unrelated to multiple probes at each TTL.

Q1.2 (2 points) In Project 1 (Traceroute), when a traceroute **probe** is sent across a link on the network, which headers are on the packet? Select all that apply.

[ ] A TCP header
[x] B UDP header
[x] C IP header
[ ] D ICMP header
[ ] E None of the above

Solution:

The traceroute probe is a UDP-over-IP packet. Note that the ICMP header is only used in the response to the probe, but not the probe itself.

Q1.3 (2 points) What does this line from Project 2 (Routing) do?

```
self.table[dst] = TableEntry(dst=dst,
                             port=self.table[dst].port,
                             latency=INFINITY,
                             expire_time=api.current_time()+self.ROUTE_TTL)
```

(A) Poisons an expired route.

(B) Sends a poison reverse advertisement.

(C) Accepts an advertisement from a router.

(D) Enables split horizon.

Solution:

(A) is correct. This line creates a new forwarding table entry with `latency=INFINITY`, which is used to poison an expired route.

(B) is false because no advertisement is sent here.

(C) is false because we would not hard-code the latency to infinity when accepting an advertisement.

(D) is false because split horizon involves sending an advertisement to everybody except the next-hop, but this code is not sending any advertisements.

Q1.4 (1 point) True or false: The modern internet uses circuit switching, because Layer 4 guarantees reliable delivery.

(A) True

(B) False

Suppose CodaBot and EvanBot are sharing a link with capacity 10 Gbps. CodaBot's peak demand is 6 Gbps, and EvanBot's peak demand is 7 Gbps. We have no other information about their demands.

Q1.5 (1 point) If we use statistical multiplexing to allocate bandwidth, is the 10 Gbps link sufficient to satisfy both CodaBot and EvanBot's demands?

(A) Always

(B) Sometimes

(C) Never

Solution:

If CodaBot and EvanBot's peaks occur at different times, and their non-peak demand is something low like 1 Gbps, then the 10 Gbps link is sufficient for both of their demands.

However, if the peak demand occurs at the same time, then the 10 Gbps link is insufficient.

Alternatively, if the non-peak demand is a high number like 5 Gbps, then the 10 Gbps link is also insufficient.

Q1.6 (1 point) If we **do not** use statistical multiplexing to allocate bandwidth, is the 10 Gbps link sufficient to satisfy both CodaBot and EvanBot's demands?

Ⓐ Always Ⓑ Sometimes Ⓒ Never

Solution:

Our intended answer was "**Never**."

Without statistical multiplexing, we would have to statically allocate bandwidth to CodaBot and EvanBot. To satisfy their demands, we need to allocate $6 + 7 = 13$ Gbps in total, so the 10 Gbps link is insufficient.

However, after the exam, some students offered an alternate interpretation of the question wording, where the answer would be "**Sometimes**".

If we do not use statistical multiplexing, we would have to offer a static amount of bandwidth to EvanBot and CodaBot, e.g. 5 Gbps per user. In this case, over time, the fixed 5 Gbps will sometimes satisfy EvanBot's demand (e.g. off-peak times), and sometimes not satisfy EvanBot's demand (e.g. peak times).

Since the wording of the question was not clear enough to point toward one interpretation, we gave students the benefit of the doubt and granted points for both "Sometimes" and "Never" as answers.

Some students also tried to argue that "Sometimes" is the right answer because EvanBot and CodaBot's peak demands might not occur at the same time (e.g. anytime one user uses the link, the other does not), which causes the link to never exceed its 10 Gbps capacity. **This is not a valid explanation**, because if we do not use statistical multiplexing, each bot must receive a static bandwidth allocation.

Therefore, even if the two users' peak demands don't overlap, a static allocation would mean that at some instant of peak demand, a user's demand is not being satisfied. For example, if we allocate 5 Gbps per user, at any instant of EvanBot's peak demand, EvanBot's demand will not be met, regardless of what is happening with the other 5 Gbps allocated to CodaBot.

Q2 *IP Header* **(12 points)**

Consider a link with MTU (maximum transmission unit) of 520 bytes. A router wants to send an IPv4 packet of size 8020 bytes (20 bytes of header and 8000 bytes of payload) along the link.

Q2.1 (2 points) If the router fragments this packet, what is the minimum number of fragments that must be sent along the link?

Ⓐ 2 Ⓑ 4 Ⓒ 8 Ⓓ 15 Ⓔ 16 Ⓕ 20

Solution:

There are 8000 bytes of payload, and each fragment fits 500 bytes of payload (520 byte MTU, minus 20 bytes of IP header).

Therefore, the total number fragments needed is $8000/500 = 16$.

Q2.2 (2 points) In total, how many extra bytes are sent along the link due to fragmentation (compared to sending the 8020 bytes as a single unit)?

Ⓐ 0 Ⓑ 20 Ⓒ 150 Ⓓ 300 Ⓔ 8000 Ⓕ 8020

Solution:

With or without fragmentation, the 8000 bytes of payload must be sent. The extra bytes come solely from the headers of the fragments sent.

Without fragmentation, a single 20-byte header is sent.

With fragmentation, 16 headers are sent, each of size 20 bytes, for a total of $16 \cdot 20 = 320$ bytes.

Therefore, $320 - 20 = 300$ extra bytes are sent.

Q2.3 (3 points) Which fields of the IP header are the same in every fragment of this packet? Select all that apply.

- [A] Source IP
- [B] Destination IP
- [C] Identification
- [D] Flags
- [E] Fragment Offset
- (F) None of the above

Solution:

The source and destination IP are the same in every fragment, since each fragment came from the same end host, and is being sent toward the same end host.

The identification field is the same for every fragment. Recall that the ID field is how we associate different fragments as being part of the same original packet.

The flags are different. For example, only the last fragment will have the "More Fragments" flag set to false. All other fragments will have the "More Fragments" flag set to true.

The fragment offsets are different, since this number corresponds to what part of the original packet is being sent in this fragment, and each fragment is sending a different part of the original packet.

Q2.4 (2 points) How would the router handle the 8020-byte packet?

- (A) User packet
- (B) Control traffic
- (C) Punt traffic

Solution:

The 8020-byte packet is a user packet that requires additional processing (fragmentation), so it would be considered punt traffic.

Note that control traffic is false because that refers to packets destined for the router itself (e.g. advertisements).

Q2.5 (3 points) IPv6 would drop this packet instead of fragmenting the packet. Select all true statements about this design choice.

- [A] This improves performance at the routers.
- [B] This eliminates the need for some fields in the IP header.
- [C] The sender cannot recover from packet loss, so the recipient will never receive the data.
- (D) None of the above

Solution:

(A): True. The router no longer has to support fragmentation, so there is less specialized work for the router to do.

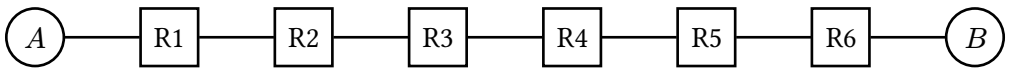
(B): True. The ID field, fragmentation flags, and offset field are no longer necessary.

(C): False. The sender can re-send the data in smaller packets, so that the recipient still receives the data.

Q3 *Links* **(14 points)**

Consider the topology below, where each link has bandwidth 10 Gbps, and propagation delay 2 sec.

For all subparts:
- A router must receive all bits of a payload before sending that payload along the next link.
- Queue sizes are infinite, and queues are processed in FIFO order.



Q3.1 (2 points) For this subpart, assume there is no other traffic on the network.

A sends a single 50 Gbit packet to B. What is the packet delay, in seconds?

Ⓐ 14 Ⓑ 21 Ⓒ 28 Ⓓ 35 Ⓔ 42 Ⓕ 47 Ⓖ 49 Ⓗ 56

Solution:

On each link, there is $50 \text{ Gbit} \ / \ 10 \text{ Gbps} = 5$ sec of bandwidth delay, and 2 sec of propagation delay, for a total delay of 7 sec.

Since each router must receive all bits of a payload before sending the payload along the next link, there is no overlapping of the delays. In other words, we must wait the full 7 seconds for a router to receive the packet, before the router can start sending that packet.

There are 7 links, and each link contributes 7 sec of delay, for a total of 49 sec of delay.

Q3.2 (3 points) For this subpart, assume that, at the moment A starts sending the 50 Gbit packet to B, there is a queue of 300 Gbits at router R5. How does the packet delay (in seconds) change in this scenario, compared to the delay from the previous subpart?

Ⓐ Decrease by 7
Ⓑ Decrease by 10
Ⓒ Decrease by 30
Ⓓ Increase by 7
Ⓔ Increase by 10
Ⓕ Increase by 30
Ⓖ No change
Ⓗ Not enough information

Solution:

From the previous subpart, we know that each link contributes 7 sec of delay, so the packet reaches R5 at time $t = 35$ sec.

The 300 Gbit queue takes $300 \text{ Gbit} \ / \ 10 \text{ Gbps} = 30$ sec to drain.

By the time the packet reaches R5, there is no queue, so there is no additional queuing delay incurred. Therefore, the packet delay is unchanged.

Q3.3 (3 points) Suppose we change the propagation delay of each link from 2 sec to x sec. The bandwidth is unchanged at 10 Gbps. A sends a single 50 Gbit packet to B.

Select the inequality that describes when the packet delay would be affected by a 300 Gbit queue at R5 (compared to the case where there is no queue at R5).

(A) $x < 1$ (C) $x < 2$ (E) $x < 3$ (G) $x > 4$ (I) $x < 5$

(B) $x > 1$ (D) $x > 2$ (F) $x > 3$ (H) $x > 4$ (J) $x > 5$

(K) The packet delay is never affected by the queue, for any value of x.

(L) The packet delay is always affected by the queue, for any value of x.

Solution:

On each link, there is 50 Gbit / 10 Gbps of bandwidth delay, and x sec of propagation delay, for a total delay of $5 + x$ sec.

There are 5 links between A and R5, so the packet arrives at R5 at $5(5 + x) = 25 + 5x$ sec.

The 300 Gbit queue takes 300 Gbit / 10 Gbps = 30 sec to drain.

For the packet delay to be affected by the queue, we need the packet to arrive while there is still a queue, i.e. we need the arrival time at R5 to be earlier than the draining time. In an inequality, this is:

$$25 + 5x < 30$$
$$x < 1$$

This means that if the propagation delay is shorter than 1 sec, then the packet will be affected by queuing delays.

Q3.4 (3 points) Suppose we change the bandwidth of each link from 10 Gbps to y Gbps. The propagation delay is unchanged at 2 sec. A sends a single 50 Gbit packet to B.

Select the inequality that describes when the packet delay would be affected by a 300 Gbit queue at R5 (compared to the case where there is no queue at R5).

(A) $y < 5$ (C) $y < 7$ (E) $y > 10$ (G) $y < 14$ (I) $y < 30$

(B) $y > 5$ (D) $y > 7$ (F) $y > 10$ (H) $y > 14$ (J) $y > 30$

(K) The packet delay is never affected by the queue, for any value of y.

(L) The packet delay is always affected by the queue, for any value of y.

Solution:

On each link, there is 50 Gbit / y Gbps of bandwidth delay, and 2 sec of propagation delay, for a total delay of $\frac{50}{y} + 2$ sec.

There are 5 links between A and R5, so the packet arrives at R5 at $5\left(\frac{50}{y} + 2\right) = \frac{250}{y} + 10$ sec.

The 300 Gbit queue takes 300 Gbit / y Gbps $= \frac{300}{x}$ sec to drain.

For the packet delay to be affected by the queue, we need the packet to arrive while there is still a queue, i.e. we need the arrival time at R5 to be earlier than the draining time. In an inequality, this is:

$$\begin{aligned} \frac{250}{y} + 10 &< \frac{300}{y} \\ 250 + 10y &< 300 \\ 10y &< 50 \\ y &< 5 \end{aligned}$$

This means that if the bandwidth is slower than 5 Gbps, then the packet will be affected by queuing delays.

Q3.5 (1 point) What is the bandwidth-delay product of each link?

(A) 20 Gbps (D) 5 Gbps (G) 0.2 Gbps

(B) 20 Gbit (E) 5 Gbit (H) 0.2 Gbit

(C) 20 sec (F) 5 sec (I) 0.2 sec

Solution:

The bandwidth-delay product is $10\frac{\text{Gbit}}{\text{sec}} \times 2 \text{ sec} = 20 \text{ Gbit}$.

Intuitively, the BDP represents the maximum amount of data in the link at any instant, so the units of measurement should measure data (Gbit), not a rate or a time.

Q3.6 (2 points) EvanBot suggests implementing reliablity in the routers. In which scenarios would this be a good idea?

Ⓐ Always, because the suggestion follows the end-to-end principle.

Ⓑ In scenarios where EvanBot cares about performance.

Ⓒ In scenarios where EvanBot cares about correctness, but not performance.

Ⓓ Never, because the suggestion violates the end-to-end principle.

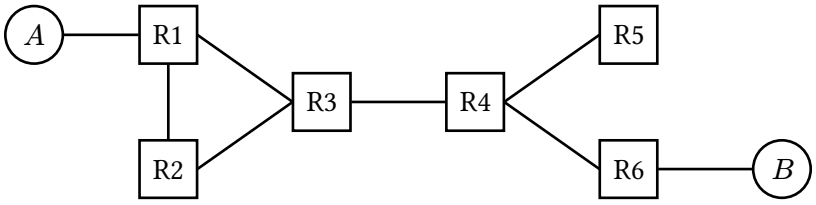
Solution:

The end-to-end principle says that reliability in routers is not necessary or sufficient for correctness (ruling out option C), but can be used as a performance enhancement (i.e. option B).

Note that the end-to-end principle is more of a guideline than a hard rule, so it's not something we need to strictly follow in all cases (ruling out options A and D).

Q4 *Distance-Vector* **(16 points)**

Consider running the distance-vector algorithm from lecture on this topology. All unlabeled links cost 1.



Assumptions:

- At $t = 0$, all forwarding tables are empty.
- At $t = 1$, static routes are installed.
- At every time step starting at $t = 2$, each router (1) expires routes, then (2) processes advertisements and updates its table, then (3) sends out advertisements.
- Propagation delay is negligible, i.e. an advertisement sent out at $t = 2$ can be processed at $t = 3$.
- Routing table entries expire after 5 seconds of receiving no advertisements.
- Split horizon, poison reverse, and route poisoning are disabled.
- All costs 16 or greater are considered infinite.

Q4.1 (2 points) What is the first time step where, after processing advertisements at that time step, every router has at least one entry in its forwarding table?

Ⓐ $t = 1$ Ⓑ $t = 2$ Ⓒ $t = 3$ Ⓓ $t = 4$ Ⓔ $t = 5$ Ⓕ $t = 6$

Solution:

$t = 0$: All forwarding tables are empty.

$t = 1$: R1 and R6 have static routes installed.

$t = 2$:

- R1 sends out an advertisement to R2 and R3.
- R6 sends out an advertisement to R4.

$t = 3$:

- R2, R3 now has a route to A (from R1's advertisement).
- R4 now has a route to B (from R6's advertisement.)
- R2, R3, R4 send out advertisements.

$t = 4$:

- R4 now has a route to A (from R3's advertisement).
- R3, R5 now have a route to B (from R4's advertisement).
- R3, R4, R5 send out advertisements.

Note: An earlier version of the solutions incorrectly marked $t = 3$ as the correct answer. This was a mistake; $t = 4$ is the only correct answer to this subpart.

Q4.2 (2 points) What is the first time step where, after processing advertisements at that time step, every router has two entries in its forwarding table?

Ⓐ $t = 1$ Ⓑ $t = 2$ Ⓒ $t = 3$ Ⓓ $t = 4$ Ⓔ $t = 5$ Ⓕ $t = 6$

Solution:

Continuing from the previous subpart:

$t = 5$:
- R5, R6 now have a route to A (from R4's advertisement).
- R1, R2 now have a route to B (from R4's advertisement).

Note: An earlier version of the solutions incorrectly marked $t = 4$ as the correct answer. This was a mistake; $t = 5$ is the only correct answer to this subpart.

Some time later, the network has converged. After convergence, host A goes down.

Q4.3 (3 points) For the next two subparts, every router except **R4** expires its route to A.

Which router(s) will eventually have a cost-16 entry for A? Select all that apply.

A R1 B R2 C R3 D R4 E R5 F R6 Ⓖ None

Solution:

R4 has the entry: "A is 3 away, via R3."

This gets advertised to R3, which accepts (since it has no route to A).

Now R3 has the entry: "A is 4 away, via R4."

Since split horizon is disabled, this gets advertised back to R4, which accepts (since it got an update from its own next-hop).

Now R4 has the entry: "A is 5 away, via R3."

Since split horizon is disabled, this gets advertised back to R3, which accepts (since it got an update from its own next-hop).

Now R3 has the entry: "A is 6 away, via R4."

This exchange between R3 and R4 continues, until eventually both R3 and R4 have a cost-16 route to A.

Also, note that as R3 installs increasing-cost entries, R3 will advertise them to R1 and R2, which will continue accepting them (since R1 and R2 are getting updates from their next-hop of R3).

Similarly, as R4 installs increasing-cost entries, R4 will advertise them to R5 and R6, which will continue accepting them (since R5 and R6 are getting updates from their next-hop of R4).

Q4.4 (2 points) Which modification(s) will fix the problem in the previous subpart, so that no routers have a cost-16 entry for A? Consider each modification independently.

[A] Enable split horizon.

[B] Enable poison reverse.

Ⓒ None of the above

Solution:

Split horizon will eliminate all cost-16 entries for A.

R4 has a cost-3 entry to A, via R3. Because split horizon is enabled, this entry is not advertised to the next-hop, i.e. R3 does not receive this advertisement. Therefore, R3 will not have an entry for A.

Since R3 will not have an entry for A, neither will R1 and R2. (Any advertisement from R4 to R1/R2 must go through R3.)

R4′s entry to A will get advertised to R5 and R6, even with split horizon enabled, because R5 and R6 are not the next-hop toward A.

R5 and R6 will accept this advertisement and each install a cost-4 entry to A, via R4.

Since split horizon is enabled, R5 and R6 will not advertise this entry back toward R4.

At this point, the network has reached steady-state again. R1/R2/R3 have no entry to A, and R4/R5/R6 have a finite-cost entry to A. Nobody has a cost-16 entry to A.

Poison reverse will not eliminate all cost-16 entries for A.

R4 has a cost-3 entry to A, via R3. Because poison reverse is enabled, a cost-16 poison advertisement is sent to the next-hop of R3.

Since R3 does not have an entry for A, R3 will accept the poison advertisement and install a cost-16 entry for A.

At this point, at least one router (R3) already has a cost-16 entry for A, so we are done.

Note: An earlier version of the solutions incorrectly marked “poison reverse” as a correct answer. This was a mistake; the answer shown above (split horizon only) is the only correct answer to this subpart.

Q4.5 (3 points) Again, consider the scenario where the network converges, and then host A goes down.

For this subpart only, every router except **R2** expires its route to A.

Which router(s) will eventually have a cost-16 entry for A? Select all that apply.

[A] R1 [B] R2 [C] R3 [D] R4 [E] R5 [F] R6 Ⓖ None

Solution:

R2 has the entry: "A is 2 away, via R1."

This gets advertised to R1, which accepts (since it has no route to A).

Now R1 has the entry: "A is 3 away, via R2."

Since split horizon is disabled, this gets advertised back to R2, which accepts (since it got an update from its own next-hop).

Similar to the previous subpart, this process repeats, and R1 and R2 get increasing-cost entries, until they eventually both have a cost-16 route to A.

Also, similar to the previous subpart, these increasing-cost routes to A get advertised to R3, which in turn advertises that route to R4, R5, R6.

Q4.6 (4 points) Which modification(s) will fix the problem in the previous subpart, so that no routers have a cost-16 entry for A? Consider each modification independently.

[A] Enable split horizon.

[B] Enable poison reverse.

[C] When A goes down, delete the R2-to-R3 link.

[D] When A goes down, delete the R1-to-R2 link.

[E] None of the above

Solution:

Split horizon is false.

R2 is the only router with an entry for A, which says that A is 2 away via R1.

R2 advertises this to R3 (which accepts, because it has no entry for A). R3 now has a cost-3 entry to A, via R2.

Then, R3 advertises to R1 (which accepts, because it has no entry for A). R1 now has a cost-4 entry to A, via R3.

Then, R1 advertises to R2 (which accepts, because this is an advertisement from the next-hop). R2 now has a cost-5 entry to A, via R1.

This advertisement will continue being sent around in a loop until R1, R2, R3 all have cost-16 entries to A.

Poison reverse is false.

With poison reverse enabled, R2 will send a poison cost-16 advertisement to its next-hop of R1.

Since R1 has no entry for A, the advertisement is accepted, and R1 now has a cost-16 entry for A.

Solution: (continued)

Deleting the R2-to-R3 link is false.

This will not fix the back-and-forth advertisements between R1 and R2.

Deleting the R1-to-R2 link is false.

When R1-to-R2 is deleted, R2 will propagate its route to A across the network. This will cause other routers to get finite-cost entries for A:

- R3 will have "A is 3 away via R2."
- R1 will have "A is 4 away via R3."
- R4 will have "A is 4 away via R3."
- and so on.

However, after 5 seconds, R2's entry ("A is 2 away via R1") will expire, because the periodic update from R1 will not arrive (since the R1-to-R2 link is down).

At this point, R3 will advertise back to R2, which will accept (because it has no entry for A).

Now:

- R2 has "A is 4 away via R3."
- R3 has "A is 3 away via R2."

Now, R2 advertises to R3, which must accept the update from its next-hop. So we have:

- R2 has "A is 4 away via R3."
- R3 has "A is 5 away via R2."

R2 and R3 will now advertise back-and-forth in a loop until their costs both reach 16 (i.e. infinity).

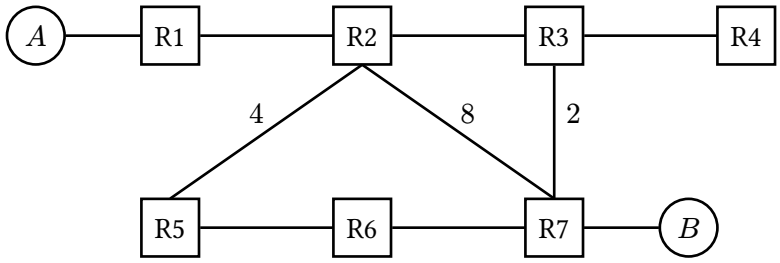
Note: An earlier version of the solutions incorrectly marked A, B, D as correct choices. This was a mistake; the answer shown above (none of the above) is the only correct answer to this subpart.

Q5 *Link-State* **(13 points)**

Consider running the link-state algorithm from lecture on the topology below.

Link costs correspond to packet travel times. All unlabeled links cost 1.

Assume there are no network errors, all routers use the same shortest-path algorithm, and advertisements are timestamped (as seen in lecture).



The routers exchange advertisements using this process:

- At $t = 0$, every router sends out hello messages to direct neighbors.
- As soon as a router knows about a neighbor, the router floods that information. For example, at $t = 2$, R3 sends an advertisement that says R3 and R7 are neighbors.
- At each time step, every router (1) processes advertisements, then (2) sends out any necessary advertisements.
- For this question, ignore periodically re-sending advertisements, i.e. assume all advertisements are sent just once.

Q5.1 (3 points) After processing advertisements at $t = 2$, which statements are true? Select all that apply.

[A] R1 knows that R1 and R2 are neighbors.

[B] R1 knows that R2 and R3 are neighbors.

[C] R1 knows that R2 and R5 are neighbors.

Ⓓ None of the above

Solution:

(A): True. R2 sends a hello message at $t = 0$, and the message is received by R1 at $t = 1$.

(B): True. R3 sends a hello message at $t = 0$, and the message is received by R2 at $t = 1$. Then, R2 sends an advertisement "R2 and R3 are neighbors" at $t = 1$, and the message is received by R1 at $t = 2$.

(C): False. R5 sends a hello message at $t = 0$, but it will not arrive at R2 until $t = 4$.

Q5.2 (3 points) After processing advertisements at $t = 5$, which statements are true? Select all that apply.

[A] R6 knows that R1 and R2 are neighbors.

[B] R6 knows that R2 and R3 are neighbors.

[C] R6 knows that R2 and R5 are neighbors.

Ⓓ None of the above

Solution:

(A): True. At $t = 1$, R2 knows. Then at $t = 2$, R3 knows. Then at $t = 4$, R7 knows. Then at $t = 5$, R6 knows.

(B): True. At $t = 1$, R3 knows. Then at $t = 3$, R7 knows. Then at $t = 4$, R6 knows.

(C): True. At $t = 4$, R5 knows. Then at $t = 5$, R6 knows.

Note: In Q5.3 to Q5.5, there were two different interpretations of "duplicate" that students considered:

Definition A: To define duplicates, we look at the message *without* its associated counter value. This means that these two messages *would* be considered duplicates:
- [ID 359] R3 and R4 are neighbors.
- [ID 360] R3 and R4 are neighbors.

The two different IDs could come from the message generated by R3 (receiving a hello from R4), and the message generated by R4 (receiving a hello from R3).

Definition B: To define duplicates, we look at both the message *and* its associated counter value (i.e. timestamp). Under this definition, the two messages above *would not* be considered duplicates.

Note that in both definitions, the implementation of flooding is unchanged: if you see a packet that is a "duplicate" (based on your specific definition) of a previous packet, you should not send out that packet again.

Also, note that after the initial hello phase, if a later router receives and floods a packet, the ID/ timestamp should not change. This is the behavior that avoids infinite flooding, and it exists in either definition.

Since the question was unclear about which definition we used, we granted points for both interpretations.

On future exams, we'll do our best to clarify if this affects the answer to the question.

Q5.3 (2 points) What is the first time step where R6 **receives** a duplicate advertisement about R3 and R4 being neighbors?

Ⓐ $t = 3$ Ⓑ $t = 4$ Ⓒ $t = 5$ Ⓓ $t = 6$ Ⓔ $t = 7$ Ⓕ $t = 8$

Ⓖ R6 never receives a duplicate advertisement.

Solution:

Using **Definition A** (ignore timestamp), the answer is $t = 7$:

The first advertisement arrives at $t = 4$, via R4–R3–R7–R6.

The second advertisement arrives at $t = 7$, via R4–R3–R2–R5–R6.

Since the two packets take totally different paths after R3 starts flooding the packet, no amount of duplicate checking (regardless of definition) will prevent the two duplicates from arriving at R6. In other words, R7, R2, R5 all see this packet only once, so the packet gets sent along both paths with nobody noticing a duplicate and dropping the packet.

One more alternate answer for **Definition A** (ignore timestamp) is $t = 6$:

The first advertisement still arrives at $t = 4$ as above.

The second advertisement arrives at $t = 6$, via R4–R3–R7–R6–R5–R6.

This uses the assumption that packets are flooded out of all links, including the incoming link. (On future exams, we'll do our best to clarify if this affects the answer to the question.)

Under **Definition B** (check timestamp), the answer is $t = 5$:

As before, the first advertisement (from the R4-to-R3 hello) arrives at $t = 4$, via R4–R3–R7–R6.

The second advertisement originates from R3's hello to R4. Then, R4 will flood this advertisement (with a separate timestamp) back to R3. Since the advertisement has a different timestamp, R3 will flood this distinct advertisement to R7, which will then flood to R6.

In total, the second advertisement travels along the path R3–R4–R3–R7–R6, so the arrival time at R6 is $t = 5$.

Note that the $t = 5$ path (R3–R4–R3–R7–R6) should not be considered in Definition A. If timestamps are ignored, R3 will not flood the advertisement a second time (notice that R3 is used twice on this path). In other words, Definition A requires considering only paths that don't loop back onto themselves.

Q5.4 (2 points) What is the first time step where R6 **sends** a duplicate advertisement about R3 and R4 being neighbors?

Ⓐ $t = 3$ Ⓑ $t = 4$ Ⓒ $t = 5$ Ⓓ $t = 6$ Ⓔ $t = 7$ Ⓕ $t = 8$

Ⓖ R6 never sends a duplicate advertisement.

Solution:

Using **Definition A** (ignore timestamp), the answer is "never."

If we don't consider the timestamp, then no matter how many times R6 receives the fact that R3 and R4 are neighbors, this fact is not flooded multiple times.

Note that we did not accept the $t = 6$ alternate answer from the previous subpart here, because that answer requires using Definition A, and if you used Definition A, then "never" is the beset answer.

Using **Definition B** (consider timestamp), the answer is $t = 5$:

As seen in the previous subpart, the two differently-timestamped advertisements (the first generated by R3 from the R4-to-R3 hello, and the second generated by R4 from the R3-to-R4 hello), arrive at $t = 4$ and $t = 5$ respectively.

Since we are considering timestamps, the second advertisement is treated as a non-duplicate message, and R6 floods it $t = 5$.

Note: An earlier version of the solutions incorrectly marked $t = 7$ as the answer, based on the R4–R3–R2–R5–R6 from the previous subpart, but we think this is not correct under either definition.

Under Definition A, R6 would consider any two "R3 and R4 are neighbors" messages to be duplicates, regardless of timestamp, and would not send a duplicate advertisement.

Under Definition B, the two advertisements arriving at $t = 4$ and $t = 7$ have the same timestamp, since they both originated from the R4-to-R3 hello. Therefore, they would be considered duplicates, and nothing would get flooded at $t = 7$. In any case, the flooding at $t = 5$ occurs earlier under this definition.

Q5.5 (2 points) In total, how many times does R1 receive an advertisement about R6 and R7 being neighbors?

Ⓐ 0 Ⓑ 1 Ⓒ 2 Ⓓ 3 Ⓔ 4 Ⓕ 5

Ⓖ More than 5

Solution:

Using **Definition A** (ignore timestamp), the answer is 1.

Every path from either R6/R7 to R1 must pass through R2 first. If we ignore timestamps, no matter how many times R2 sees the fact "R6 and R7 are neighbors," R2 will only flood this fact to R1 a single time.

Under **Definition B** (consider timestamp), the answer is 2.

Two advertisements that say "R6 and R7 are neighbors" get generated:
- One when R6 hears the hello from R7, and
- the other when R7 hears the hello from R6.

Each of these advertisements gets flooded over the network.

Note that under this definition, the two advertisements have different timestamps, so a router that receives both advertisements will flood both of them.

Note that 3 is incorrect, regardless of which definition you used.

The (incorrect) reasoning for 3 would have something to do with finding three paths from R6 to R1 (a similar incorrect argument finds three paths from R7 to R1):
- R6–R5–R2–R1
- R6–R7–R2–R1
- R6–R7–R3–R2–R1

However, under Definition A (ignore timestamp), R2 would not flood this advertisement three times to R1.

Also, under Definition B (consider timestamp), these advertisements all originate from R6 (via an R7-to-R6 hello), so they would have the same timestamp. Therefore, R2 would still not flood this advertisement three times to R1.

Q5.6 (1 point) At convergence, what is the next-hop toward A in R7's forwarding table?

Ⓐ R2 Ⓑ R3 Ⓒ R6 Ⓓ None

Solution:

The shortest path from R7 to A is: R7–R3–R2–R1.

Assuming no network errors and all routers using the same algorithm, the network should converge to valid shortest paths.

Therefore, the next-hop from R7 toward A is R3.

Q6 *Longest Prefix Matching* **(14 points)**

IP addresses are written in binary (base-2), where every bit is either 0 or 1. In this question, we will consider writing IP addresses in *ternary* (base-3) instead, where every *trit* is either 0, 1, or 2. (A trit is the ternary equivalent of a bit.)

Q6.1 (1 point) If IPv4 addresses are 32 trits long, how many total IP addresses are available?

Ⓐ 2^{32} Ⓑ 2^{35} Ⓒ 3^{32} Ⓓ 3^{35} Ⓔ 32^3 Ⓕ 35^3

Solution: Each trit can be one of 3 possible values, and there are 32 trits, for a total of $\underbrace{3 \times 3 \times \ldots \times 3}_{32 \text{ times}} = 3^{32}$ addresses.

Recall that classful addressing creates three types of IPv4 addresses:

Class A: 24 host bits/trits Class B: 16 host bits/trits Class C: 8 host bits/trits

In Q6.2 to Q6.4, create three classes of ternary 32-trit IPv4 addresses, where each class has exactly the number of host trits shown above, and as many network trits as possible.

Hint: Remember that some trit(s) must be used to determine which class the address belongs to.

Q6.2 (1 point) How many network trits are in a ternary Class A address?

Ⓐ 5 Ⓑ 6 Ⓒ 7 Ⓓ 8 Ⓔ 9 Ⓕ 10

Solution: In every address, we need exactly one trit to determine the class. For example, the top trit can be assigned 0 for Class A, 1 for Class B, and 2 for Class C.

In a Class A address, there are 32 trits total, with 24 host trits and 1 class trit. This leaves $32 - 24 - 1 = 7$ network trits.

Q6.3 (1 point) How many network trits are in a ternary Class B address?

Ⓐ 11 Ⓑ 12 Ⓒ 13 Ⓓ 14 Ⓔ 15 Ⓕ 16

Solution: In a Class B address, there are 32 trits total, with 16 host trits and 1 class trit. This leaves $32 - 16 - 1 = 15$ network trits.

Note that this is different from the binary version, where 2 class bits are needed, leaving only 14 network bits.

Q6.4 (1 point) How many network trits are in a ternary Class C address?

Ⓐ 20 Ⓑ 21 Ⓒ 22 Ⓓ 23 Ⓔ 24 Ⓕ 25

Solution: In a Class C address, there are 32 trits total, with 8 host trits and 1 class trit. This leaves $32 - 8 - 1 = 23$ network trits.

As in the previous subpart, note that this is different from the binary version, where 3 class bits are needed, leaving only 21 bits.

For Q6.5 and Q6.6, consider classless addressing, with ternary 32-trit IPv4 addresses.

Hint: The powers of 3 are:

3^0	3^1	3^2	3^3	3^4	3^5	3^6	3^7	3^8	3^9	3^{10}
1	3	9	27	81	243	729	2187	6561	19683	59049

Q6.5 (2 points) EvanBot is given the 10.0.735.0/23 range. How many separate /27 ranges can be allocated out of this range?

Ⓐ 1 Ⓑ 3 Ⓒ 9 Ⓓ 27 Ⓔ 81 Ⓕ 729

Solution: The given IP address, written out in base-3, with fixed bits colored red, is:

`00000011.00000000.0100002`0.00000000

The bottom 9 trits are unfixed.

In a /27 range, the bottom $32 - 27 = 5$ bits should be unfixed.

Out of the 9 unfixed trits, this leaves $9 - 5 = 4$ trits that can be used to identify a specific /27 range. This gives us $3^4 = 81$ possible /27 ranges.

Q6.6 (2 points) Which ranges contain the address 10.0.0.22? Select all that apply.

☐ A 10.0.0.9/30 ☐ C 10.0.0.12/31 Ⓔ None of the above

☒ B 10.0.0.18/30 ☒ D 10.0.0.21/31

Solution: Fixed bits are colored red. For an option to be correct, all fixed (red) bits must match the given IP address.

The given IP address: 00000011.00000000.00000000.00000211

Option (A): 00000011.00000000.00000000.000001**00**

Option (B): 00000011.00000000.00000000.000002**00**

Option (C): 00000011.00000000.00000000.0000011**0**

Option (D): 00000011.00000000.00000000.0000021**0**

Q6.7 (3 points) In this subpart, we want to build a trie for longest prefix matching on ternary 32-trit IPv4 addresses.

The 9-entry forwarding table that we want to convert into a trie is shown below. Every prefix is a /2, and the two fixed bits of each prefix are provided for your convenience.

Fixed Bits	Port
00	4
01	5
02	5

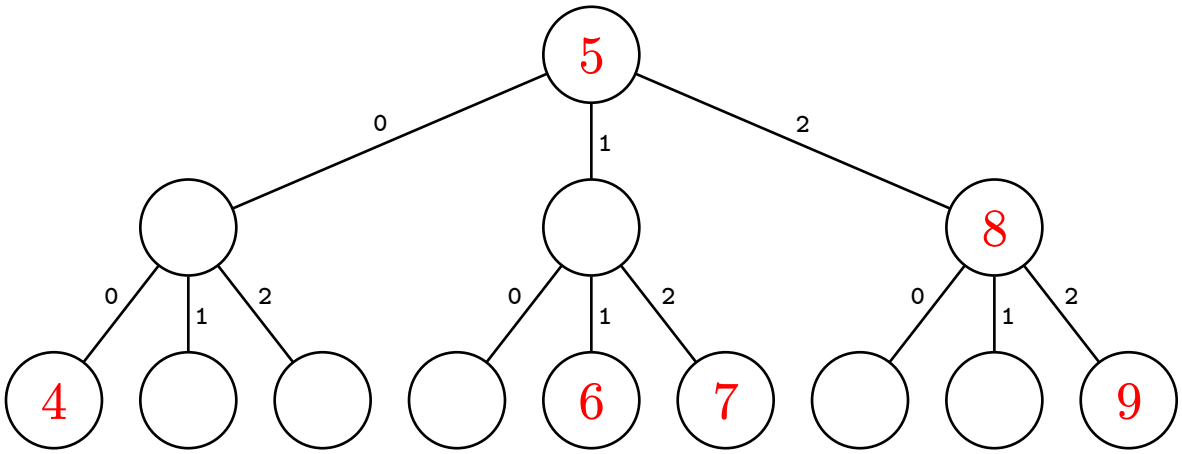
Fixed Bits	Port
10	5
11	6
12	7

Fixed Bits	Port
20	8
21	8
22	9

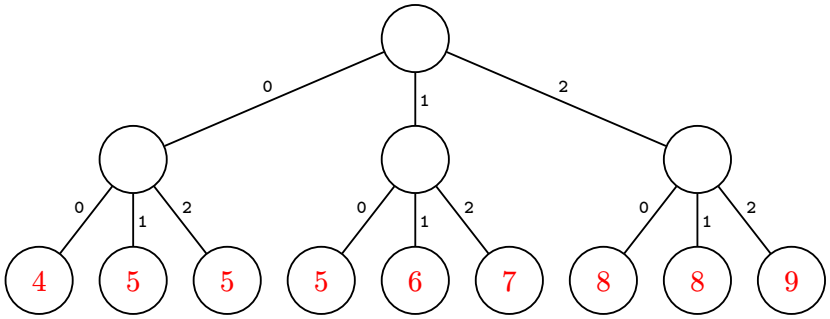
Fill in the trie below by writing port numbers (4, 5, ..., 9) in the circles.

Not all nodes will be used; leave unused nodes blank, and leave nodes with no label blank.

For full credit, your trie should use the minimum possible number of nodes.

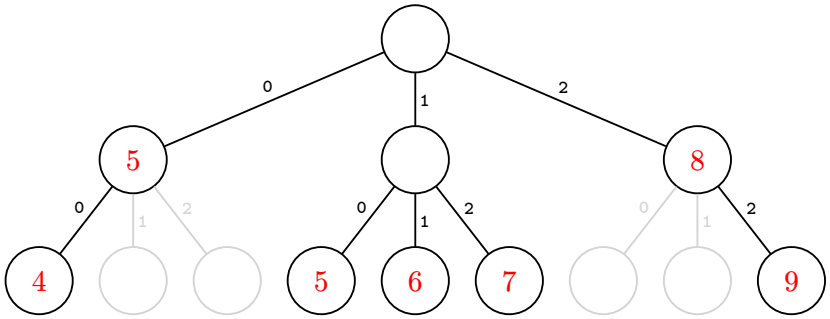


Solution: The generic trie, with no optimizations:

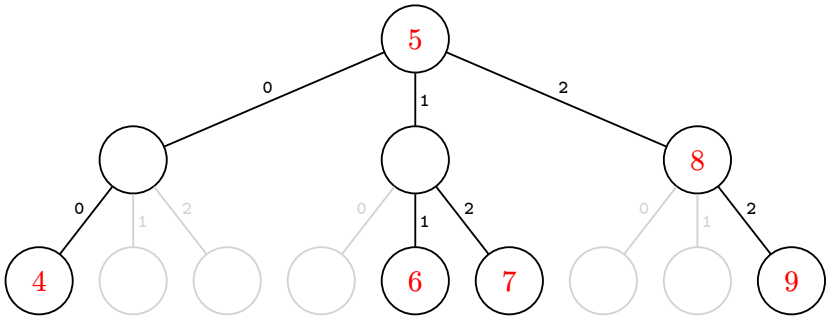


We can push nodes `01` and `02` up to the parent, and the resulting left sub-tree still produces the same forwarding decisions: Port 4 if `00`, else Port 5.

Likewise, we can push nodes `20` and `21` up to the parent, and the resulting right sub-tree still produces the same forwarding decisions: Port 9 if `22`, else Port 8.



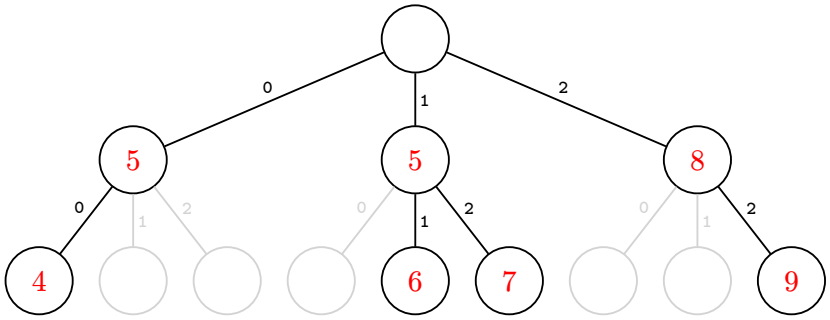
Finally, we can push nodes `0` and `10` up to the root, and the resulting tree still produces the same forwarding decisions: `0` and `10` still map to Port 5, and the nodes mapping to Ports 4, 6, 7 are unchanged.



Solution:

Alternate solution:

Same as the solution above, but instead of Port 5 in the root, Port 5 is written in nodes `0` and `1`. All forwarding decisions remain the same, and the number of nodes used is the same (i.e. the minimal number of nodes).



In this solution, you can also put any value you want in the root node, because the value at the root doesn't affect any of the forwarding decisions in this particular trie. Filling in the root doesn't add any extra nodes to the trie, either, so the number of nodes remains minimal.

In Q6.8 and Q6.9, consider a **binary** trie built for longest prefix matching on binary 32-bit IPv4 addresses:
- A default route is added in the root of the trie.
- Every non-default prefix in the trie is exactly 6 bits long (i.e. they represent /6 ranges).
- There is at least one non-default prefix, but we do not know the exact number of prefixes in the trie.

Without changing anything about the trie, we now interpret the trie as a **ternary** trie for longest prefix matching on ternary 32-trit IPv4 addresses.

Q6.8 (1 point) When interpreted as a ternary trie, what is the **minimum** number of ternary addresses that get forwarded on a non-default route?

Ⓐ 3^3 Ⓑ 3^6 Ⓒ 2^{26} Ⓓ 3^{26} Ⓔ 2^{32} Ⓕ 3^{32}

Solution: To get the minimum number of addresses, we install just one 6-bit prefix, representing one /6 range. This range fits 3^{26} addresses.

Q6.9 (2 points) When interpreted as a ternary trie, what is the **maximum** number of ternary addresses that get forwarded on a non-default route?

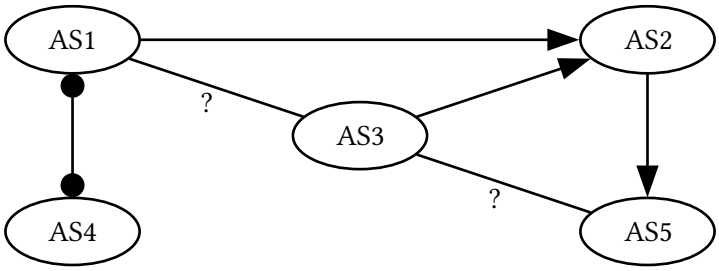
Ⓐ 2^{26} Ⓒ 64×2^{26} Ⓔ 729×2^{26} Ⓖ 2^{32}

Ⓑ 3^{26} Ⓓ 64×3^{26} Ⓕ 729×3^{26} Ⓗ 3^{32}

Solution: To get the maximum number of addresses, we install as many 6-bit prefixes as possible. Since the trie is initially built as a binary trie, there are $2^6 = 64$ possible 6-bit prefixes. Each prefix represents a /6 range, so the prefixes together cover 64×3^{26} addresses in total.

Q7 *BGP* **(22 points)**

Consider the AS graph below, where all ASes follow the Gao-Rexford rules.



The AS1–AS3 and AS3–AS5 relationships are unknown. In other words, the two edges marked ? could each be ●—● or —▶ or ◀— .

Q7.1 (2 points) For this subpart only, suppose the AS3–AS5 edge does not exist.

How many ways are there to label AS1–AS3 edge, such that the resulting AS graph is a valid hierarchy that guarantees reachability and convergence?

Ⓐ 0 Ⓑ 1 Ⓒ 2 Ⓓ 3

Solution:

Recall that an AS graph must meet two conditions to guarantee reachability and convergence:
1. Hierarchical: Traversing up the graph (from customers to providers) leads to a Tier 1 AS.
2. Acyclic: The AS graph has no cycles.

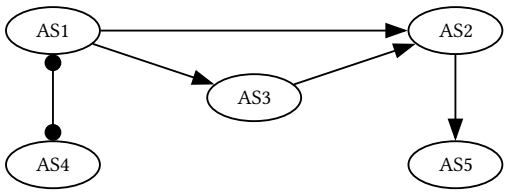
Also, recall that a Tier 1 AS has no providers, and is peered with every other Tier 1 AS.

The potential Tier 1 ASes are AS1, AS3, AS4.

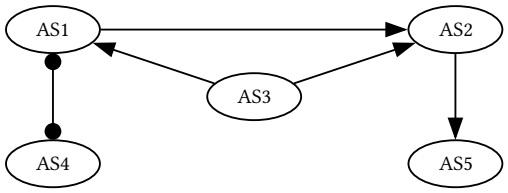
Note that AS2 and AS5 could not be Tier 1 ASes, since they have at least one provider already (even before labeling the unknown edges).

Now, we can go through the 3 possible labels on the AS1–AS3 edge. For each possibility, we check that there are no cycles, and that every AS can reach a Tier 1 AS by traversing up customer-provider links.

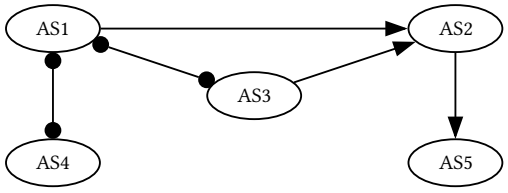
1. AS1 → AS3: Valid.
   AS1 and AS4 are the two Tier 1 AS.



2. AS1 ← AS3: Invalid.
   There is no valid path from AS4 to AS3. In the AS4 — AS1 ← AS3 path, AS1 has a peer on one side and a provider on the other, so AS1 will not participate.



3. AS1 — AS3: Invalid.
   There is no valid path from AS4 to AS3. In the AS4 — AS1 — AS3 path, AS1 has peers on both sides and thus will not participate.



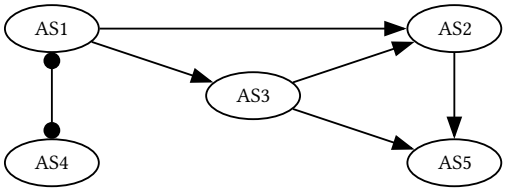
Q7.2 (3 points) How many ways are there to label the two unknown edges, such that the resulting AS graph is a valid hierarchy that guarantees reachability and convergence?

Ⓐ 0 Ⓑ 1 Ⓒ 2 Ⓓ 3 Ⓔ 4 Ⓕ 5 Ⓖ 6 Ⓗ 7 Ⓘ 8 Ⓙ 9

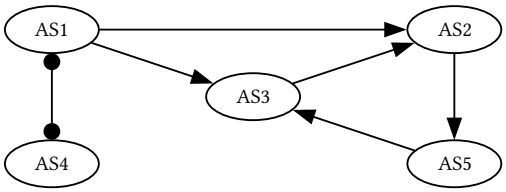
Solution:

Similar reasoning from the previous subpart, but with more permutations to check. (Sorry, this is indeed a little tedious.)

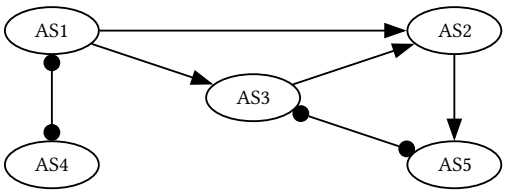
1. AS1 → AS3 and AS3 → AS5: Valid.
   AS1 and AS4 are the Tier 1 ASes.



2. AS1 → AS3 and AS3 ← AS5: Invalid.
   A cycle AS3 → AS2 → AS5 has formed.

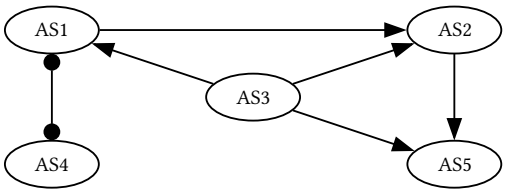


3. AS1 → AS3 and AS3 — AS5: Valid.
   AS1 and AS4 are the Tier 1 ASes.

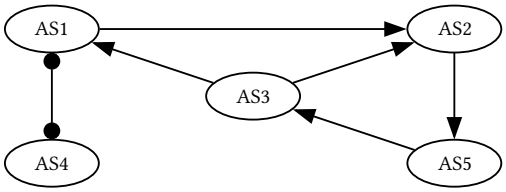


Solution: (continued)

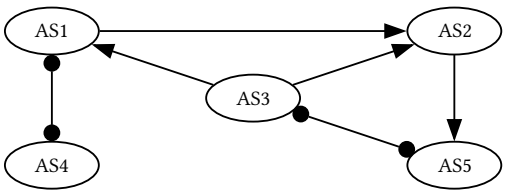
4. AS1 ← AS3 and AS3 → AS5: Invalid.
   There is no valid path from AS4 to AS3.



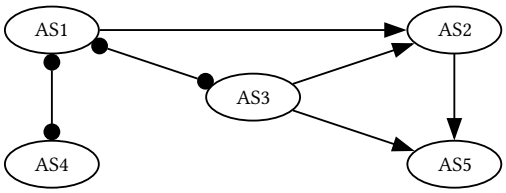
5. AS1 ← AS3 and AS3 ← AS5: Invalid.
   A cycle AS3 → AS2 → AS5 has formed.



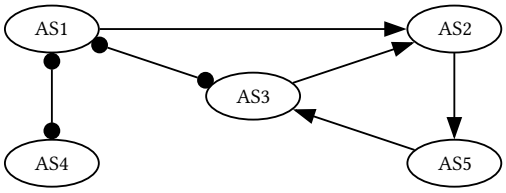
6. AS1 ← AS3 and AS3 — AS5: Invalid.
   There is no valid path from AS4 to AS3.



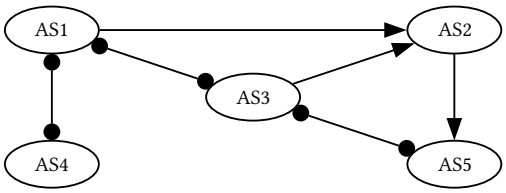
7. AS1 — AS3 and AS3 → AS5: Invalid.
   There is no valid path from AS4 to AS3.



8. AS1 — AS3 and AS3 ← AS5: Invalid.
   A cycle AS3 → AS2 → AS5 has formed.



9. AS1 — AS3 and AS3 — AS4: Invalid.
   There is no valid path from AS4 to AS3.



Q7.3 (3 points) Which of these could be the set of Tier 1 ASes, for some labeling of the unknown edges? Select all that apply.

[A] AS1 only
[C] AS4 only
[E] AS1 and AS4
Ⓖ None
[B] AS3 only
[D] AS1 and AS3
[F] AS1, AS3, and AS4

Solution: See solution to the previous subpart.

Note: Some students tried to select other answers like "AS1 and AS3" based on some of the invalid permutations of edge labelings. For example, in the 9th permutation (both peering links), one could argue that AS1 and AS3 have no providers and are Tier 1 ASes. Below is our response on Ed for why this reasoning was not granted points:

Tier 1 isn't just "no provider" - it's "no provider *and* peered with every other Tier 1." Those two conditions together are what make the hierarchy work. Under a labeling where AS1–AS3 and AS3–AS5 are both peer links, AS1, AS3, and AS4 all lack providers, but they're not all peered with each other (no AS3–AS4 edge), so they don't form a valid Tier 1 set. There's no labeling of the unknown edges that produces {AS1, AS3} or {AS1, AS3, AS4} as the Tier 1 set under the full definition.

The question could have had clearer wording, but the Tier 1 definition itself rules out these options regardless of whether we're filtering for valid labelings.

For Q7.4 to Q7.6, select whether it is possible for packets to be sent from the source AS to the destination AS assuming a valid AS hierarchy. In other words, is there an AS path from source to destination where all intermediate ASes agree to export the path?

- "Always" means that a path exists, for any valid labeling of the unknown edges.
- "Sometimes" means that a path exists in some valid graphs and does not in others, depending on how the unknown edges are labeled.
- "Never" means that a path does not exist in any valid graph.

Q7.4 (2 points) Source: AS3, Destination: AS5

Ⓐ Always
Ⓑ Sometimes
Ⓒ Never

Solution:

Regardless of the unknown edges, we can use the AS3 → AS2 → AS5 path.

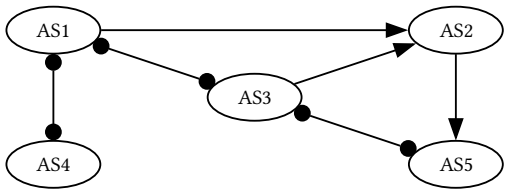
Q7.5 (2 points) Source: AS4, Destination: AS3

Ⓐ Always Ⓑ Sometimes Ⓒ Never

Solution:

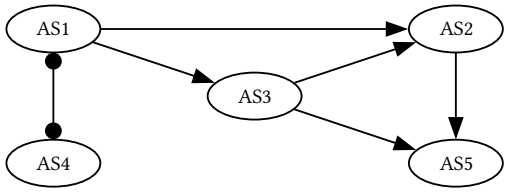
Our intended answer was **Sometimes**, for the following reason:

If AS1 — AS3 and AS3 — AS5 are both peering links:



Then, no path exists:
- AS4 — AS1 — AS3 fails because AS1 has peers on either side.
- AS4 — AS1 → AS2 ← AS3 fails because AS2 has providers on either side.
- AS4 — AS1 → AS2 → AS5 — AS3 fails because AS5 has a provider and a peer as neighbors.

However, if we have one of the valid arrangements like AS1 → AS3 and AS3 → AS5:



Specifically, the AS4 — AS1 → AS3 path works.

However, after the exam, some students offered an alternate interpretation of the question wording, where the answer would be **Always**, for the following reason:

If "valid" was interpreted to mean "valid hierarchy that also guarantees reachability and convergence," then only two out of the nine permutation from earlier need to be considered. In both of those permutations, the AS4 — AS1 → AS3 path works.

Our intent was for "valid" to specifically refer to a valid hierarchy (i.e. no loops), but since the wording was ambiguous, we granted points for both "Sometimes" and "Always" on this subpart.

Q7.6 (2 points) Source: AS4, Destination: AS5

Ⓐ Always Ⓑ Sometimes Ⓒ Never

Solution:

Regardless of the unknown edges, we can use the AS4 — AS1 → AS2 → AS5 path.

In each of Q7.7 to Q7.10, two advertisements are shown. Select the advertisement that will be imported, according to the tiebreaking priorities from lecture. If a route attribute is not shown, it could be any value.

Reminder: There could be multiple links between the same two ASes, even if the AS graph only shows one edge.

Q7.7 (2 points) If a router in AS4 hears about the two advertisements below, which one is imported?

Advertisement I: A router in AS1 sends: "Destination AS2 with ASPATH [AS1, AS2]."

Advertisement II: A different router in AS1 sends: "Destination AS2 with ASPATH [AS1, AS3, AS2]."

Ⓐ Always Advertisement I. Ⓑ Always Advertisement II.

Ⓒ Not enough information: Depends on the unknown edge labels.

Ⓓ Not enough information: Depends on something else besides the unknown edge labels.

Solution:

The first priority is Gao-Rexford rules, which are tied because both advertisements are coming from a customer.

The second priority is shorter AS path. Here, Advertisement I has the shorter AS path, so it is selected.

Note that the edge labels do not affect the selection of advertisement in this case.

Q7.8 (2 points) If a router in AS2 hears about the two advertisements below, which one is imported?

Advertisement I: A router in AS5 sends: "Destination AS5 with ASPATH [AS5], and MED 1."

Advertisement II: A different router in AS5 sends: "Destination AS5 with ASPATH [AS5], and MED 2."

Ⓐ Always Advertisement I.

Ⓑ Always Advertisement II.

Ⓒ Not enough information: Depends on the unknown edge labels.

Ⓓ Not enough information: Depends on something else besides the unknown edge labels.

Solution:

The first priority is Gao-Rexford rules, which are tied because both advertisements are coming from a customer.

The second priority is shorter AS path, which is also tied because both AS paths are [AS5].

The third priority is closest egress router. However, we don't know the egress routers for the two routers in AS5, so there is not enough information to determine which advertisement gets imported.

Note that the unknown edge labels do not affect which advertisement gets accepted. Even if we knew the exact edge labels, we still wouldn't know which of the two advertisements has the closer egress router.

Q7.9 (2 points) If a router in AS3 hears about the two advertisements below, which one is imported?

Advertisement I: A router in AS2 sends: "Destination AS5 with ASPATH [AS2, AS5]."

Advertisement II: A router in AS1 sends: "Destination AS5 with ASPATH [AS1, AS2, AS5]."

Ⓐ Always Advertisement I.

Ⓑ Always Advertisement II.

Ⓒ Not enough information: Depends on the unknown edge labels.

Ⓓ Not enough information: Depends on something else besides the unknown edge labels.

Solution:

The first priority is Gao-Rexford rules.

AS2 is a customer, but we don't know whether AS1 is a customer, peer, or provider.

If AS1 is a peer or provider, then Advertisement I from AS2 will get accepted.

If AS2 is a customer, then we have a tie and move on to the second priority of shorter AS path, which causes Advertisement I to be accepted.

Therefore, Advertisement I is accepted in all cases.

Q7.10 (2 points) If a router in AS5 hears about the two advertisements below, which one is imported?

Advertisement I: A router in AS3 sends: "Destination AS1 with ASPATH [AS3, AS1]."

Advertisement II: A router in AS2 sends: "Destination AS1 with ASPATH [AS2, AS1]."

Ⓐ Always Advertisement I.

Ⓑ Always Advertisement II.

● Ⓒ Not enough information: Depends on the unknown edge labels.

● Ⓓ Not enough information: Depends on something else besides the unknown edge labels.

Solution:

Our intended answer was **"Depends on unknown edge labels,"** for the following reason:

The first priority is Gao-Rexford rules.

AS2 is a provider, but we don't know whether AS3 is a customer, peer, or provider.

If AS3 is a customer or peer, then Advertisement II from AS3 will get accepted.

If AS3 is a provider, then we'd move on to some later tiebreaking rules.

At this point, we can already see that there is not enough information, and the decision depends on the unknown edge labels.

However, after the exam, some students offered an alternate interpretation of the question wording, where the answer would be **"Depends on something else"**, for the following reason:

As mentioned above, if AS3 is a provider, additional tiebreaking rules would be required – therefore, the answer depends on something else besides the unknown edge labels.

Since the wording of the question was not clear enough to point toward one interpretation, we gave students benefit of the doubt and granted points for both "Depends on unknown edge labels" and "Depends on something else."

Comment Box

Congrats for making it to the end of the exam! Leave any thoughts, comments, feedback, or doodles here.

Nothing in the comment box will affect your grade.

Ambiguities

If you feel like there was an ambiguity on the exam, you can put it in the box below.

For ambiguities, you must qualify your answer and provide an answer for both interpretations. For example, "if the question is asking about A, then my answer is X, but if the question is asking about B, then my answer is Y." You will only receive credit if it is a genuine ambiguity and both of your answers are correct. We will only look at this box if you request a regrade.