Problem 1)

a) 4b learns about prefix x via RIP since 4a is directly connected to x
b) 3c learns about prefix x via eBGP
c) 1c learns about x via eBGP
d) 1d learns about x via iBGP

Problem 2)

Hot Potato Routing means that we hand off traffic to the next hop network ASAP and as a result could be asymmetric.

C1 to C2: J => H => I => Exchange Point => F => D => C => B
= 5 + 5 + 5 + 10 + 35 + 20 + 5 = 85 ms
C2 to C1: B => C => A => G => H => J
= 5 + 10 + 5 + 10 + 5 = 35 ms

Problem 3)

a) I'll keep track of the forwarding tables while walking through every event.

NOTE: Interface 3 marks the physical port that can reach to the host
NOTE: [Switch 1, port 1] is identified as Switch 1.1 and [Router 1, port 2] is identified as Router 1.2 for clarity reasons

Time = 1: Host A sends an IP datagram to Host G
[A] => [Hub 1] => [Switch 1, port 1]
[Switch 1] is reached, thus [Switch 1] broadcasts to [ports 2 and 3]
[Switch 1, port 3] => [Router 1, port 1] => [Router 1, port 2] => [Hub 3] => [G]

[Switch 1] broadcasts ONCE here

| **Switch 1** | |
|---|---|
| **Host** | **Interface** |
| A | 1 |

| **Switch 3** | |
|---|---|
| **Host** | **Interface** |

Time = 2: Host G sends an IP datagram to Host A
[G] => [Hub 3] => [Router 1, port 2] => [Router 1, port 1] => [Switch 1, port 3]
Since [A] in forwarding table, [Switch 1] only forwards to [port 1]

[Switch 1, port 1] => [Hub 1] => [A]

| Switch 1 | |
|---|---|
| **Host** | **Interface** |
| A | 1 |
| Router 1.1 | 3 |

| Switch 3 | |
|---|---|
| **Host** | **Interface** |

Time = 3: Host D sends an IP datagram to Host L
[D] => [Hub 2] => [Switch 1, port 2]
You reach a switch, but you don't broadcast since [Router 1] is already in the forwarding table, so you can just immediately go to [Switch 1, port 3] (or switch 1.3 for naming conventions). [Switch 1] now knows [D] is on [port 2] so add to the forwarding table.
[Switch 1, port 3] => [Router 1, port 1] => [Router 1, port 3] => [Switch 3, port 1]
[Switch 3] has been reached, has to broadcast since nothing is in its table yet
[Switch 3, port 2] => [Hub 4] => [L]

[Switch 3] broadcasts ONCE here

| Switch 1 | |
|---|---|
| **Host** | **Interface** |
| A | 1 |
| Router 1.1 | 3 |
| D | 2 |

| Switch 3 | |
|---|---|
| **Host** | **Interface** |
| Router 1.3 | 1 |

Time = 4: Host D sends an IP datagram to Host I

[D] => [Hub 2] => [Switch 1, port 2]
You don't have to broadcast since [Switch 1.3] to [Router 1.1] is alr in [Switch 1] forwarding table. Also [D] is already in [Switch 1]'s forwarding table.
[Switch 1, port 3] => [Router 1, port 1] => [Router 1, port 2] => [Hub 3] => [I]

| **Switch 1** | |
|---|---|
| **Host** | **Interface** |
| A | 1 |
| Router 1.1 | 3 |
| D | 2 |

| **Switch 3** | |
|---|---|
| **Host** | **Interface** |
| Router 1.3 | 1 |

<u>Time = 5: Host F sends an IP datagram to Host A</u>
[F] => [Hub 2] => [Switch 1, port 2] => [Switch 1, port 1] => [Hub 1] => [A]
We can take advantage of the forwarding table here to avoid broadcasting again.
Also [Switch 1] now knows that [F] is under [port 2], so add to the forwarding table.

| **Switch 1** | |
|---|---|
| **Host** | **Interface** |
| A | 1 |
| Router 1.1 | 3 |
| D | 2 |
| F | 2 |

| **Switch 3** | |
|---|---|
| **Host** | **Interface** |
| Router 1.3 | 1 |

Time = 6: Host K sends an IP datagram to Host G
[K] => [Hub 4] => [Switch 3, port 2]
[Switch 3] has [Router 1.3] saved in its forwarding table so no need to broadcast again.
[Switch 3, port 1] => [Router 1, port 3] => [Router 1, port 2] => [Hub 3] => [G]
Also [Switch 3] now knows that K is under [port 2] so add it to the forwarding table.

| **Switch 1** | |
|---|---|
| **Host** | **Interface** |
| A | 1 |
| Router 1.1 | 3 |
| D | 2 |
| F | 2 |

| **Switch 3** | |
|---|---|
| **Host** | **Interface** |
| Router 1.3 | 1 |
| K | 2 |

Time = 7: Host J sends an IP datagram to Host F
[J] => [Hub 4] => [Switch 3, port 2]
[Switch 3] has [Router 1.3] saved in its forwarding table so no need to broadcast again.
[Switch 3, port 1] => [Router 1, port 3] => [Router 1, port 1] => [Switch 1, port 3] =>
[Switch 1, port 2] => [Hub 2] => [F]
[Switch 3] now knows that [J] is under [port 2] so put it in the forwarding table.

| **Switch 1** | |
|---|---|
| **Host** | **Interface** |
| A | 1 |
| Router 1.1 | 3 |
| D | 2 |

| | |
|---|---|
| F | 2 |

| Switch 3 | |
|---|---|
| **Host** | **Interface** |
| Router 1.3 | 1 |
| K | 2 |
| J | 2 |

[Switch 1] broadcasts a total of 1 time
[Switch 3] broadcasts a total of 1 time

b)

| Switch 1 | |
|---|---|
| **Host** | **Interface** |
| A | 1 |
| Router 1.1 | 3 |
| D | 2 |
| F | 2 |

| Switch 3 | |
|---|---|
| **Host** | **Interface** |
| Router 1.3 | 1 |
| K | 2 |
| J | 2 |

c) t = 10, Host A sends Broadcast IP datagram in the subnet
[A] => [Hub 1]
[Hub 1] broadcasts to [B] and [C]
[Hub 1] => [Switch 1, port 1]

[Switch 1] broadcasts to [port 2, 3]
[Switch 1, port 2] → [Hub 2] → [D], [E], and [F]
[Switch 1, port 3] → [Router 1]

B, C, D, E, and F will receive the broadcast IP datagram

Problem 4)

a)

| Frame | Destination MAC address | Source MAC address | Devices that receive the frame | New entries added |
|---|---|---|---|---|
| 1 | FF:FF:FF:FF:FF:FF | 00:00:00:00:00:03 | Node A, Router | B => Port 3 |
| 2 | 00:00:00:00:00:03 | 00:00:00:00:00:02 | Node B | A => Port 2 |
| 3 | 00:00:00:00:00:02 | 00:00:00:00:00:03 | Node A | |
| 4 | 00:00:00:00:00:03 | 00:00:00:00:00:02 | Node B | |

1: Node B attempts to broadcast request ARP to Node A since it is within same subnet
2: ARP reply–Node A replies to Node B with Node A's IP so Node B knows that Node A's IP is attached to its MAC address
3: Node B sends datagram to Node A
4: Node A rends reply back to Node B

b)

| Frame | Destination MAC address | Source MAC address | Devices that receive the frame | New entries added |
|---|---|---|---|---|
| 1 | FF:FF:FF:FF:FF:FF | 00:00:00:00:00:03 | Node A, Router | |
| 2 | 00:00:00:00:00:03 | 00:00:00:00:00:01 | Node B | Router => Port 1 |
| 3 | 00:00:00:00:00:01 | 00:00:00:00:00:03 | Router | |
| 4 | FF:FF:FF:FF:FF:FF | 00:00:00:00:00:04 | Node C | |
| 5 | 00:00:00:00:00:04 | 00:00:00:00:00:05 | Router | |
| 6 | 00:00:00:00:00:05 | 00:00:00:00:00:04 | Node C | |
| 7 | 00:00:00:00:00:04 | 00:00:00:00:00:05 | Router | |
| 8 | 00:00:00:00:00:03 | 00:00:00:00:00:01 | Node B | |

1: Node B attempts to broadcast ARP since it doesn't know Router MAC address
2: ARP reply–Router replies with own IP address so receiver can confirm (reply is UNICAST)
3: Node B send actual datagram to router since destination isn't in the same subnet as Node B
4: ARP request from Router to Node C
5: ARP reply–Node C responds with its own IP address

6: Router forwards datagram to Node C
7: C is then able to send reply to Router
8: Node B receives the packet from router from 01 since 01 is same subnet as Node B