RCOMP - Redes de Computadores (Computer Networks)

2025/2026

Theoretical-practical lesson 05

- IPv4 routing and static routing tables.
- IPv4 dynamic routing.

Routing

Routing is the activity of analysing a packet's **destination address** and, in conformity, decide **where to send it**, thus every node executes this task. Nevertheless, we usually only name as router a node that performs this task for packets that have been sent by other nodes and not by the node itself.

To make decisions about to where packets should be sent, some knowledge about other nodes location is required. On a layer two Ethernet switch, that information is stored in the MAC table, where each node location is stored.

By convention, the expression **routing** is reserved for layer three, whereas in layer two, the equivalent activity is named as **switching**.

Unlike switching in layer two that handles with node addresses one by one, routing in layer three uses network addresses, this reduces the amount of information to be handled. Instead of specifying every node address in a network, specifying the network's address is enough. This information is stored in the **routing table**, routing tables may be manually or automatically built.

End-nodes are nodes that only send packets of their own and don't retransmit other nodes' packets (packet forwarding is disabled). Still, every end-node is also required to have a routing table.

IPv4 routing decisions

Every IPv4 node is required to have a routing table, it's based on the routing table that the node takes routing decisions.

A routing table is made of a column with a sequence of **known networks**, each specified by a network address and a network prefix-length (or network mask).

The routing activity consists of confronting a packet's IPv4 destination address with each of these **known networks**.

This confrontation aims at determining if the packet's **destination address** belongs to that **known network**. Routers do this by:

1st – Performing a bit-by-bit logical **and** operation between the packet's destination IP address and the network mask of the known network.

2nd – Then, checking if the result is the known network address. If so, this means that such destination address belongs to that network, it's a match and the remaining routing table will be ignored. Otherwise, proceed for the same confrontation with the next known network in the routing table (next line).

In essence, the router first assumes the packet destination address may belong to the known network and performs the test to check if that's so.

The routing table

When a node wants to send a packet, and the packet's destination address belong to a known network, the routing table also has instructions about how to transmit that packet, that's the role of the routing table's second column, usually called the **next-hop**. There are two options for how to transmit a packet: direct transmission (local interface) or using a neighbour router (a local IP address).

Direct transmission means the network to be reached is directly connected to the node; thus, no third parties (routers) are required to intervene.

Using a neighbour router means the network to be reached is not directly connected to the node, thus the packet must be sent to a neighbour router. That router will, in turn, confront the packet destination address with its own routing table and act accordingly, but that's another story, and not a concern for this node anymore.

The details about how the packet is transmitted in each of these two cases differ. On **direct transmission**, ARP is used to obtain the MAC address matching the IPv4 destination address, then a layer two frame is created, and the MAC address is used to fill the frame destination address, the IPv4 packet is placed as payload, and the frame is directly sent to the destination node.

Using a neighbour router

In contrast, if transmitting a packet requires the use of a neighbour router (next-hop), then ARP is used to obtain the **MAC address of the neighbour router**. A layer two frame is created, and the router MAC address is used to fill the frame's destination address, the IPv4 packet is placed as payload and the frame is sent to the neighbour router.

This is why the next-hop must always be an address of a directly connected network, otherwise, ARP can't do its job.

Here is an example of a routing table:

Network	Next-hop
192.160.0.0/16	Direct – Interface 1
192.150.0.0/16	Direct – Interface 2
192.140.0.0/24	192.160.0.1
195.10.9.0/27	192.160.0.1
15.20.50.0/24	192.150.0.1

We can see this node is aware of five networks, two of which are directly connected to it. In addition, three other **remote networks** are available by using **neighbour routers**.

As stated before, neighbour routers must belong to a directly connected network. Also notice that this routing table tells nothing about how far the remote networks are, they could be connected to the mentioned neighbour router or on the other side of the world.

The default route

Imagine a routing table for an IPv4 node connected to the internet, it can't hold all networks existing over the internet, however, **if a packet doesn't belong to a known network in the routing table, it won't get transmitted**.

This issue is solved by adding a default route to the table, it's represented by network 0.0.0.0 with prefix length zero. This means, any destination IPv4 address belongs to this network (because the network mask is zero, the bit-by-bit and logical operation with any address will always result in 0.0.0.0).

Of course, the default route always ought to be the last row in the routing table, in fact, subsequent lines will never be reached.

The next-hop entry for the default route is usually called the **default-router** or the **default-gateway**, if there's an internet connection, that should be the neighbour router that is suited to send packets toward the internet.

If other networks have the same next-hop as the default-route, they can normally be removed from the routing table. If those lines are removed, then packets previously matching them will fall to the default-route, and thus, will end up being sent to the same next-hop.

End-nodes' routing table

Every IPv4 node has a routing table, in the simplest case, it's derived from its own network configuration. The most basic scenario is an end-node with a single network interface, in that case, the configuration data is the local **IPv4 node address**, the **network mask**, and also the **default gateway**. The routing table is based on this data. Example:

Interface IPv4 address: 193.100.10.50

Interface network mask: 255.255.255.0

Default gateway: 193.100.10.1

Then, the routing table comes as:

Network	Next-hop
193.100.10.0/24	Direct – Interface
0.0.0.0/0	193.100.10.1

Usually, routers have more extensive routing tables, even though for better performance they should be kept as small as possible. Often, routing tables can be simplified, this means, reducing the number of lines but keeping the same behaviour. One already seen simplification case encompasses the aggregation to the default-route.

Aggregation to the default route

Simplifying routing tables is essential, it consists of combining several lines of a routing table into a single line, and yet, keeping the same behaviour. There are several rules to obey, the first one is:

Two lines may be combined into one only if they have the same next-hop

Only after checking this requisite, further analysis should be carried.

The easiest simplification is aggregation to the default-route, in principle, any line with the same next-hop as the default-route may be removed. There are exceptions to this rule, here is one example:

Network	Next-hop
193.100.10.0/24	Direct – Interface
120.5.10.0/24	193.100.10.1
120.5.0.0/16	193.100.10.5
0.0.0.0/0	193.100.10.1

The second line has the same next-hop as the default-route; however, it can't be removed because that would change the behaviour.

This happens because the third line's network includes the second line's network, as it is, packets with destination address belonging to network 120.5.10.0/24 are sent to 193.100.10.1. If the second line was removed, they would match the third line and would be sent to 193.100.10.5. **Order matters.**

Networks aggregation

One other chance to simplify routing tables is by combining two networks into a single, double sized equivalent network. This should be tried only after aggregation to the default route. Once more, the next-hop has to be the same.

Though, another criterion must be met, **both networks to be aggregated must have the same prefix-length** (same network mask).

Aggregating two networks is achieved by reducing both networks prefix-length in one bit. As result, one bit that was previously a network bit will become a host bit. This may or not lead to a change in each network's address, if both result in the same address, then they can be aggregated. Examples:

170.10.1.0/24 and 170.10.0.0/24: If we reduce one bit to the prefix-length, the first network will become 170.10.0.0/23, while the second will keep the same address 170.10.0.0/23, we can therefore say 170.10.0.0/23 = 170.10.1.0/24 + 170.10.0.0/24.

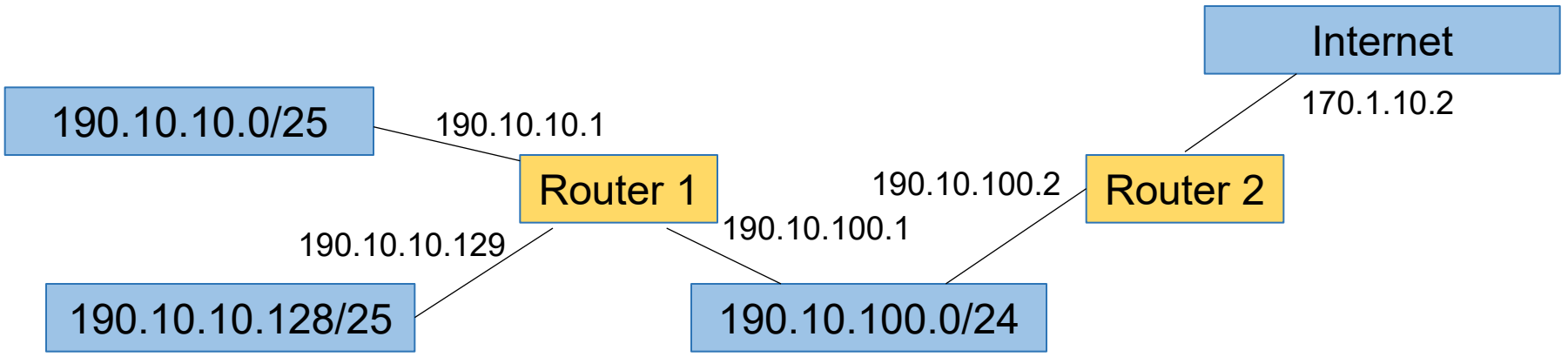
170.10.1.0/24 and 170.10.2.0/24: If we reduce one bit to the prefix-length, the first network becomes 170.10.0.0/23, while the second keeps the same address 170.10.2.0/23, unlike the previous example, **they cannot be aggregated**.

Building a routing table

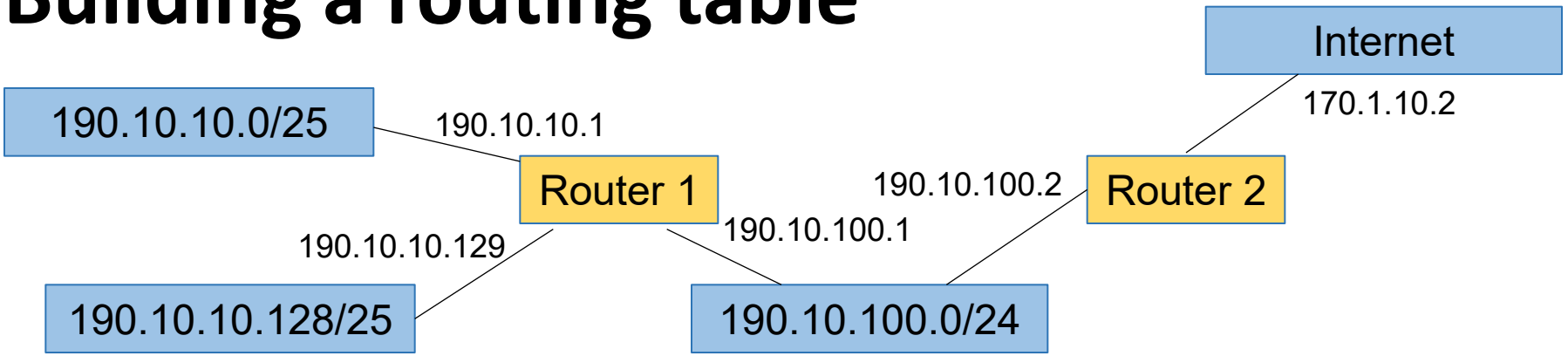
Routing tables can be manually built on each router by the network administrator (**static routing**), or they can be automatically built using routing protocols (**dynamic routing**). To build a routing table we must know the routers' relative positions and networks that interconnect them.

Each router will have its own routing table, to build it, we must focus on that router's position and, from its point of view, **list all remote networks** (not directly connected to the router). Directly connected networks are already known by the router and there's no point in adding them to the routing table.

If the infrastructure has an internet connection, we must also add a default route on each routing table. Let's take an example:



Building a routing table



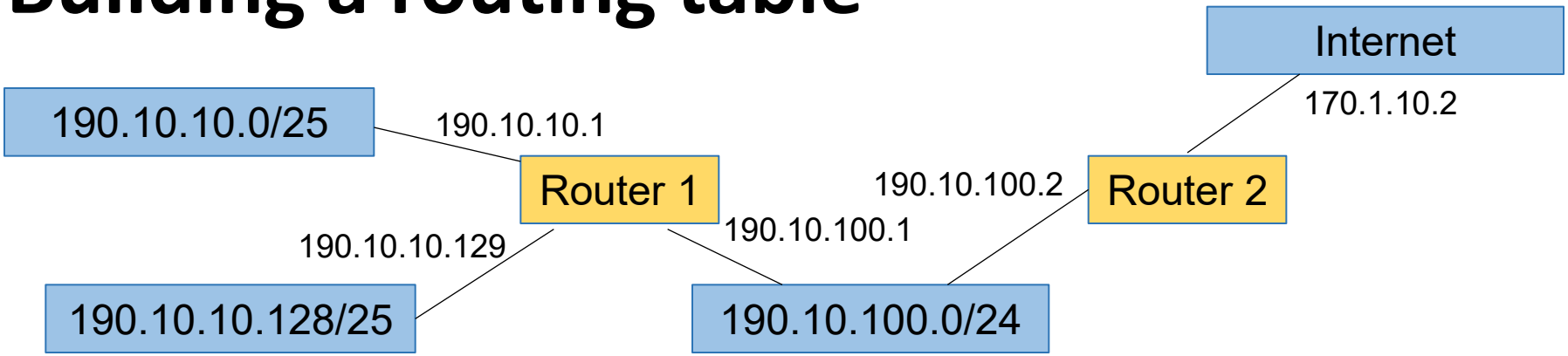
In this layout we have two routers and three local networks, Router 2 has an internet connection, near each router connection the router's IPv4 address is shown in the diagram.

The routing table for router 1 is simple, it knows all local networks, so there is no need to add any of them to its routing table, so the routing table for Router 1 requires only the default route:

Network	Next-hop
0.0.0.0/0	190.10.100.2

The meaning of this table is: any packet with a destination address not belonging to a directly connected network (190.10.10.0/25, 190.10.10.128/25 or 190.10.100.0/24) must be sent to 193.10.100.2 (Router 2).

Building a routing table



Router 2 is not aware networks 190.10.10.0/25 and 190.10.10.128/25 exist (because it's not directly connected to them). So, we must add then to the routing table, also saying that to reach them it should send the packets to 190.10.100.1 (Router 1).

Router 2 is connected to network 190.10.100.0/24, so there is no need to add it. The default route for router 2 will send packets to the internet router (170.1.10.2).

Network	Next-hop
190.10.10.0/25	190.10.100.1
190.10.10.128/25	190.10.100.1
0.0.0.0/0	170.1.10.2

Building a routing table

We can try a simplification for **Router 2** routing table.

Network	Next-hop
190.10.10.0/25	190.10.100.1
190.10.10.128/25	190.10.100.1
0.0.0.0/0	170.1.10.2

Aggregation to the default route is not possible because the next-hop is different. However, networks 190.10.10.0/25 and 190.10.10.128/25 have the same mask, and the addresses are very similar, they may become the same if we reduce one bit to the network mask. Indeed, representing the network bits in red:

190.10.10.0/25 -> **190.10.10.(0**000 0000)$_2$

190.10.10.128/25 -> **190.10.10.(1**000 0000)$_2$

This means if we reduce one bit to the mask, both address will become the same: **190.10.10.0/24**. The simplified version of the routing table is therefore:

Network	Next-hop
190.10.10.0/24	190.10.100.1
0.0.0.0/0	170.1.10.2

Dynamic routing

An IPv4 networking infrastructure may be redundant, this means there are several paths to reach the same destination network. That can be clearly seen in the following routing table:

Network	Next-hop
155.0.0.0/24	190.10.100.1
155.0.0.0/24	170.1.10.2
155.0.0.0/24	190.10.100.5

This table shows us, on this router, there are three alternative next-hops to reach network 155.0.0.0/24. The question is: what's the best choice?

To be able to answer that, an additional column must be added representing the cost of each alternative, the lowest cost alternative should be elected.

Network	Next-hop	Cost/Metric
155.0.0.0/24	190.10.100.1	4
155.0.0.0/24	170.1.10.2	6
155.0.0.0/24	190.10.100.5	1

In this new version we can now see the best option is 190.10.100.5. However, networks are dynamic, later in the day router 190.10.100.5 may crash, and then what?

With static routing, the network administrator would have to rush and go to every router removing all routes passing through the crashed router. **This is when dynamic routing becomes a must.**

Dynamic routing

Dynamic routing uses a routing protocol. The routing protocol establishes a dialogue between routers, leading them to cooperate and automatically build the routing table in all of them. **More important than building the tables once, they are permanently updated to reflect the network status**.

Also, dynamic routing enforces an effective load balancing when there are several alternatives. Each router always selects the lowest cost alternative it can find on the routing table, but cost values in routing tables are permanently updated by the routing protocol. For instance, if one router is heavily loaded with traffic, the routing protocol will increase the cost of using that alternative on all routers' tables.

Also, if one router is simply unavailable, the routing protocol removes that alternative from all routing tables.

There are similarities with redundant layer two infrastructures and STP failover, however, unlike with STP, all alternatives are permanently active and in use. With dynamic routing the traffic is distributed by all available alternatives with effective load balancing and not simple failover.

Cost/metric

Each router interacts with neighbour routers only (those connected to common networks), these neighbour routers represent all the options available to retransmit packets. Such retransmission represents only a single step (a hop) in the path of a packet to reach the destination network. Along the path, all routers must take the correct decision on transmitting the packet to the next router, otherwise, the packet will not reach the destination network.

If there are alternative paths to reach a destination network, we are interested in using the best available, however, routers are not aware of paths, they handle with next-hops (neighbour routers) only.

However, the cost/metric inserted into the routing table by the routing protocol represents **the path cost** of using each alternative next-hop. Thus, the only thing the router is required to do is select the lowest cost/metric alternative in the routing table.

In fact, when a router has alternative paths to reach a network, it will automatically remove those with higher cost, leaving only the best one. So, when the router is analysing a packet's destination address, the best alternative has already been selected.

The path cost/metric

Different routing protocols use different metrics, in the first place, there's a **hop or link metric**, this represents the cost of sending to a neighbour router and its calculation can take in account values related to the link used, like transmission rate, error rate, MTU and values related to the next hop like its performance, latency or current traffic load.

We are though interested in the **path metric**, and this is what routing protocols calculate and insert into routing tables.

The **path metric** represents the cost of, at a given router, using one particular next-hop, that ultimately represents following one path to reach the destination network.

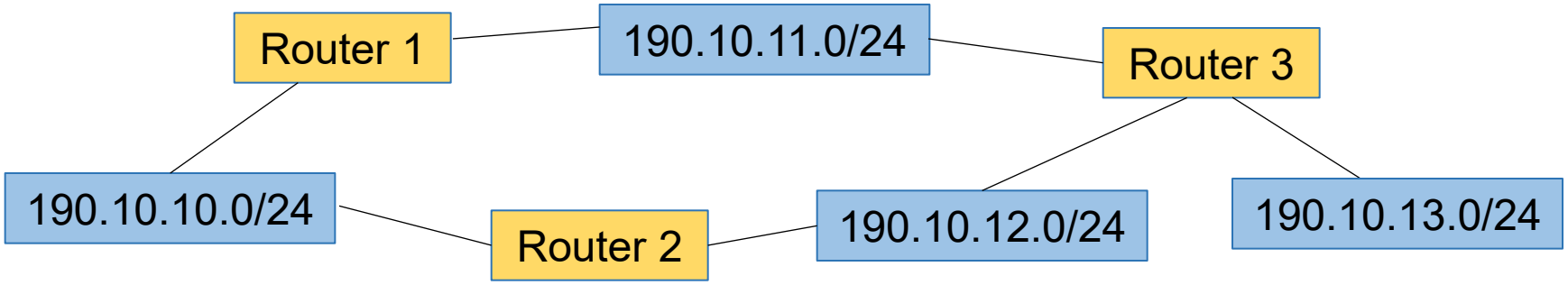
Routing protocols calculate the path metric by taking into account all hop or link metrics along the path, the straightest way to do this is simply calculating the sum of all hop or link metrics along the path, but other methods can be used.

RIP (Routing Information Protocol)

RIP is a very simple distance-vector routing protocol. Distance-vector means the knowledge about existing networks is received from neighbour routers, incorporated in the local routing tables and retransmitted to other neighbour routers. In this process, each router updates the path cost by adding to it the cost of the interface through which the information was received.

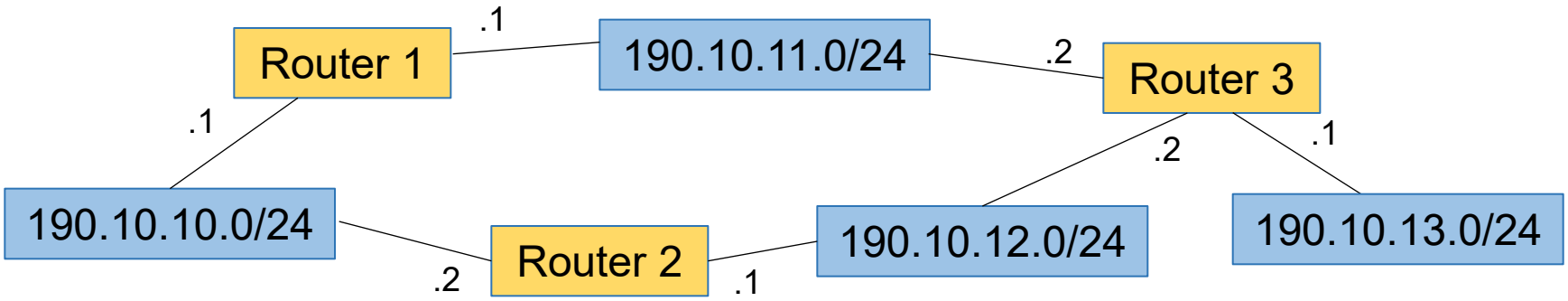
In RIP, the cost of each interface is typically 1, greater integer values can be used, however, when a path cost reaches 16, it will be accounted as unreachable. If all interfaces have cost 1, then the path metric will represent the number of hops to reach the destination network (routers to pass thru) .

As an example, let's consider the following network infrastructure:



RIP example – initial knowledge

As usual in RIP, we assume every interface has cost one. In the image below we can also see the rightmost octet of each router address in each network.



Before RIP starts to broadcast information about known networks, each router already knows the networks it's directly connected to, so routing tables are:

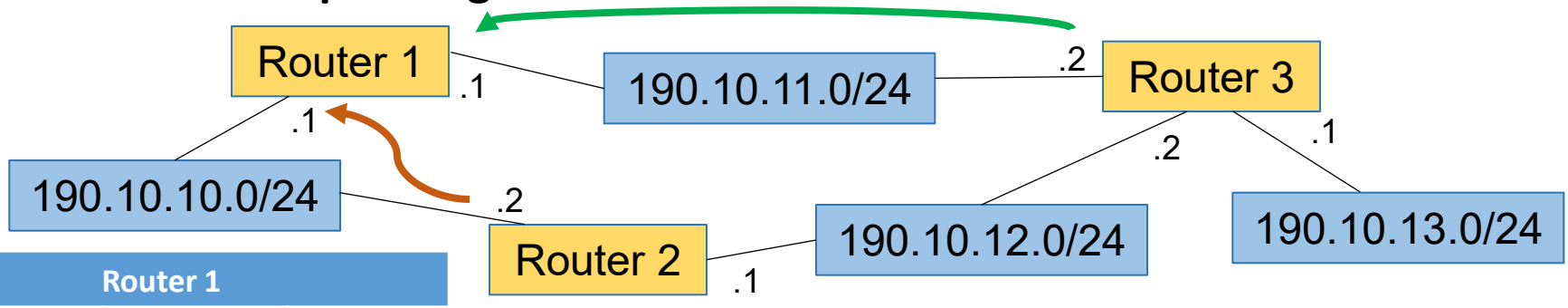
Router 1		
Network	Next-hop	Cost
190.10.10.0/24	Direct	1
190.10.11.0/24	Direct	1

Router 2		
Network	Next-hop	Cost
190.10.10.0/24	Direct	1
190.10.12.0/24	Direct	1

Router 3		
Network	Next-hop	Cost
190.10.13.0/24	Direct	1
190.10.11.0/24	Direct	1
190.10.12.0/24	Direct	1

RIP example – receiving RIP tables

Now RIP starts broadcasting, each router broadcasts a **table with the known networks and corresponding metric**.



Router 1		
Network	Next-hop	Cost
190.10.10.0/24	Direct	1
190.10.11.0/24	Direct	1
190.10.10.0/24	190.10.10.2	2
190.10.12.0/24	190.10.10.2	2
190.10.11.0/24	190.10.11.2	2
190.10.12.0/24	190.10.11.2	2
190.10.13.0/24	190.10.11.2	2

Let's focus on Router 1, though the same will be happening in all routers.

Router 1 receives one table from Router 2 (in red) and another table from Router 3 (in green).

When a router receives the RIP table it will add the incoming interface cost (one) and **sets the next-hop to the source address from where the table came from**.

The resulting routing table in Router 1 is shown on the left.

RIP example – tables cleanup

Next step is tables cleanup, every 120 seconds each router will remove the repeated networks with higher metric, leaving only the one with lower metric.

Router 1		
Network	Next-hop	Cost
190.10.10.0/24	Direct	1
190.10.11.0/24	Direct	1
~~190.10.10.0/24~~	~~190.10.10.2~~	~~2~~
190.10.12.0/24	190.10.10.2	2
~~190.10.11.0/24~~	~~190.10.11.2~~	~~2~~
190.10.12.0/24	190.10.11.2	2
190.10.13.0/24	190.10.11.2	2

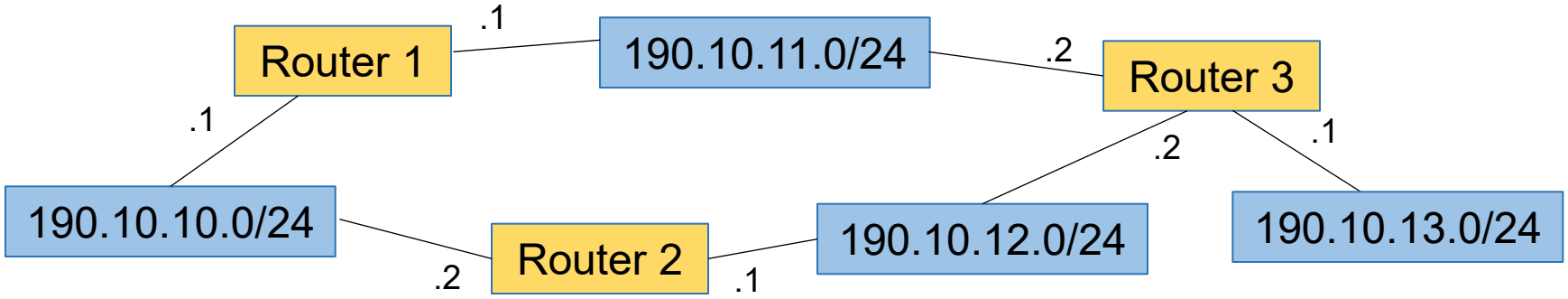
Keeping the focus on Router 1, two lines are eliminated because for each there's an alternative with a lower path cost. The remaining lines will be kept for now because existing alternatives have the same cost.

If a line is not refreshed within 180 seconds, it's removed. Information is sent between routers every 30 seconds, ultimately routing tables converge, and no further changes occur if the network stays stable.

RIP convergence time is rather long; it takes 3 minutes to detect a connection is down. Then, for each router the change is propagated through, up to 30 additional seconds must be added.

RIP example – convergence

If network connections are stable, routing tables will in due course converge. Routing tables will then look like this:



Router 1		
Network	Next-hop	Cost
190.10.10.0/24	Direct	1
190.10.11.0/24	Direct	1
190.10.12.0/24	190.10.10.2	2
190.10.12.0/24	190.10.11.2	2
190.10.13.0/24	190.10.11.2	2

Router 2		
Network	Next-hop	Cost
190.10.10.0/24	Direct	1
190.10.12.0/24	Direct	1
190.10.11.0/24	190.10.10.1	2
190.10.11.0/24	190.10.12.2	2
190.10.13.0/24	190.10.12.2	2

Router 3		
Network	Next-hop	Cost
190.10.13.0/24	Direct	1
190.10.11.0/24	Direct	1
190.10.12.0/24	Direct	1
190.10.10.0/24	190.10.12.1	2
190.10.10.0/24	190.10.11.1	2