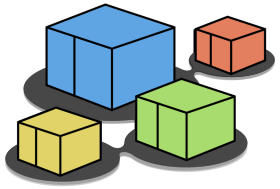
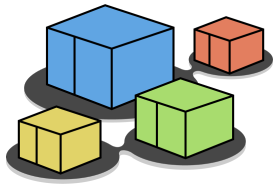


Kathará

kathara lab(s)

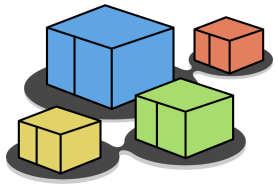
ospf with FRRouting

Version	2.0
Author(s)	L. Ariemma, T. Caiazzi, G. Di Battista, M. Rimondini, M. Patrignani
E-mail	contact@kathara.org
Web	http://www.kathara.org/
Description	A set of labs showing the operation of the ospf routing protocol in different scenarios – kathara version of an existing netkit lab



copyright notice

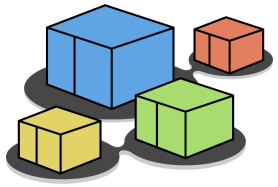
- All the pages/slides in this presentation, including but not limited to, images, photos, animations, videos, sounds, music, and text (hereby referred to as “material”) are protected by copyright.
- This material, with the exception of some multimedia elements licensed by other organizations, is property of the authors and/or organizations appearing in the first slide.
- This material, or its parts, can be reproduced and used for didactical purposes within universities and schools, provided that this happens for non-profit purposes.
- Information contained in this material cannot be used within network design projects or other products of any kind.
- Any other use is prohibited, unless explicitly authorized by the authors on the basis of an explicit agreement.
- The authors assume no responsibility about this material and provide this material “as is”, with no implicit or explicit warranty about the correctness and completeness of its contents, which may be subject to changes.
- This copyright notice must always be redistributed together with the material, or its portions.



about ospf

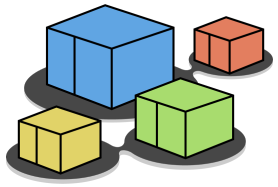
- open shortest path first
- an interior gateway protocol

	Specification	authentication Confidentiality
version 2	rfc 2328	rfc 5709
version 3 (with ipv6 support)	rfc 5340	rfc 4552



ospf: overview

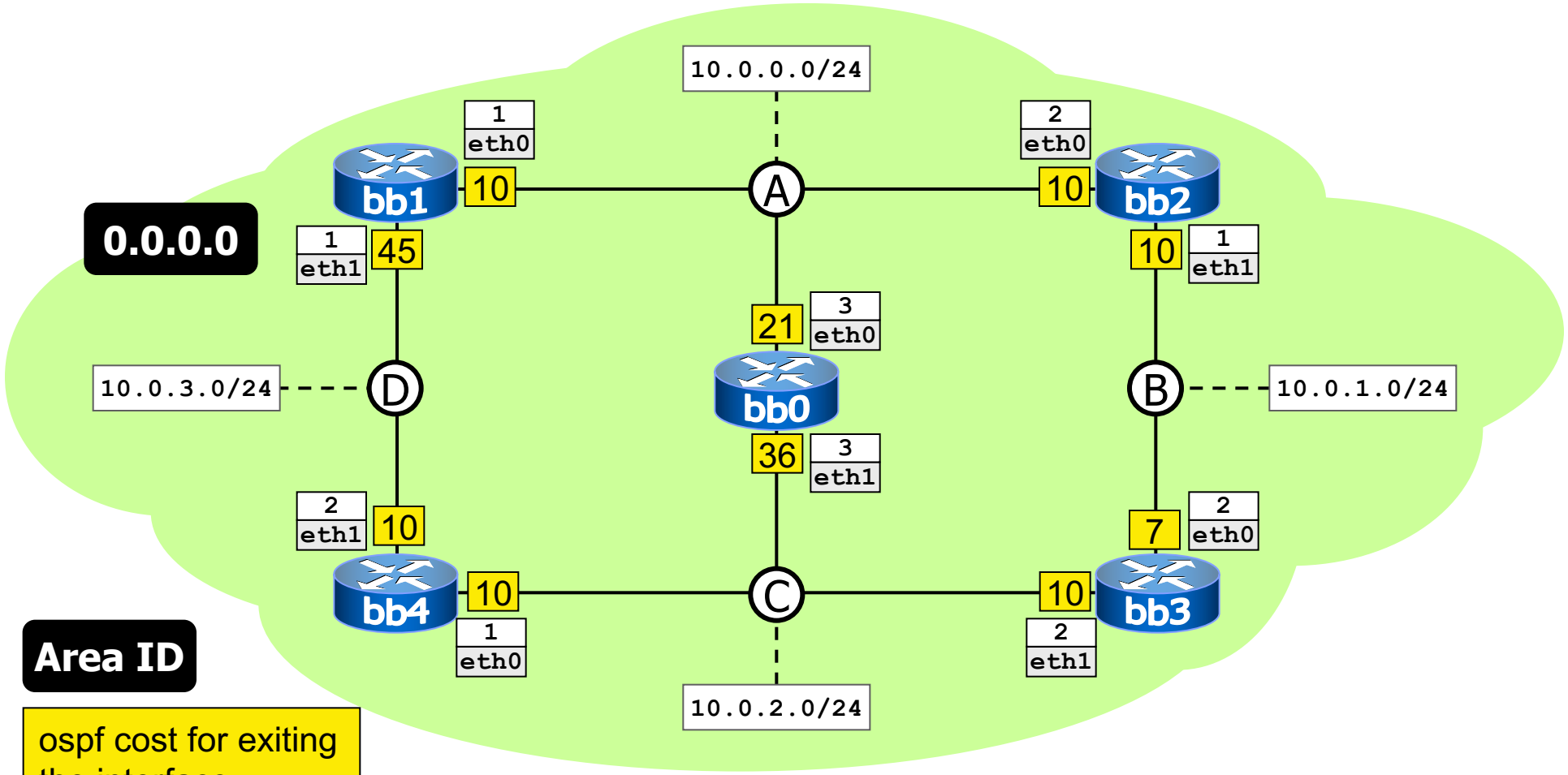
- each router floods its local state (usable interfaces, reachable neighbors) through the network, using link state advertisements (lsa)
- based on this information, each router builds and maintains a link state database (lsdb) describing the whole network topology
  - identical for (almost) all routers
  - each entry is a router's local state
- each router uses the lsdb to compute a shortest path tree rooted at itself
  - interfaces may be assigned costs

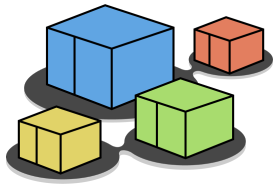


a simple ospf lab

single-area

lab topology

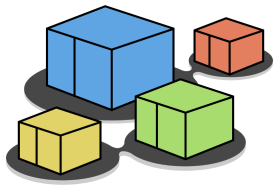




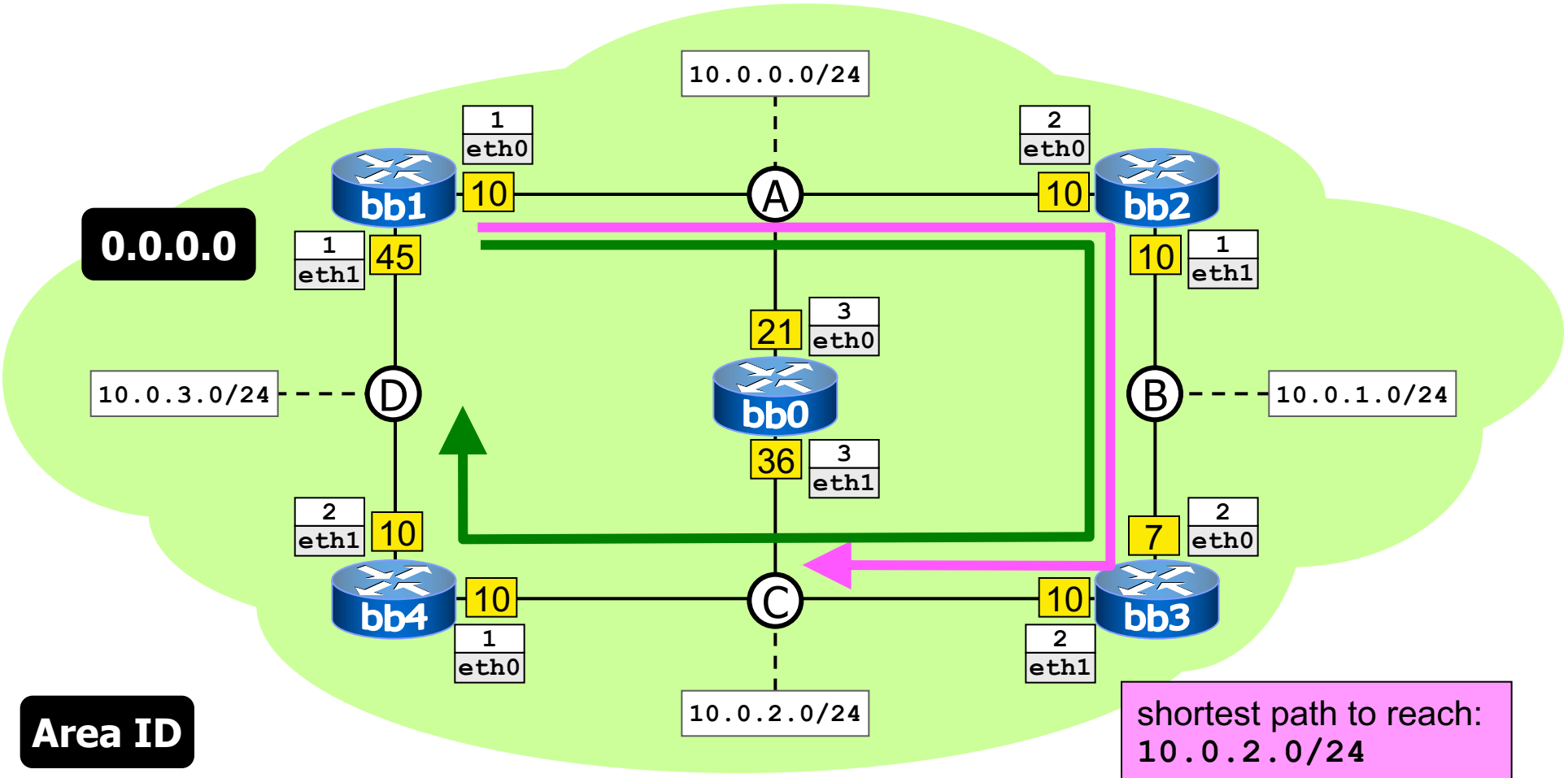
lab description

- single (backbone) area (0.0.0.0)
- each interface is assigned an ospf cost
  - default: 10
  - we have tweaked the costs to force paths taken by traffic
- to set interface costs:

```
interface eth1
ospf cost 45
```



(some) shortest paths

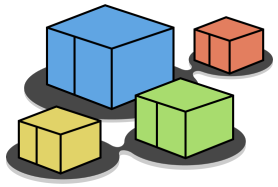


Area ID

ospf cost for exiting the interface

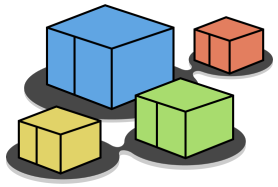
shortest path to reach: `10.0.2.0/24`

shortest path to reach: `10.0.3.0/24`



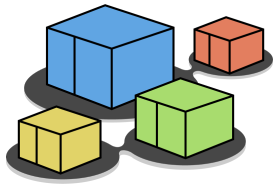
experiments

- perform traceroutes from/to different interfaces
- perform a traceroute from bb1 to 10.0.2.1
  - what path is the traceroute expected to take?
  - what path are ICMP replies expected to take?
- perform a traceroute from bb1 to 10.0.3.2
  - what path is the traceroute expected to take?
  - observe the interplay between ospf routes and directly connected networks (i.e., perform a show ip route in frr)
- try to alter the costs and observe the effect of the changes



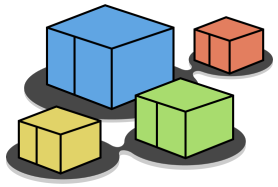
experiments

- access the ospfd cli (or the vtysh cli) on the various routers and issue the following commands:
  - **`show ip ospf route`**
  - **`show ip ospf interface`**
  - **`show ip ospf neighbor`**
  - **`show ip ospf database`**
- check that the lsdb is exactly the same for all routers



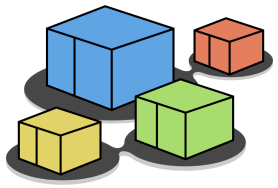
designated routers
(router interfaces designated for each network)

- for each network, one of the interfaces attached to that network is elected as designated (dr)
- priority-based election, using hello packets
  - the router (interface) sending hello packets with highest piority wins the election
  - break ties on highest router id
    - by default, a router id is the address of one of its interfaces
  - priority$\in$[0,255]
    - default priority: 1
    - priority=0 $\Rightarrow$ never become a dr
- a backup dr (i.e., the one with second highest priority) is also elected, to quickly recover from dr failures

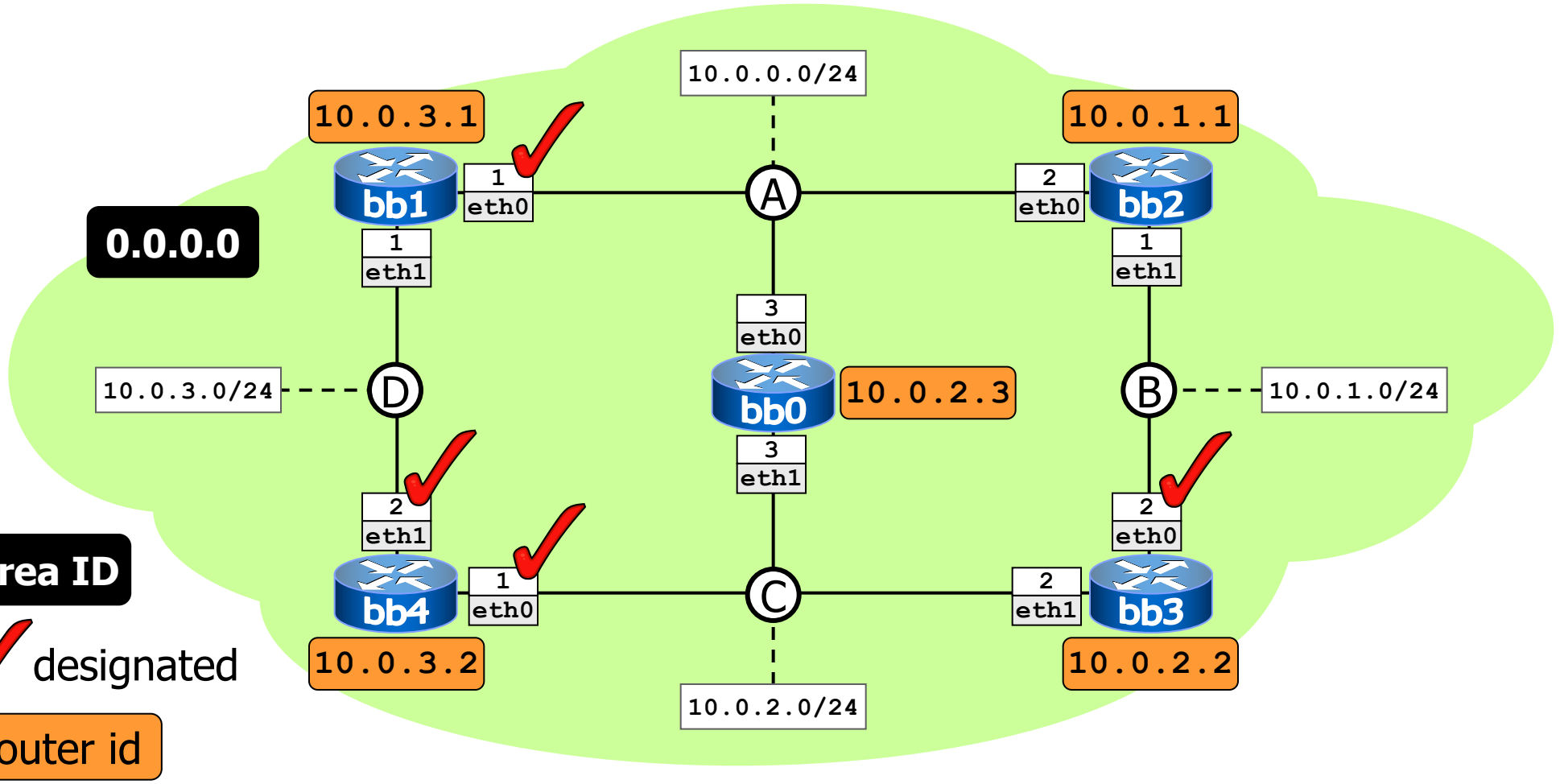


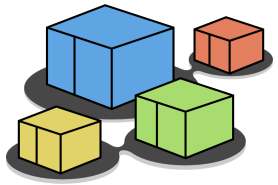
designated routers
(router interfaces designated for each network)

- a change of the dr is a change in ospf's topology model (new lsas are sent)
- for this reason, the dr is changed infrequently
  - if a router with high priority wakes up and finds that a dr already exists, it accepts that dr



designated routers
(router interfaces designated for each network)

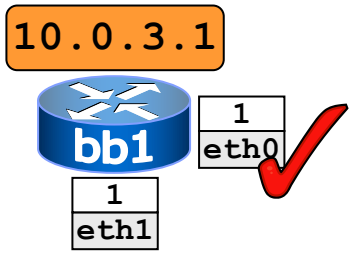




designated routers
(router interfaces designated for each network)

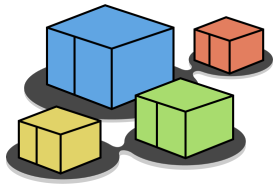
router id

✓ designated



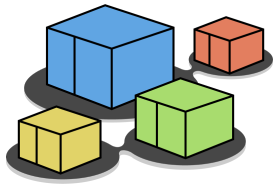
bb1

```
bb1# show ip ospf interface
eth0 is up
  ifindex 20, MTU 1500 bytes, BW 10000 Mbit <UP,BROADCAST,RUNNING,MULTICAST>
  Internet Address 10.0.0.1/24, Broadcast 10.0.0.255, Area 0.0.0.0
  MTU mismatch detection: enabled
  Router ID 10.0.3.1, Network Type BROADCAST, Cost: 10
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 10.0.3.1 I[illegible]face Address 10.0.0.1/24
  Backup Designated Router (ID) 10.0.2.3, Interface Address 10.0.0.3
  Saved Network-LSA sequence number 0x80000002
  Multicast group memberships: OSPFAllRouters OSPFDesignatedRouters
  Timer intervals configured, Hello 10s, Dead 40s, Wait 40s, Retransmit 5
    Hello due in 4.193s
  Neighbor Count is 2, Adjacent neighbor count is 2
eth1 is up
  ifindex 22, MTU 1500 bytes, BW 10000 Mbit <UP,BROADCAST,RUNNING,MULTICAST>
  ...
```

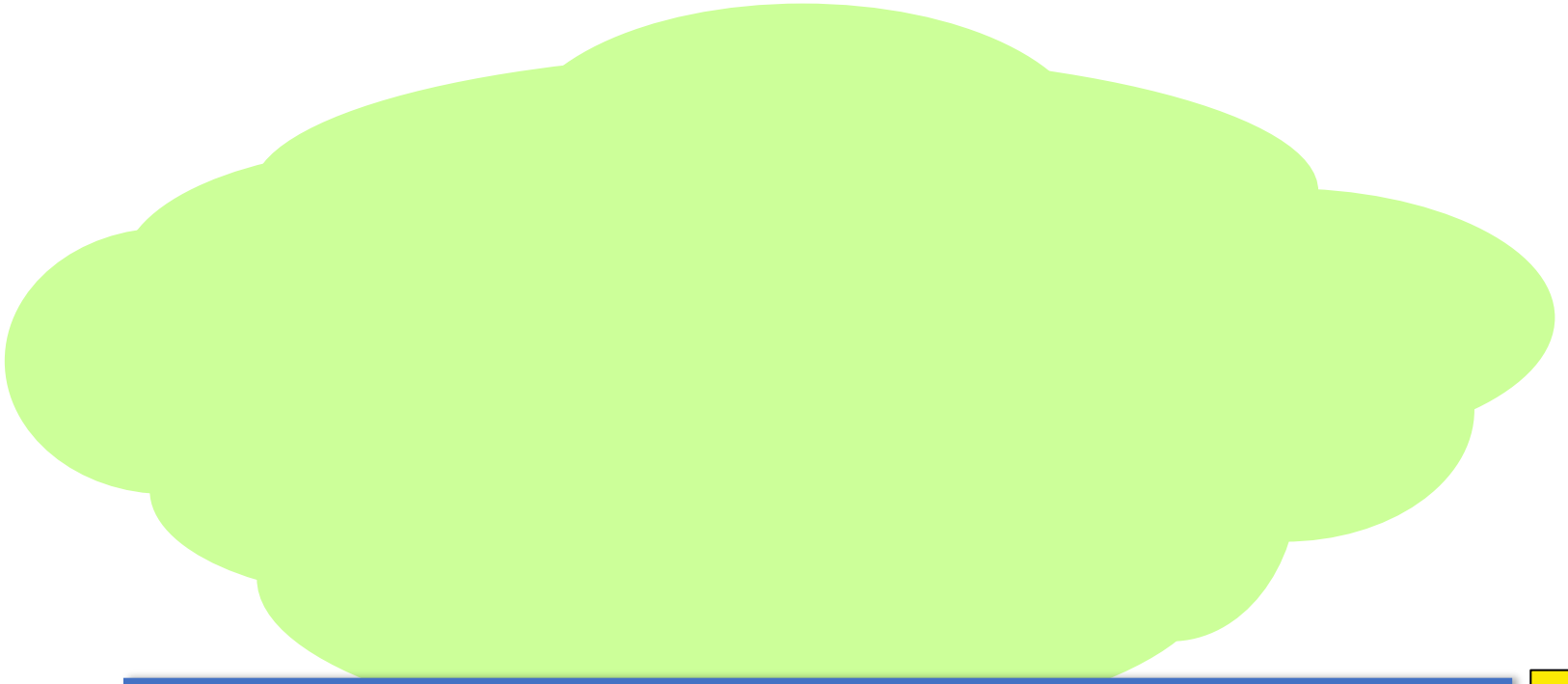


ospf's view of the network

- by exchanging link state update packets, every router learns about the complete network topology, that is:
  - routers
  - subnets
  - adjacencies between routers and networks



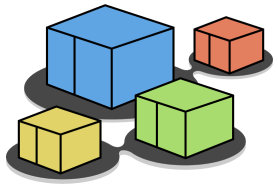
ospf's view of the network



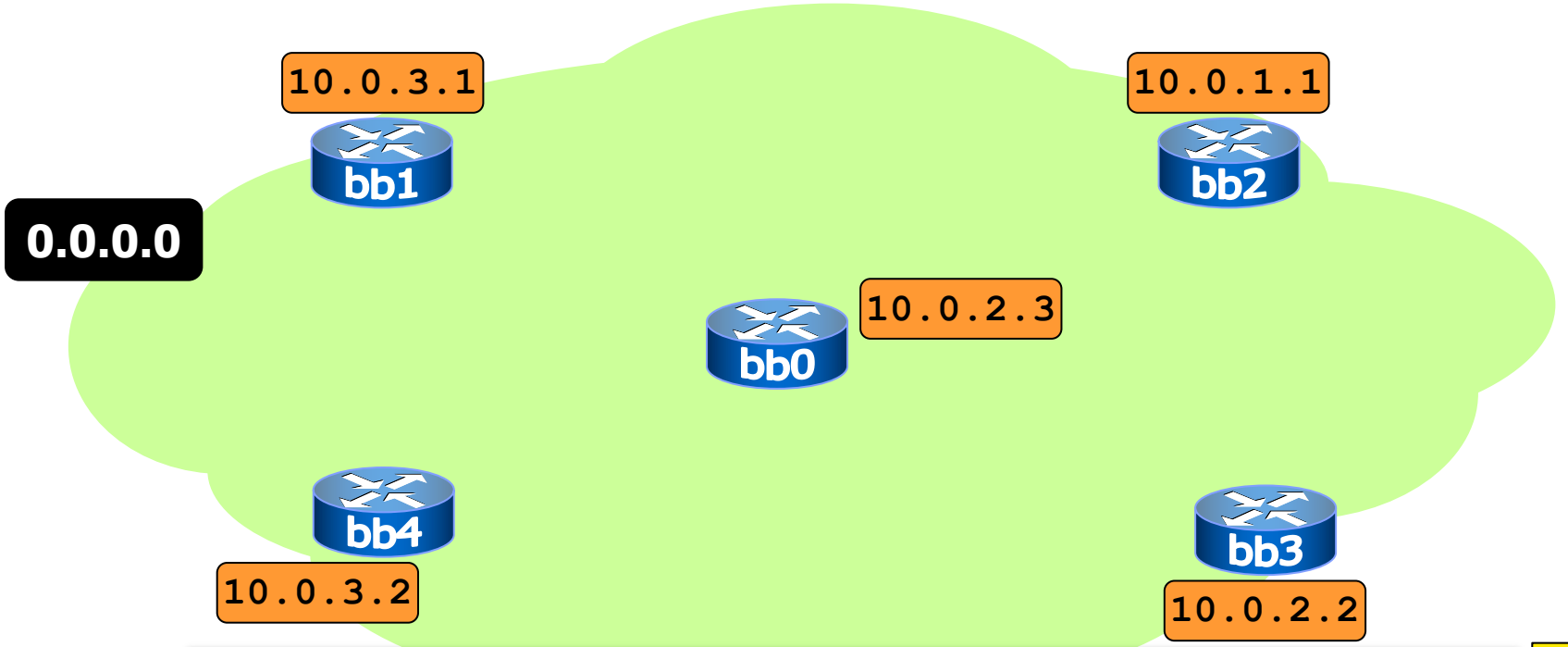
bb0

```
bb0# show ip ospf database
```

for router lsas, the `Link ID` is the router's id



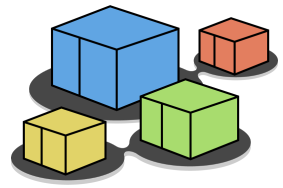
ospf's view of the network



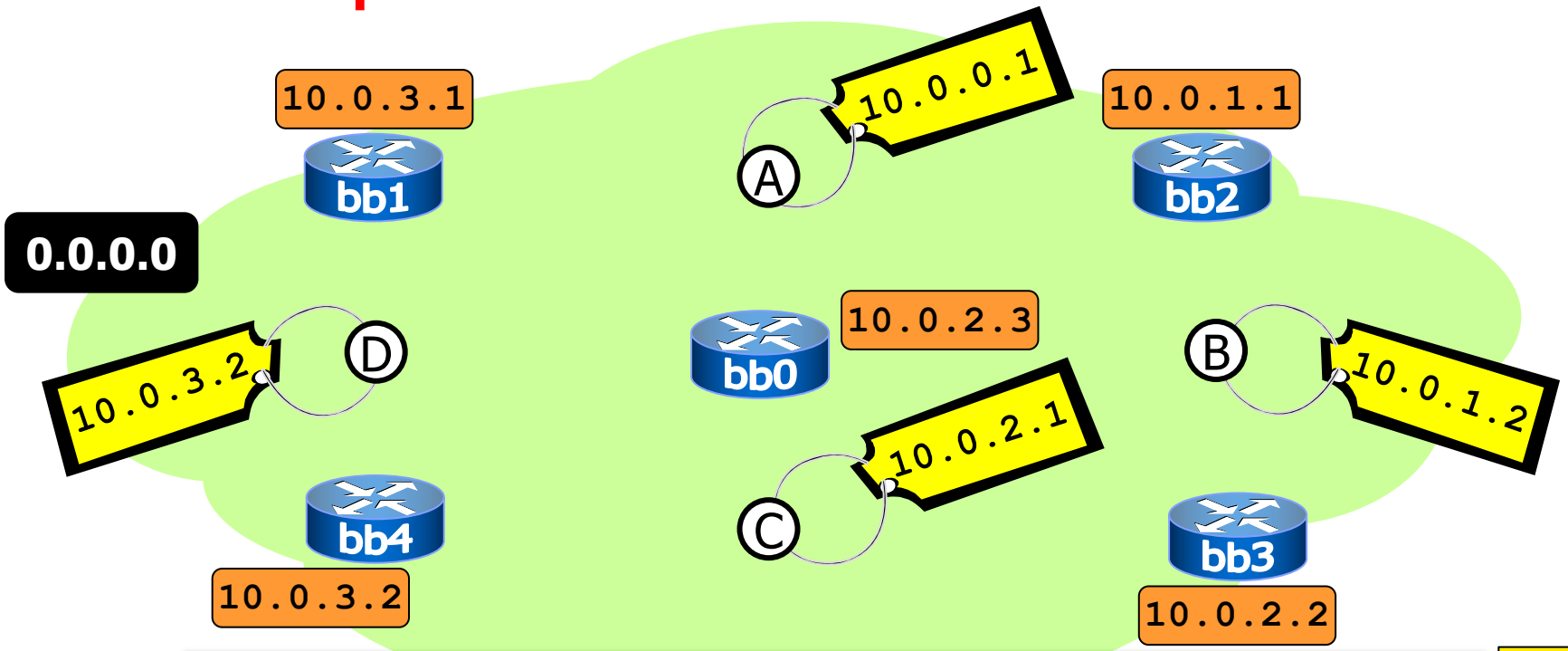
bb0

```
          OSPF Router with ID (10.0.2.3)
                Router Link States (Area 0.0.0.0)
Link ID         ADV Router      Age  Seq#       CkSum  Link
count
10.0.1.1        10.0.1.1        743 0x80000008 0xdfff 2
10.0.2.2        10.0.2.2        743 0x80000008 0xd9ff 2
10.0.2.3        10.0.2.3        742 0x8000000a 0xd9d4 2
10.0.3.1        10.0.3.1        747 0x80000009 0x268e 2
10.0.3.2        10.0.3.2        752 0x80000008 0x4091 2
```

for router lsas, the `Link ID` is the router's id



ospf's view of the network

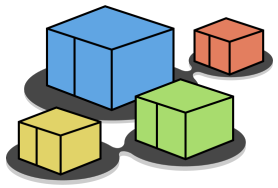


bb0

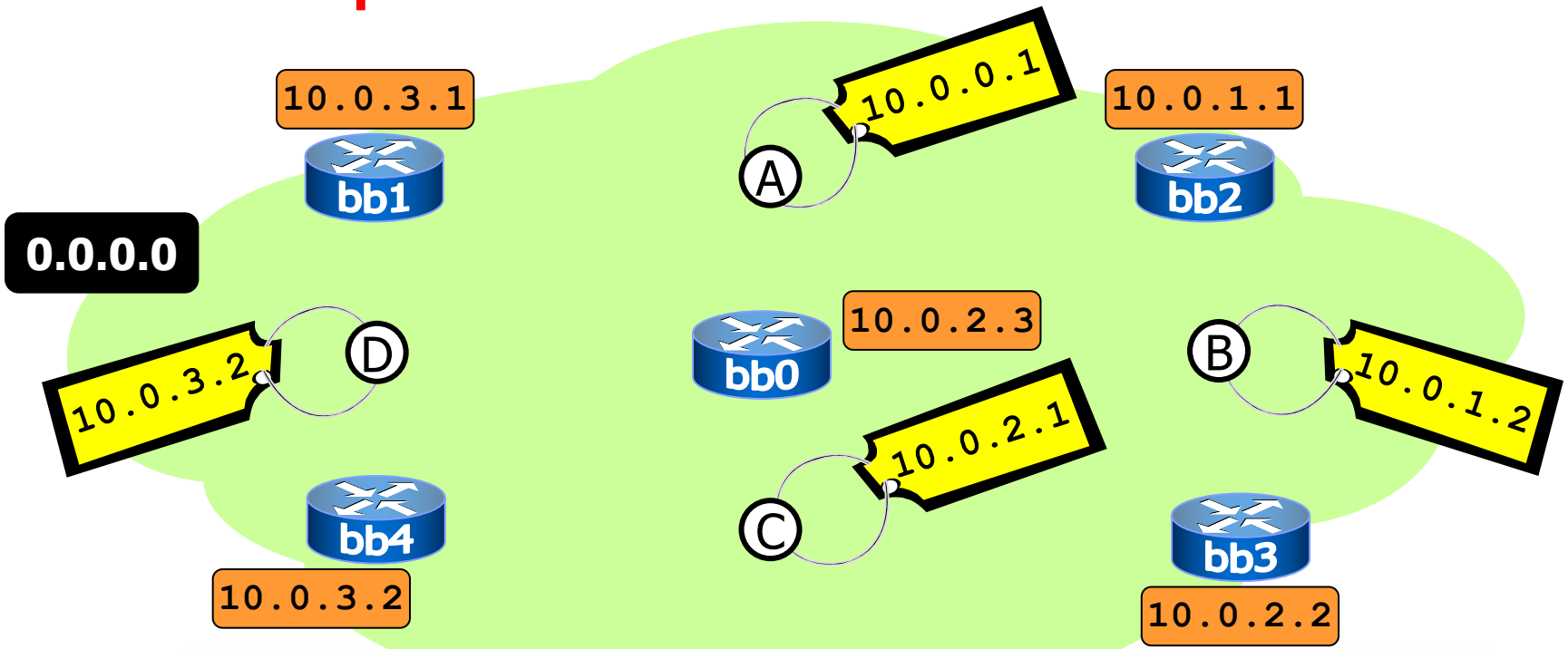
```
 Net Link States (Area 0.0.0.0)
Link ID         ADV Router      Age  Seq#       CkSum
10.0.0.1        10.0.3.1        749  0x80000002 0x69a9
10.0.1.2        10.0.2.2        743  0x80000001 0x69bb
10.0.2.1        10.0.3.2        752  0x80000002 0x729a
10.0.3.2        10.0.3.2        753  0x80000001 0x6bb3
```

for network lsas, the `Link ID` is the dr's address

last update: Oct 2021



ospf's view of the network

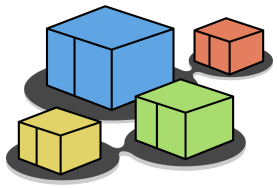


bb0

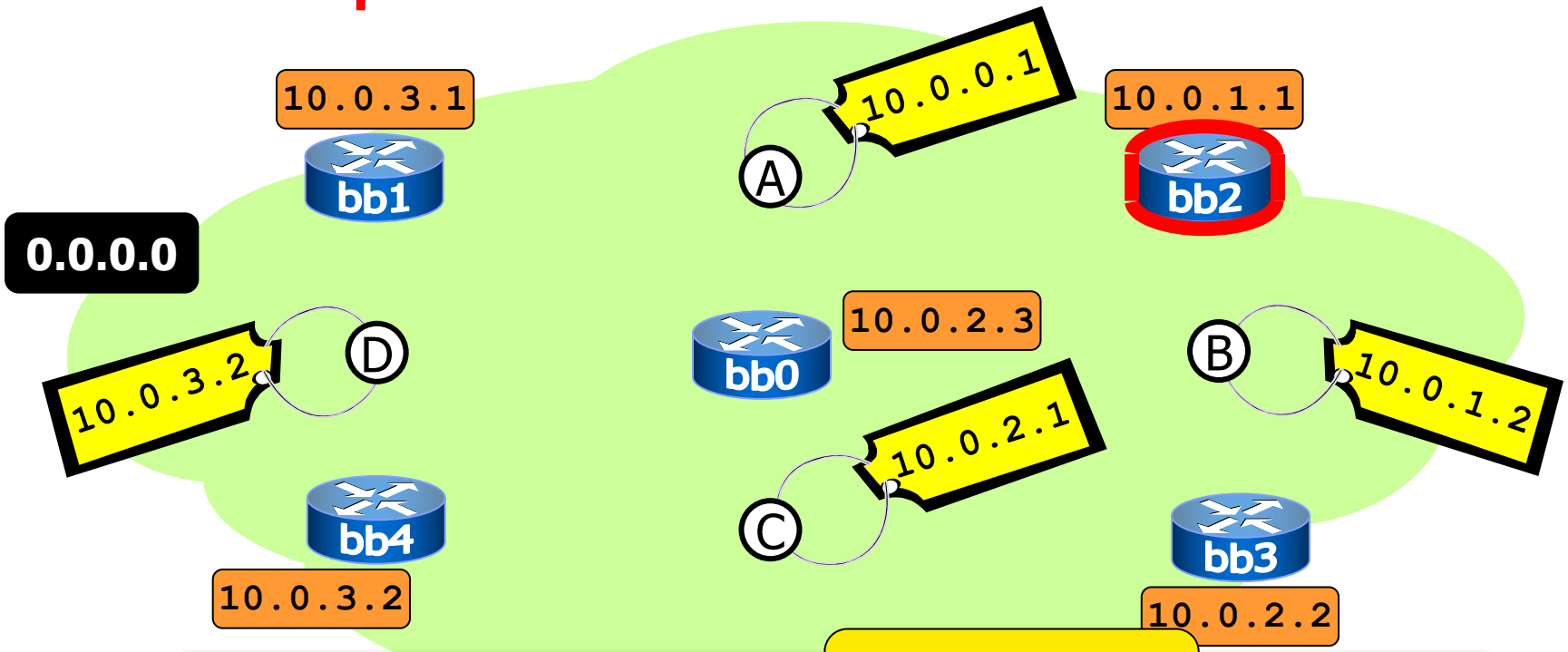
```
bb0# show ip ospf database router
 Link State ID: 10.0.1.1
   Number of Links: 2
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.0.1
     (Link Data) Router Interface address: 10.0.0.2
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.1.2
     (Link Data) Router Interface address: 10.0.1.1
```

note: the output of **`show ip ospf database router`** has been summarized

last update: Oct 2021



ospf's view of the network

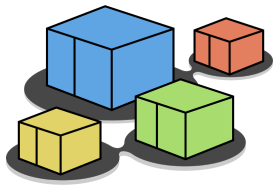



```
bb0# show ip ospf database r
 Link State ID: 10.0.1.1
   Number of Links: 2
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.0.1
     (Link Data) Router Interface address: 10.0.0.2
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.1.2
     (Link Data) Router Interface address: 10.0.1.1
```

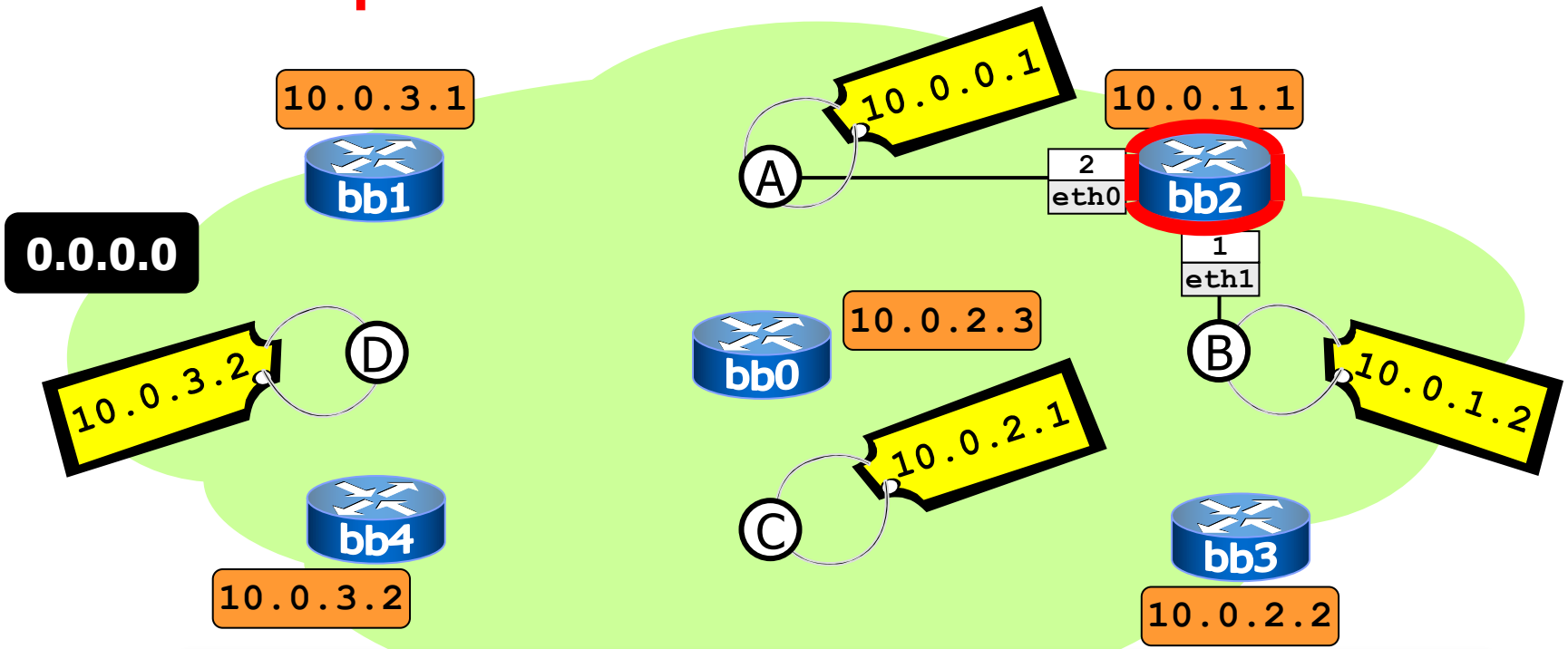
we consider this router (**bb2**)

note: the output of **`show ip ospf database router`** has been summarized

last update: Oct 2021



ospf's view of the network



bb0

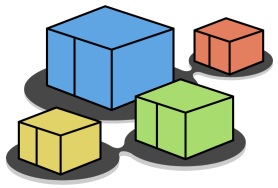
```
bb0# show ip ospf
 Link State ID: 10
   Number of Links
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.0.1
     (Link Data) Router Interface address: 10.0.0.2
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.1.2
     (Link Data) Router Interface address: 10.0.1.1
```

this router interface...

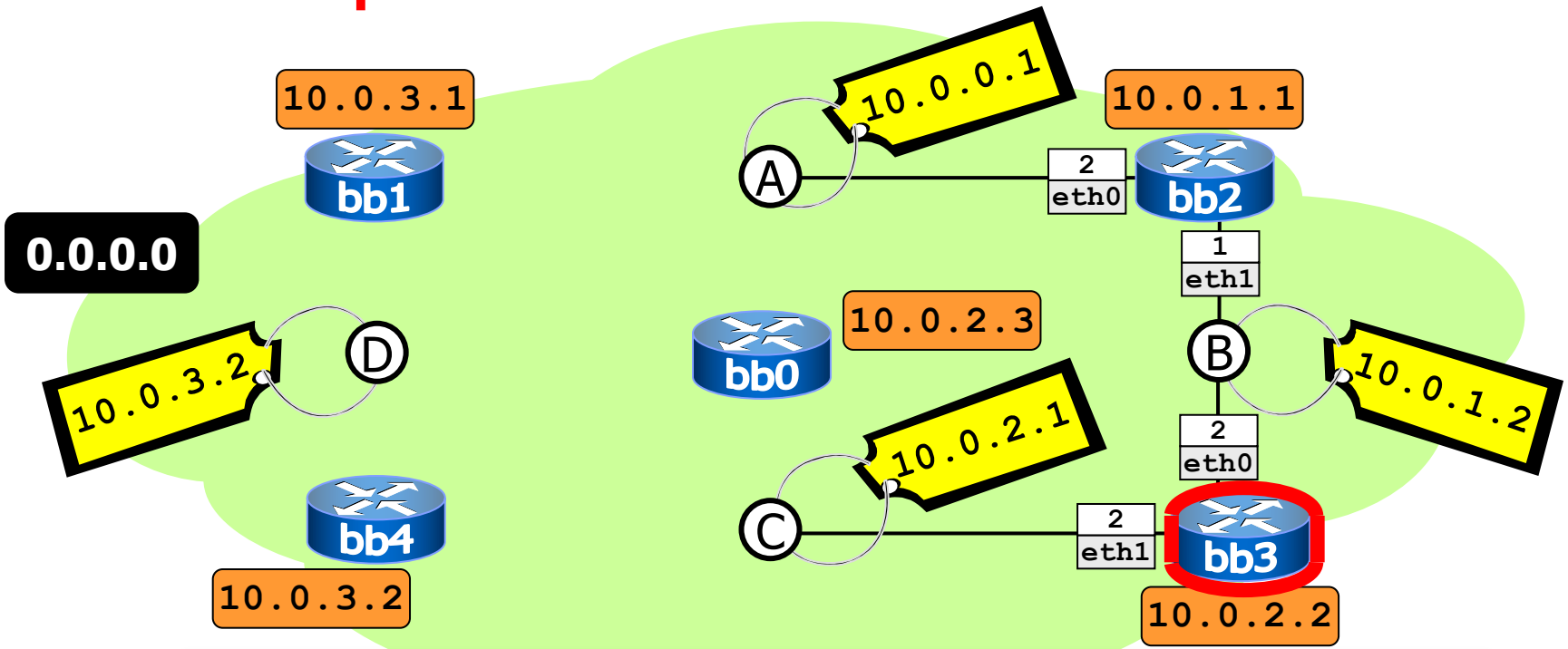
...is connected to the subnet represented by this dr

output of `ospf` `e router` has been summarized

last update: Oct 2021



ospf's view of the network



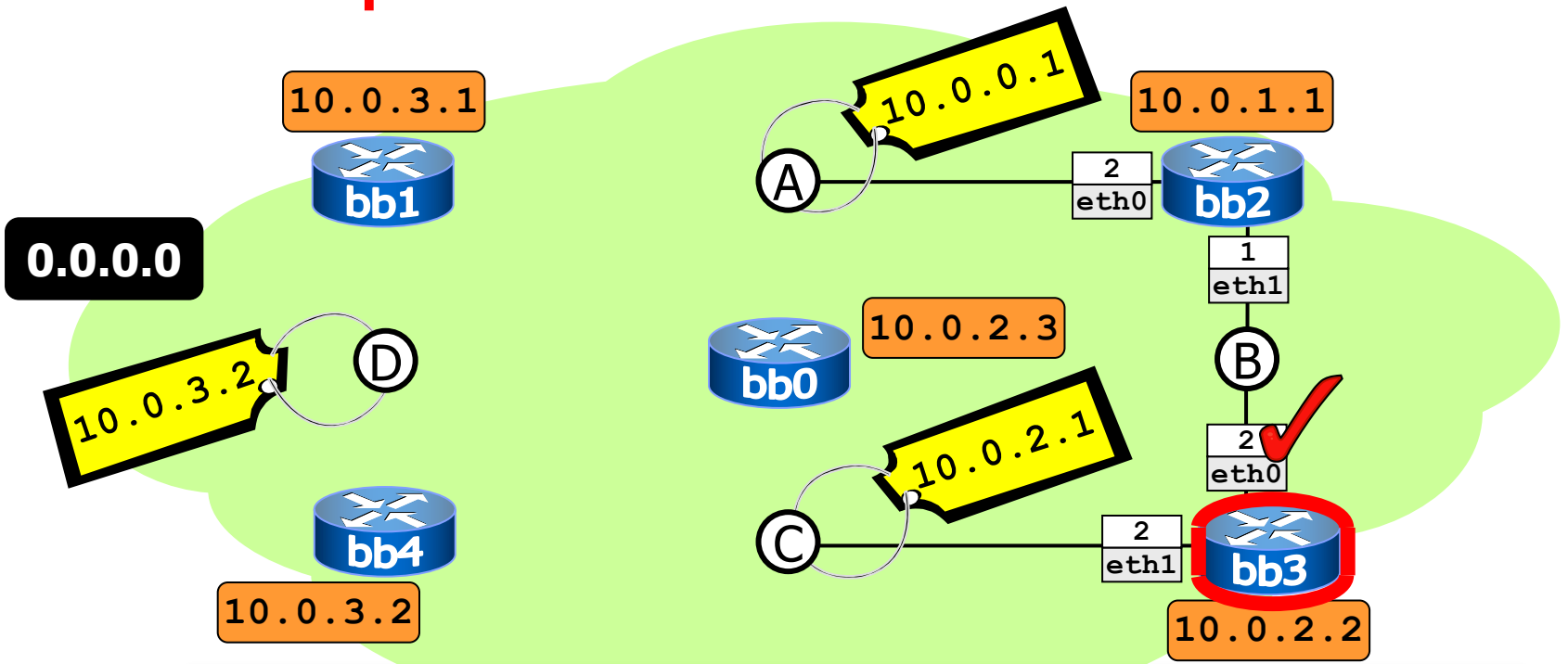
bb0

```
bb0# show ip ospf database router
 Link State ID: 10.0.1.1
   Number of Links: 2
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.0.1
     (Link Data) Router Interface address: 10.0.0.2
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.1.2
     (Link Data) Router Interface address: 10.0.1.1
```

note: the output of **`show ip ospf database router`** has been summarized

last update: Oct 2021

ospf's view of the network

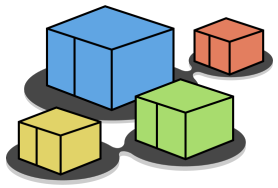


bb0

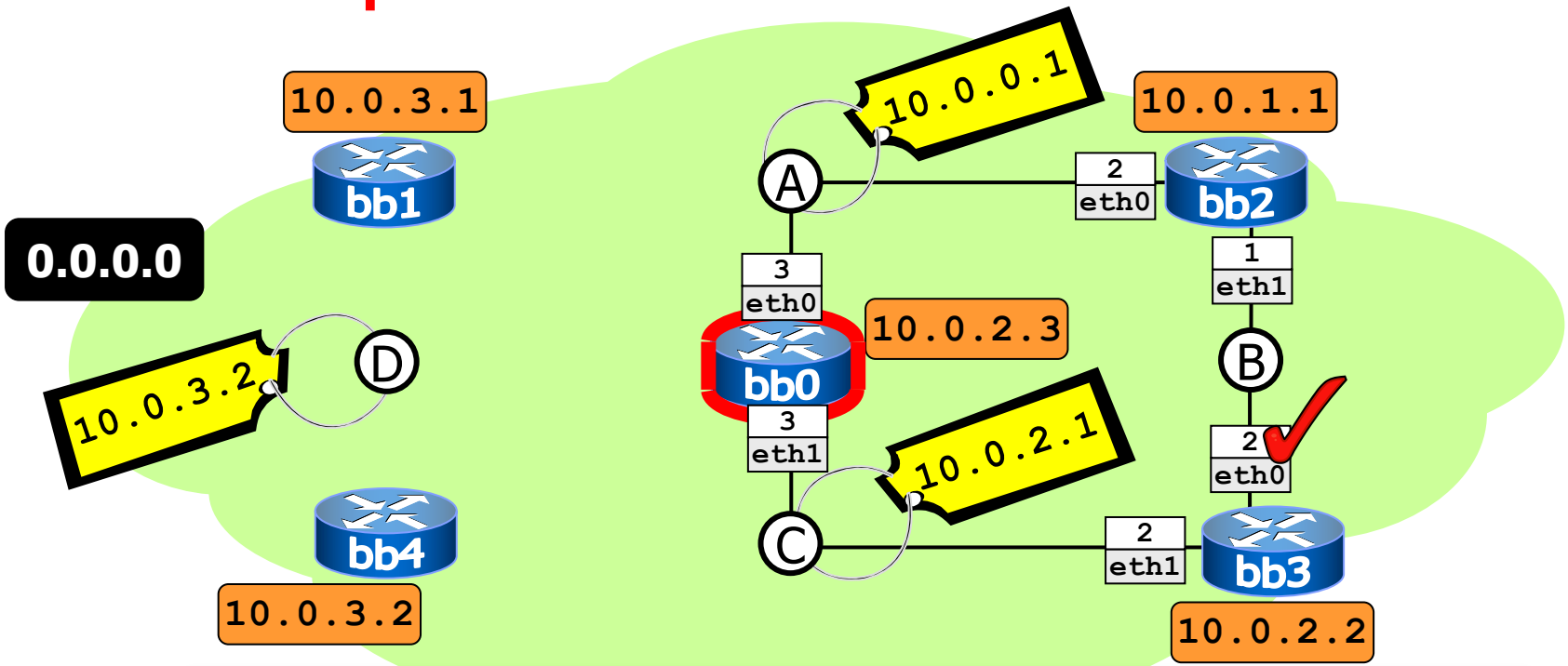
```
bb0# show ip ospf database router
 Link State ID: 10.0.1.1
   Number of Links: 2
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.0.1
     (Link Data) Router Interface address: 10.0.0.2
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.1.2
     (Link Data) Router Interface address: 10.0.1.1
```

note: the output of **`show ip ospf database router`** has been summarized

last update: Oct 2021



ospf's view of the network

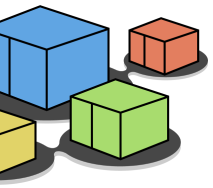


bb0

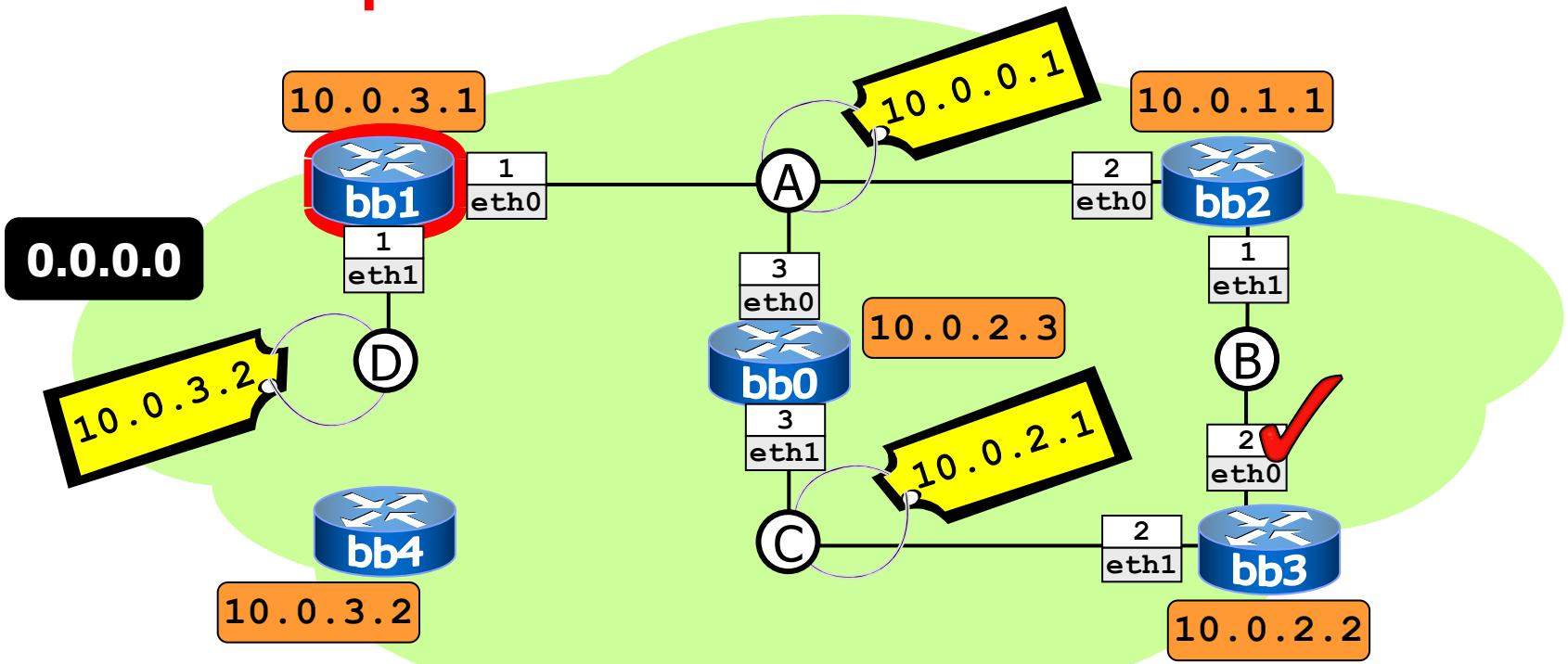
```
Link State ID: 10.0.2.3
  Number of Links: 2
   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.0.1
    (Link Data) Router Interface address: 10.0.0.3
   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.2.1
    (Link Data) Router Interface address: 10.0.2.3
```

note: the output of **`show ip ospf database router`** has been summarized

last update: Oct 2021



ospf's view of the network

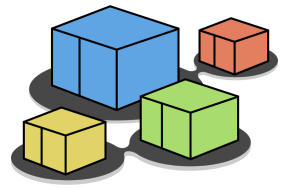


bb0

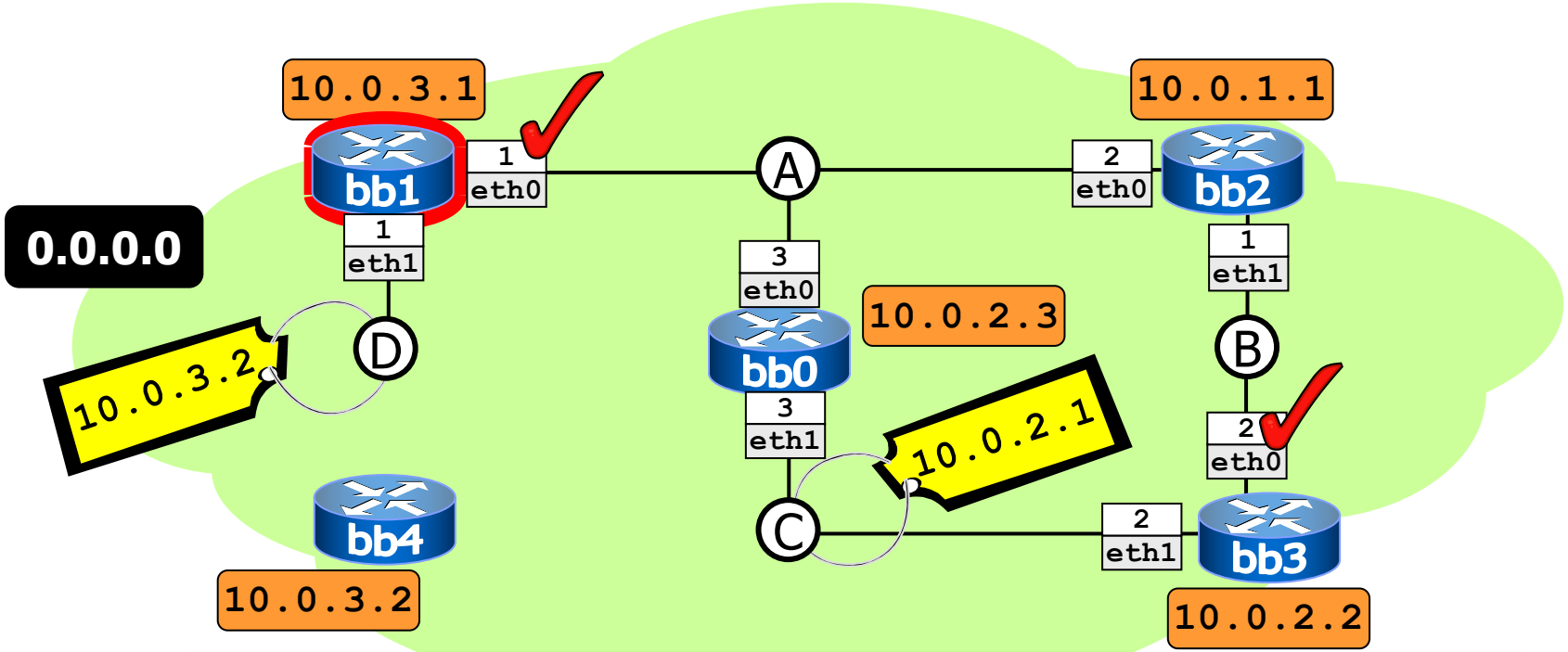
```
Link State ID: 10.0.3.1
  Number of Links: 2
   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.0.1
    (Link Data) Router Interface address: 10.0.0.1
   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.3.2
    (Link Data) Router Interface address: 10.0.3.1
```

note: the output of **`show ip ospf database router`** has been summarized

last update: Oct 2021



ospf's view of the network

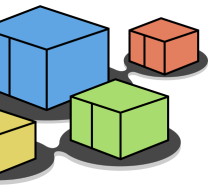


bb0

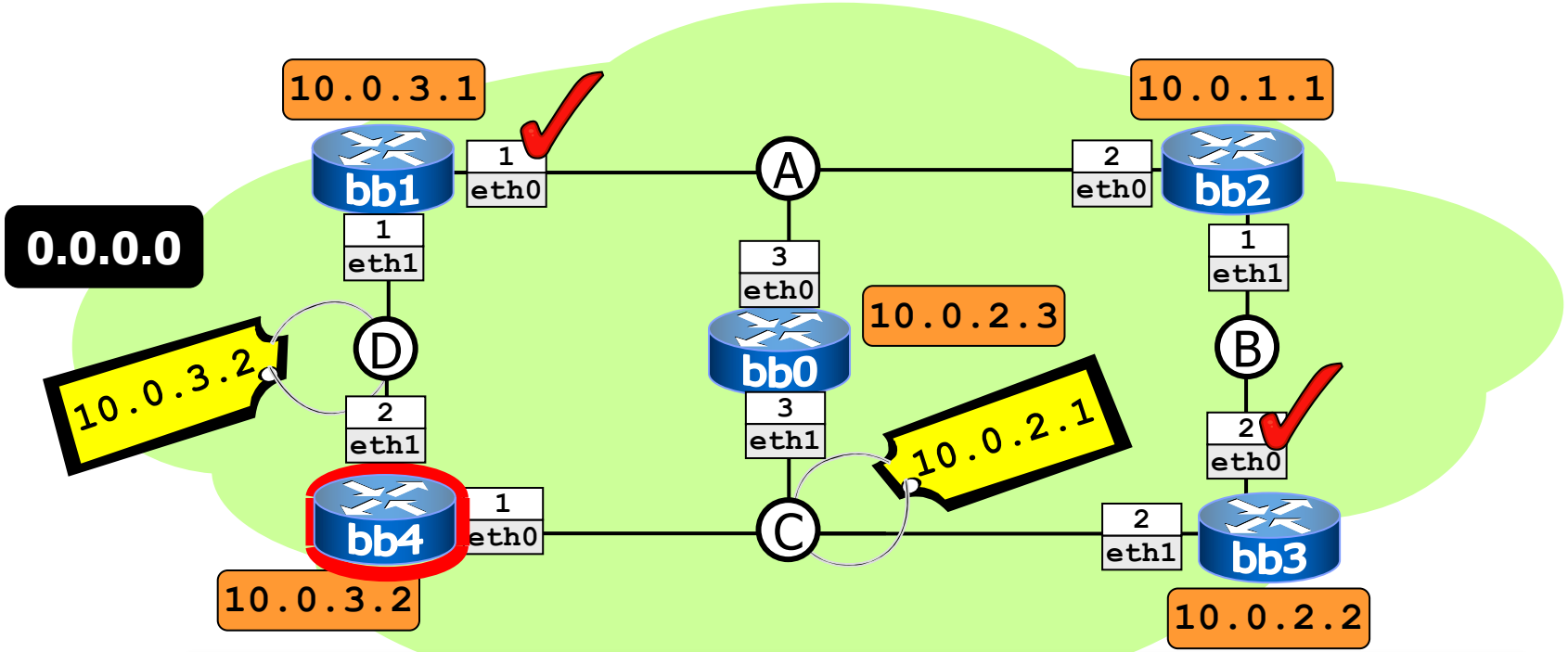
```
Link State ID: 10.0.3.1
  Number of Links: 2
   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.0.1
    (Link Data) Router Interface address: 10.0.0.1
   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.3.2
    (Link Data) Router Interface address: 10.0.3.1
```

note: the output of **`show ip ospf database router`** has been summarized

last update: Oct 2021



ospf's view of the network

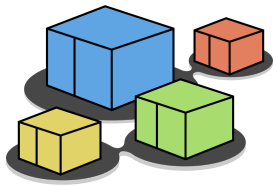


bb0

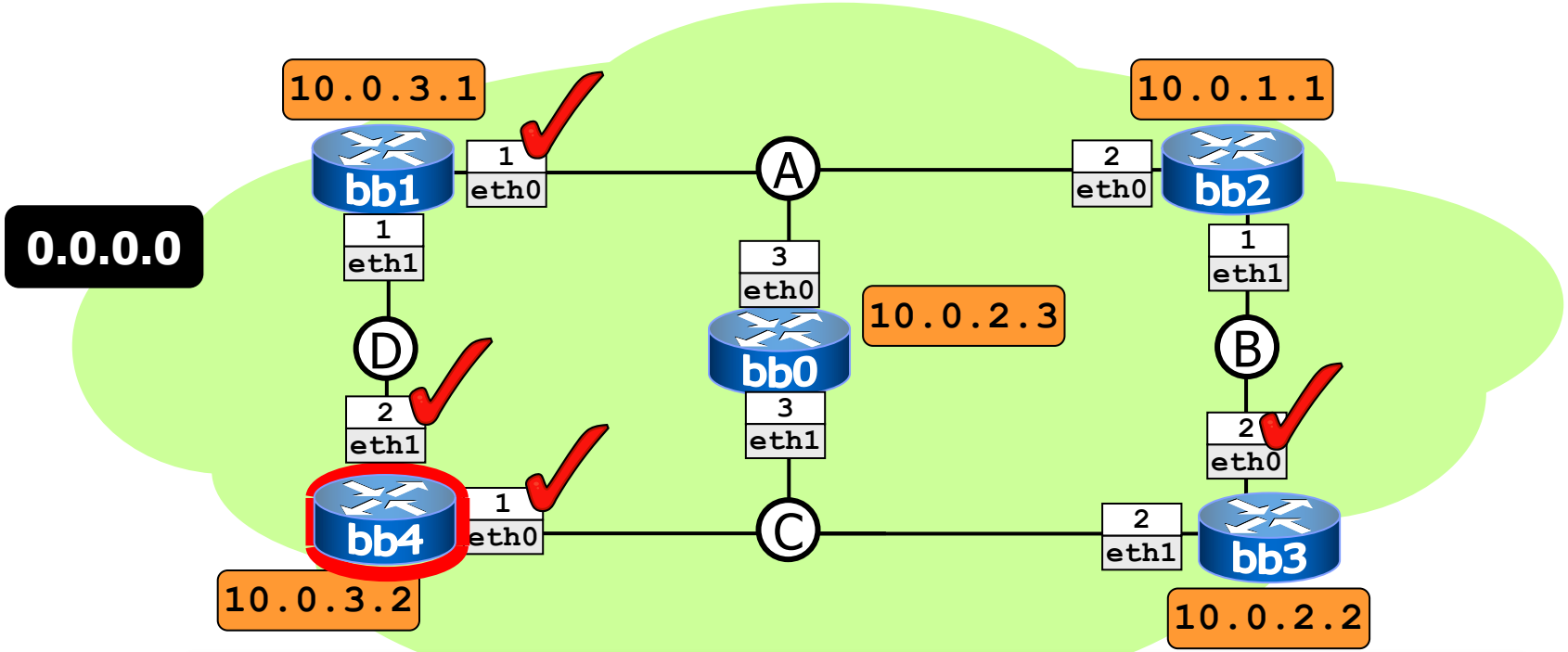
```
Link State ID: 10.0.3.2
  Number of Links: 2
   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.2.1
    (Link Data) Router Interface address: 10.0.2.1
   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.3.2
    (Link Data) Router Interface address: 10.0.3.2
```

note: the output of **`show ip ospf database router`** has been summarized

last update: Oct 2021



ospf's view of the network



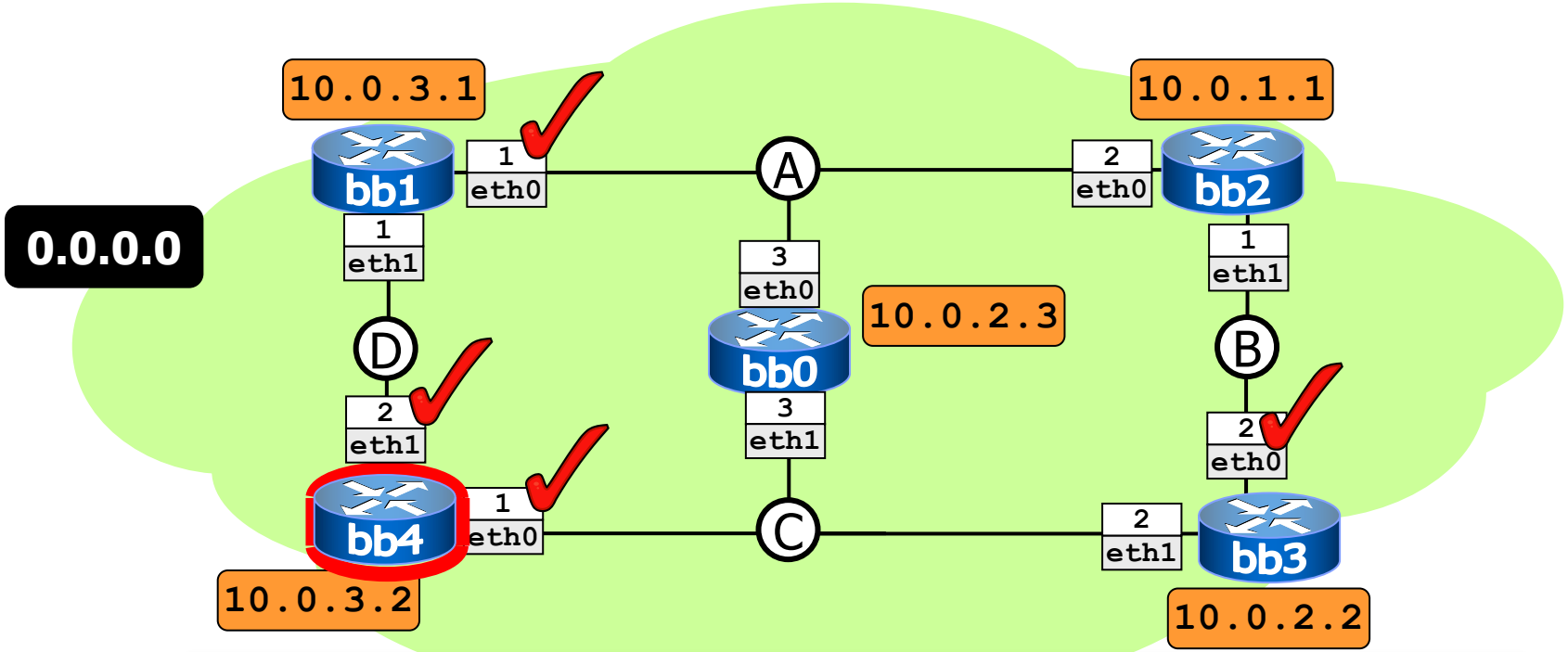
bb0

```
Link State ID: 10.0.3.2
  Number of Links: 2
   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.2.1
    (Link Data) Router Interface address: 10.0.2.1
   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.3.2
    (Link Data) Router Interface address: 10.0.3.2
```

note: the output of **`show ip ospf database router`** has been summarized

last update: Oct 2021

ospf's view of the network

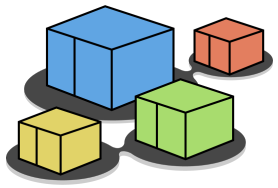


bb0

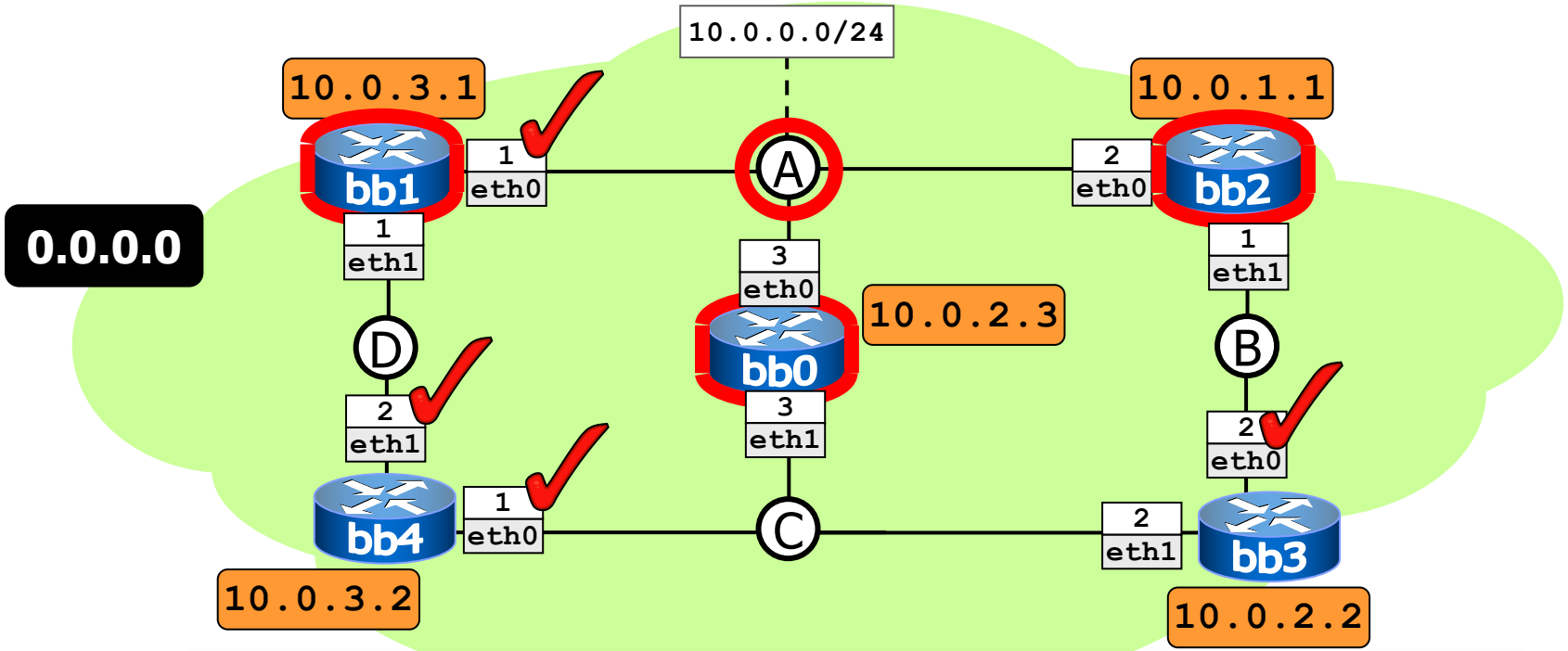
```
bb0# show ip ospf database network
```

note: the output of **`show ip ospf database network`** has been summarized

last update: Oct 2021



ospf's view of the network

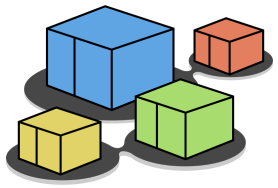


bb0

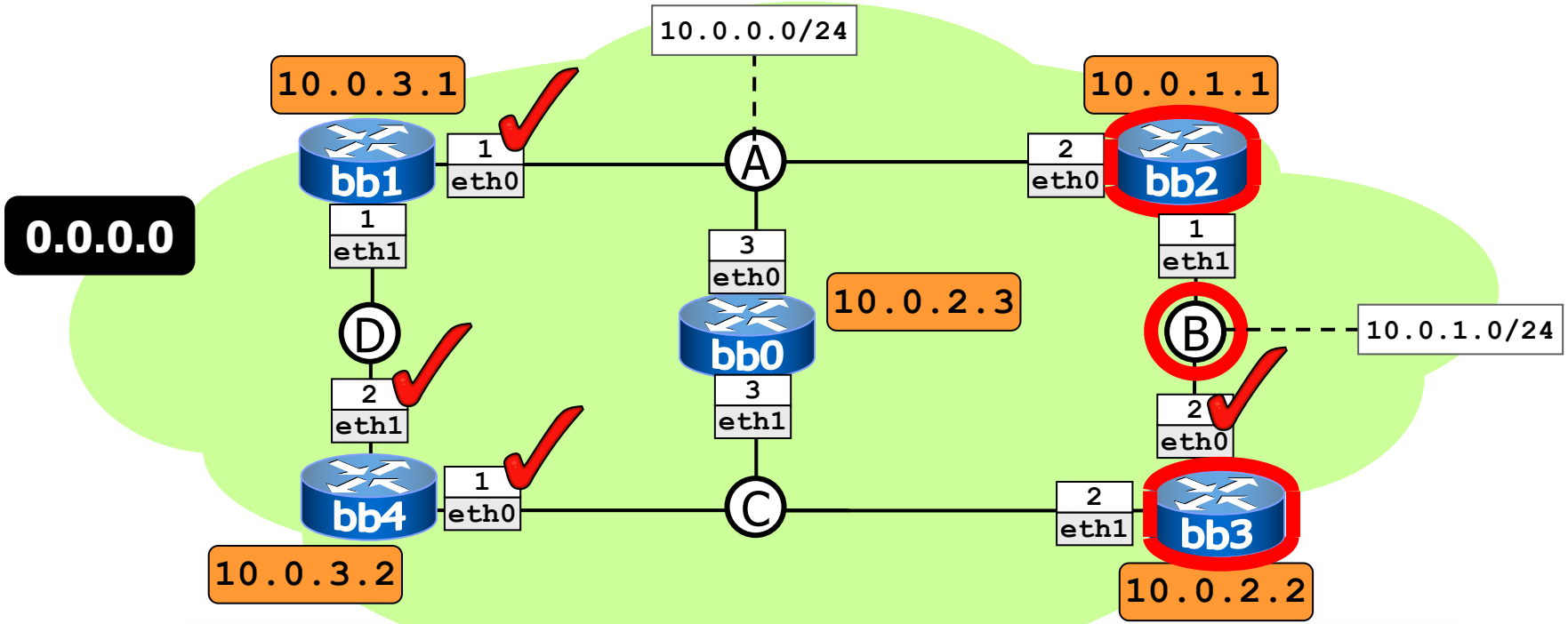
```
Link State ID: 10.0.0.1 (address of Designated Router)
 Advertising Router: 10.0.3.1
 Network Mask: /24
       Attached Router: 10.0.3.1
       Attached Router: 10.0.1.1
       Attached Router: 10.0.2.3
```

note: the output of **`show ip ospf database network`** has been summarized

last update: Oct 2021



ospf's view of the network

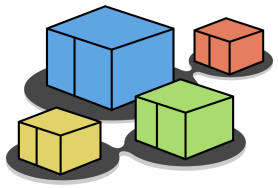


bb0

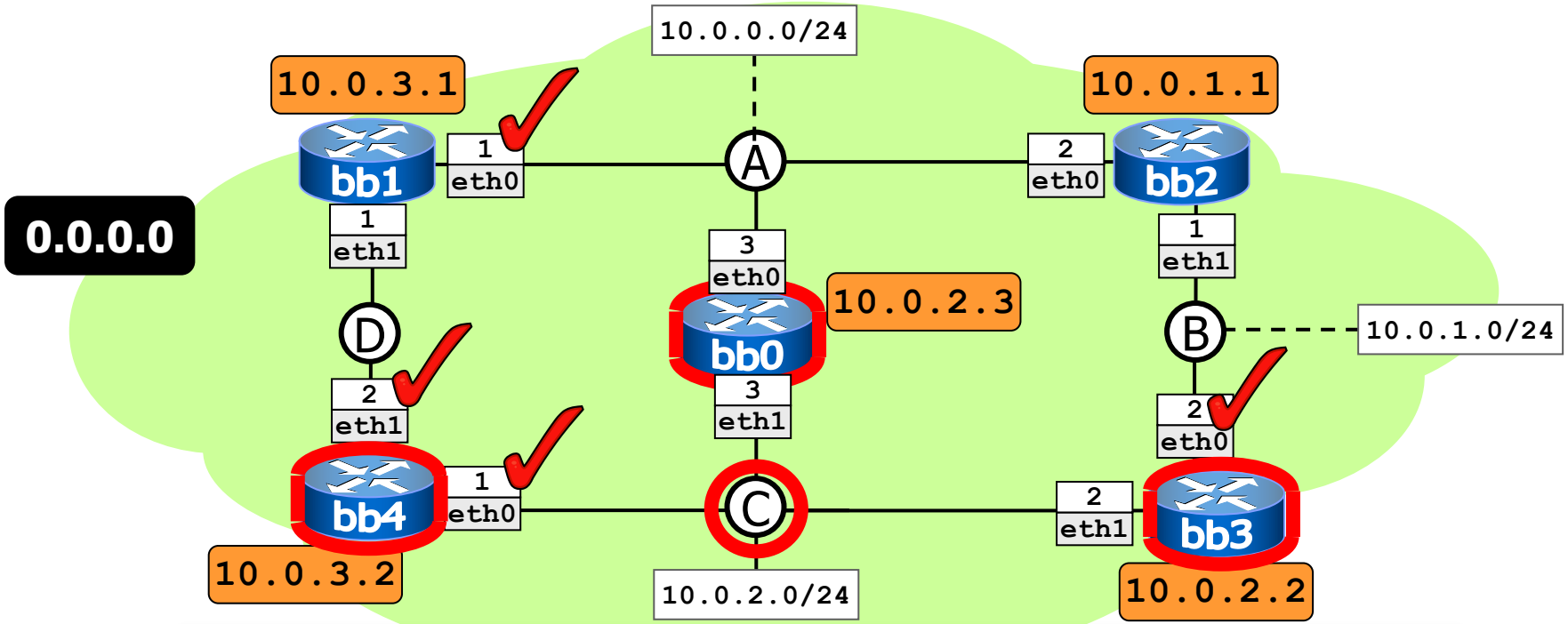
```
Link State ID: 10.0.1.2 (address of Designated Router)
 Advertising Router: 10.0.2.2
 Network Mask: /24
       Attached Router: 10.0.1.1
       Attached Router: 10.0.2.2
```

note: the output of **`show ip ospf database network`** has been summarized

last update: Oct 2021



ospf's view of the network

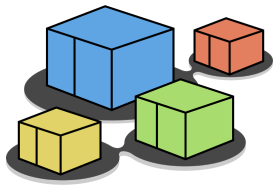


bb0

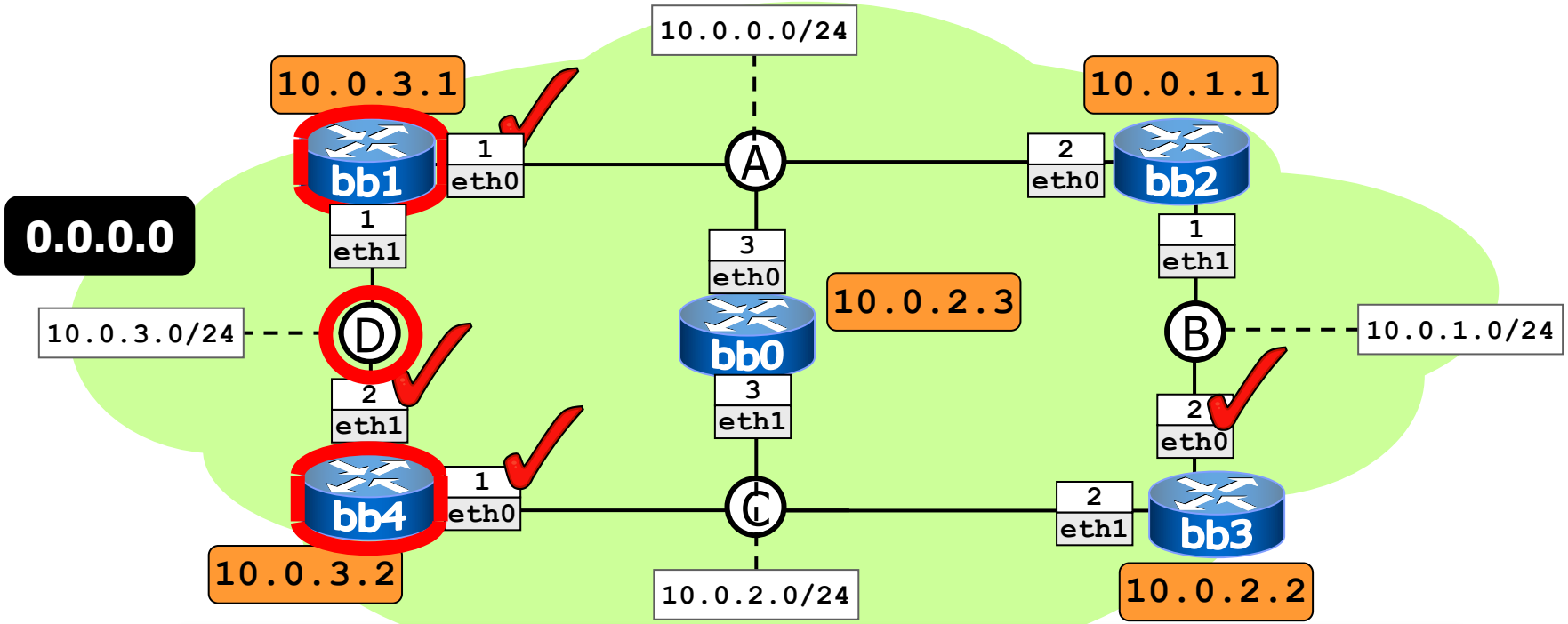
```
Link State ID: 10.0.2.1 (address of Designated Router)
 Advertising Router: 10.0.3.2
 Network Mask: /24
       Attached Router: 10.0.3.2
       Attached Router: 10.0.2.2
       Attached Router: 10.0.2.3
```

note: the output of **`show ip ospf database network`** has been summarized

last update: Oct 2021



ospf's view of the network

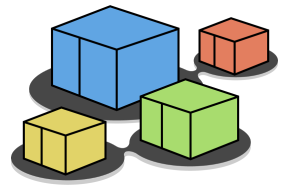


bb0

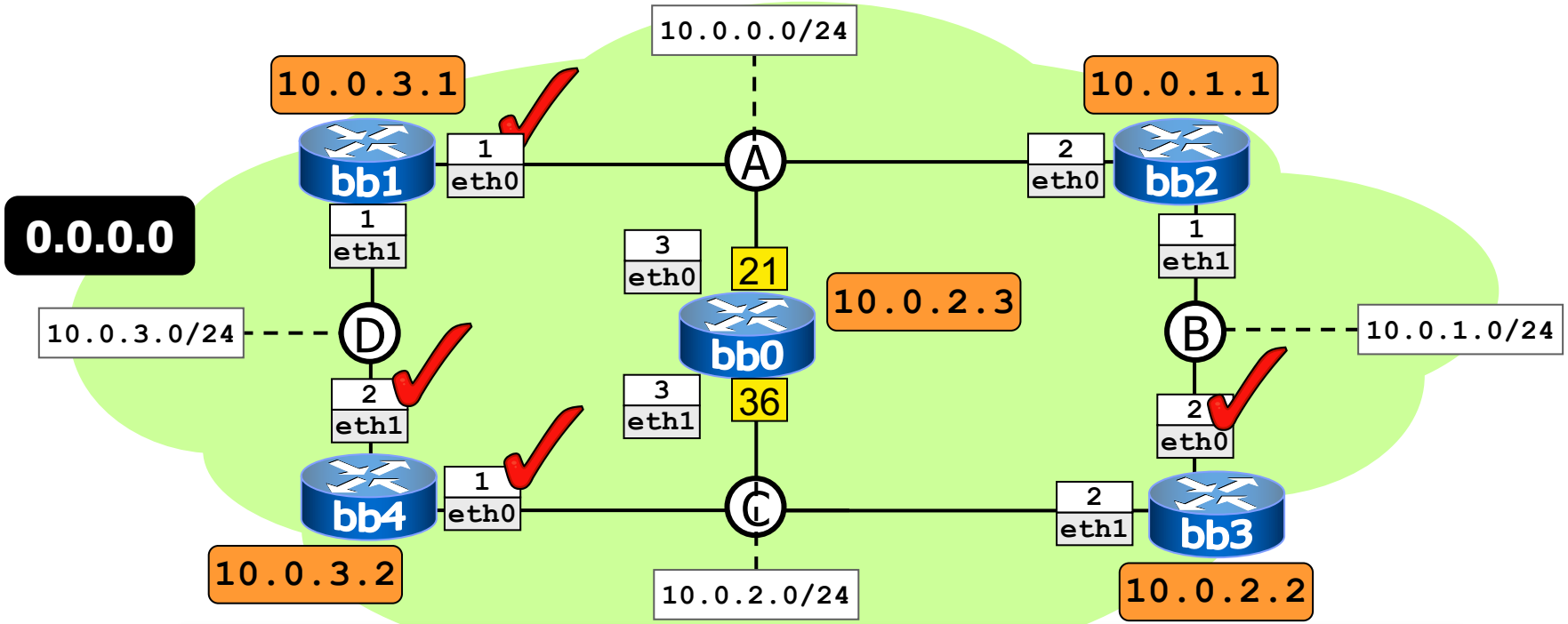
```
Link State ID: 10.0.3.2 (address of Designated Router)
Advertising Router: 10.0.3.2
Network Mask: /24
      Attached Router: 10.0.3.1
      Attached Router: 10.0.3.2
```

note: the output of **`show ip ospf database network`** has been summarized

last update: Oct 2021



ospf's view of the network



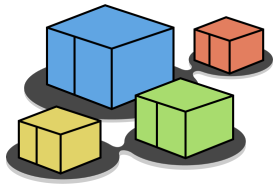
bb0

```
root@bb0:~# vtysh -e "show ip ospf interface" | egrep
"eth|Cost"
eth0 is up
  Router ID 10.0.2.3, Network Type BROADCAST, Cost: 21
eth1 is up
  Router ID 10.0.2.3, Network Type BROADCAST,
```

a shortcut to quickly get the cost

ospf interface costs can be queried on all routers

last update: Oct 2021



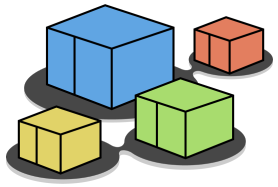
neighborhood

- router neighbors can be shown by using the **`show ip ospf neighbor`** command
- note: lsas are only sent between neighbors in **`Full`** state (i.e., capable of a bidirectional exchange of information); reaching the **`Full`** state requires that:
  - neighbors have been discovered (using hello packets)
  - bidirectional communication is possible
  - a designated router has been elected
- once reached, routers immediately synchronize their lsdbs

bb0

```
bb0# show ip ospf neighbor

    Neighbor ID Pri State           Dead Time Address     Interface         RXmtL RqstL DBsmL
10.0.3.1          1 Full/DR           30.462s 10.0.0.1    eth0:10.0.0.3         0     0     0
10.0.1.1          1 Full/DROther      30.462s 10.0.0.2    eth0:10.0.0.3         0     0     0
10.0.3.2          1 Full/DR           31.587s 10.0.2.1    eth1:10.0.2.3         0     0     0
10.0.2.2          1 Full/DROther      31.586s 10.0.2.2    eth1:10.0.2.3         0     0     0
bb0-frr#
```

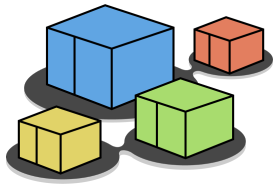


ospf routing table

- To dump the ospf routing table use **`show ip ospf route`**

bb0

```
bb0-frr# show ip ospf route
============ OSPF network routing table ============
N    10.0.0.0/24           [21] area: 0.0.0.0
                           directly attached to eth0
N    10.0.1.0/24           [31] area: 0.0.0.0
                           via 10.0.0.2, eth0
N    10.0.2.0/24           [36] area: 0.0.0.0
                           directly attached to eth1
N    10.0.3.0/24           [46] area: 0.0.0.0
                           via 10.0.2.1, eth1
============ OSPF router routing table =============
R    10.0.1.1              [21] area: 0.0.0.0, ASBR
                           via 10.0.0.2, eth0
R    10.0.2.2              [31] area: 0.0.0.0, ASBR
                           via 10.0.0.2, eth0
R    10.0.3.1              [21] area: 0.0.0.0, ASBR
                           via 10.0.0.1, eth0
R    10.0.3.2              [36] area: 0.0.0.0, ASBR
                           via 10.0.2.1, eth1
============ OSPF external routing table ===========
```



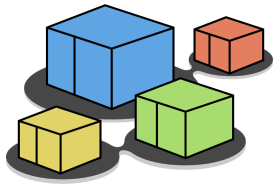
ospf routing table

- To dump the ospf routing table use **`show ip ospf route`**

bb0

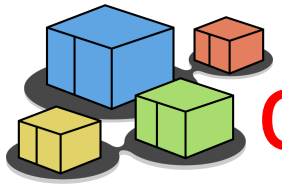
```
bb0# show ip ospf route
============ OSPF network routing table ============
N    10.0.0.0/24             [21] area: 0.0.0.0
                             directly attached to eth0
N    10.0.1.0/24             [31] area: 0.0.0.0
                             via 10.0.0.2, eth0
N    10.0.2.0/24             [36] area: 0.0.0.0
                             directly attached to eth1
N    10.0.3.0/24             [46] area: 0.0.0.0
                             via 10.0.2.1, eth1
============ OSPF router routing table =============
R    10.0.1.1                [21] area: 0.0.0.0, ASBR
                             via 10.0.0.2, eth0
     10.0.2.2                [31] area: 0.0.0.0, ASBR
                             via 10.0.0.2, eth0
     10.0.3.1                [21] area: 0.0.0.0, ASBR
                             via 10.0.0.1, eth0
R    10.0.3.2                [36] area: 0.0.0.0, ASBR
                             via 10.0.2.1, eth1
============ OSPF external routing table ===========
```

route cost



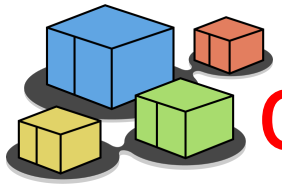
experiments

- issue the **`show ip ospf database`** and **`show ip ospf neighbor`** commands on different routers
- capture and look at exchanged ospf packets using **`tcpdump`**



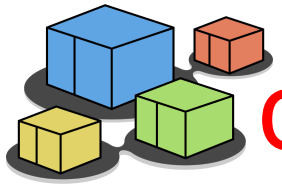
ospf is fast at detecting topology changes

- case #1: link fault
  - bring down a single network interface using **`ifconfig`**
    - the change is immediately propagated by the router inside lsa packets
    - routing tables are immediately updated (**`show ip ospf route`**)



ospf is fast at detecting topology changes

- case #1: link fault
  - bring down a single network interface using **`ifconfig`**
    - the change is immediately propagated by the router inside lsa packets
    - routing tables are immediately updated (**`show ip ospf route`**)
    - the lsdb is handled a little differently...



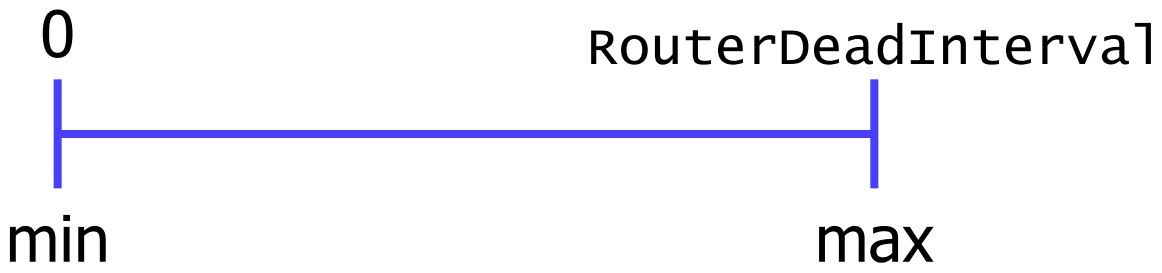
ospf is fast at detecting topology changes

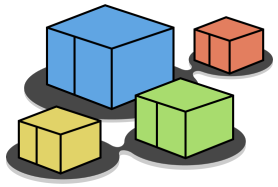
- case #1: link fault
  - bring down a single network interface using **`ifconfig`**
    - if this brings down a dr, the information is immediately flushed from the lsdb(s)...
      - ...and eventually reannounced when a dr is re-elected
    - otherwise, ospf waits expiry of the `RouterDeadInterval` timer (default: 40s) before removing the adjacency from the lsdb (**`show ip ospf database network`**)
      - note: networks that are connected to one router only, called stub networks, are only visible using **`show ip ospf database router`**

ospf is fast at detecting topology changes

- case #1: link fault
  - bring down a single network interface using **`ifconfig`**

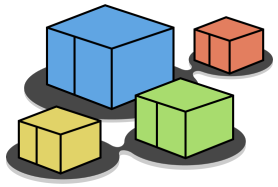
overall reaction time (estimated)





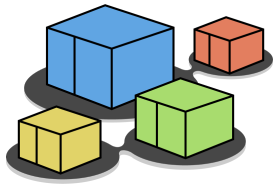
ospf is (often) fast at detecting topology changes

- case #2: router fault
  - bring down a router (by crashing it or by shutting down all its interfaces simultaneously)
  - the router has no chance to propagate lsas
    - the change *cannot* be immediately propagated
    - neighboring routers can only realize it (and update routing tables) after expiry of the `RouterDeadInterval` timer



ospf is (often) fast at detecting topology changes

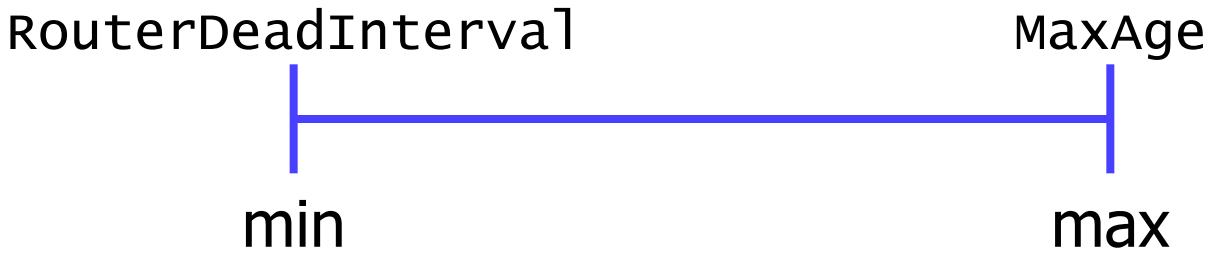
- case #2: router fault
  - bring down a router (by crashing it or by shutting down all its interfaces simultaneously)
  - after the change has been propagated...
    - ...lsdb information about networks for which the failed router was not dr is immediately flushed from other routers' lsdbs
      - the dr takes care of sending appropriate lsas
    - ...lsdb information about networks for which the failed router was dr (including those where a dr will be re-elected) and about routers is more "tough"
      - ospf waits for the lsa to expire (expiration happens when the age of the lsa reaches the `MaxAge` value of 1 hour) before taking any actions

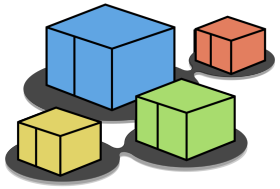


ospf is (often) fast at detecting topology changes

- case #2: router fault
  - bring down a router (by crashing it or by shutting down all its interfaces simultaneously)

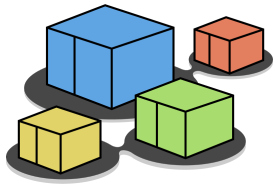
overall reaction time (estimated)





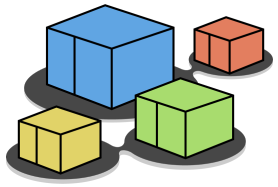
Kathará

lab: ospf-multiarea



ospf areas

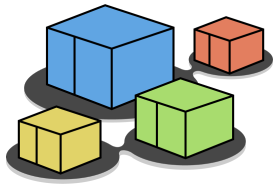
- an abstraction that simplifies administration and improves scalability
  - the topology of an area is invisible from the outside
  - routers internal to a given area don’t see the detailed external topology
- each area runs a separate instance of the link state routing algorithm
  - all routers in an area construct the same lsdb
  - each router keeps a distinct lsdb for each area it belongs to



ospf areas

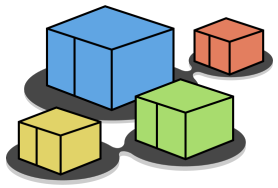
- identified by a 32-bit number,
  often in dotted decimal notation (1.2.3.4)
  - different interfaces of the same router can be assigned to different areas
  - each
    - router interface...
    - network...
    - router adjacency...

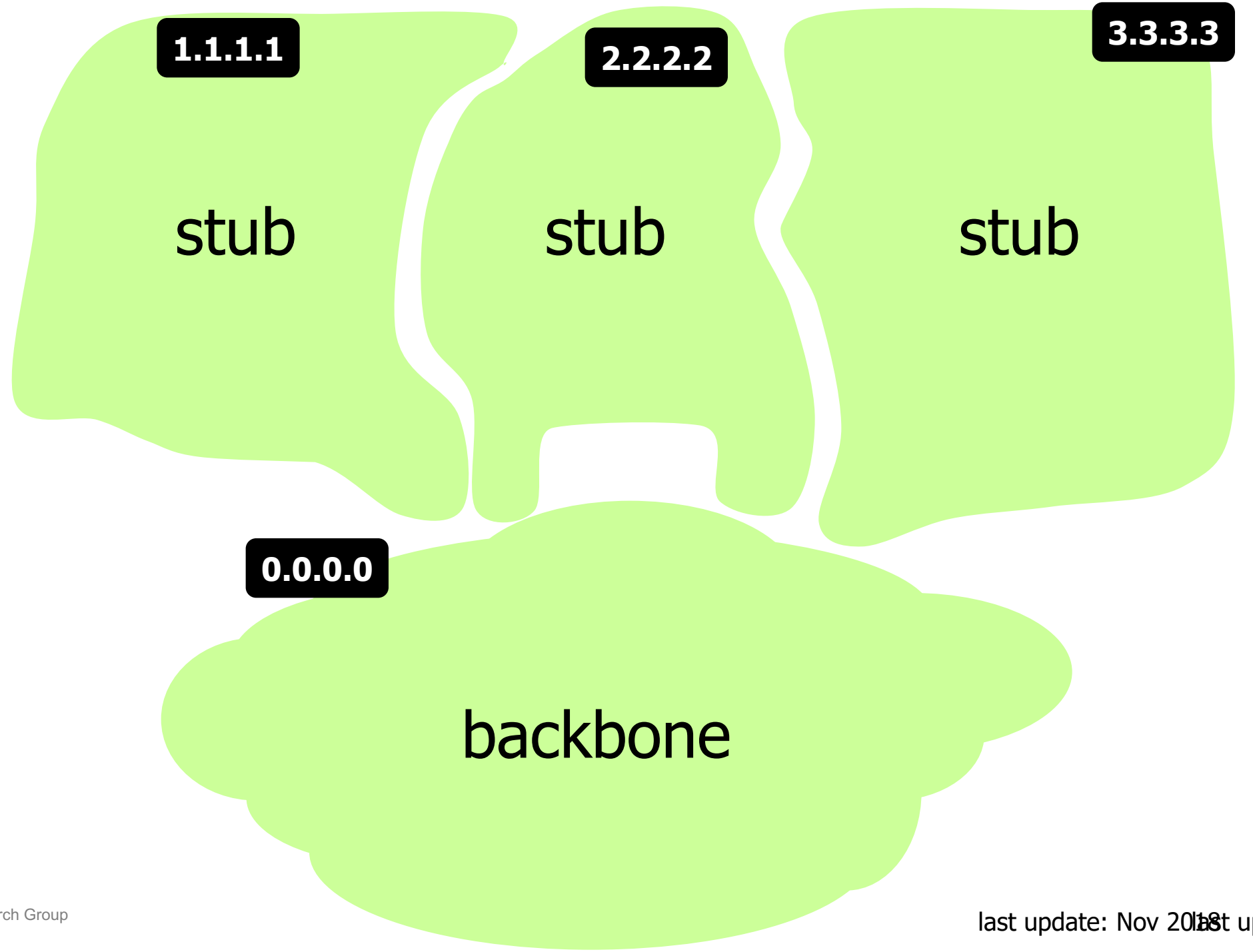
    ...is associated with a single area

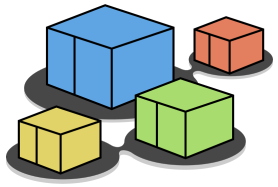


area types

- backbone (0.0.0.0)
  - must be (virtually) connected
  - all other areas are connected to it
  - contains all the area border routers
- stub
  - does not receive advertisements of external routes
  - internal nodes are offered a default route
  - cannot contain autonomous system boundary routers
  - the backbone can’t be a stub area
- transit
  - used to pass traffic from one adjacent area to another, via virtual links

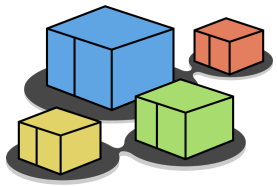


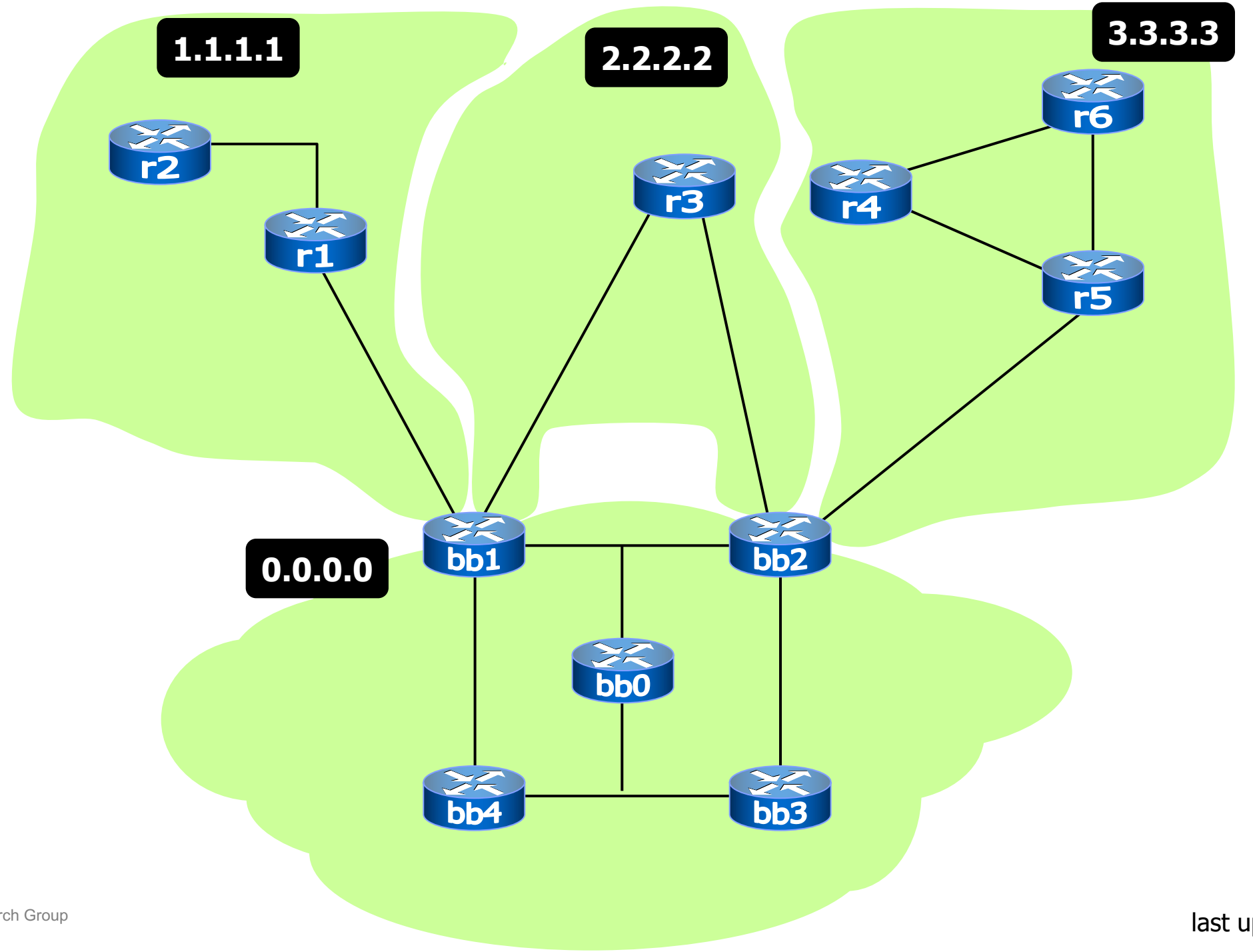




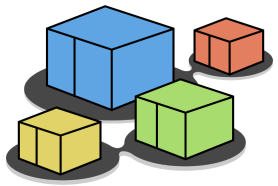
router types

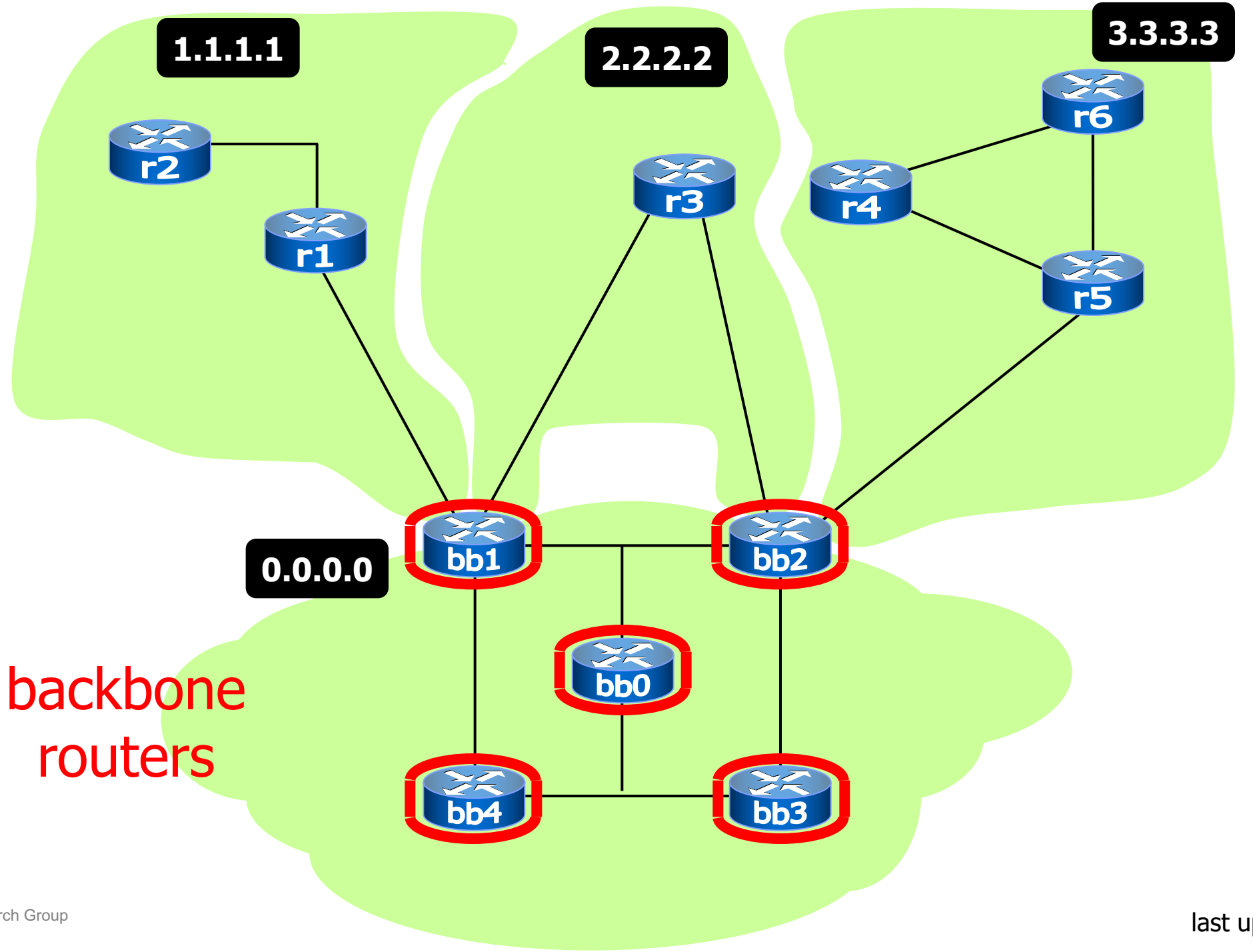
- internal router
  - all interfaces belong to the same area
- area border router (abr)
  - connects one or more areas to the backbone
  - keeps multiple lsdbs, one for each area
- backbone router
  - has at least one interface connected to the backbone
  - an abr is always a backbone router
- autonomous system boundary router (asbr)
  - imports and floods routing information from other routing protocols (typically, bgp)
- note: a router can be of more than one type



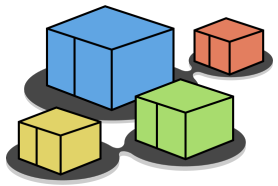


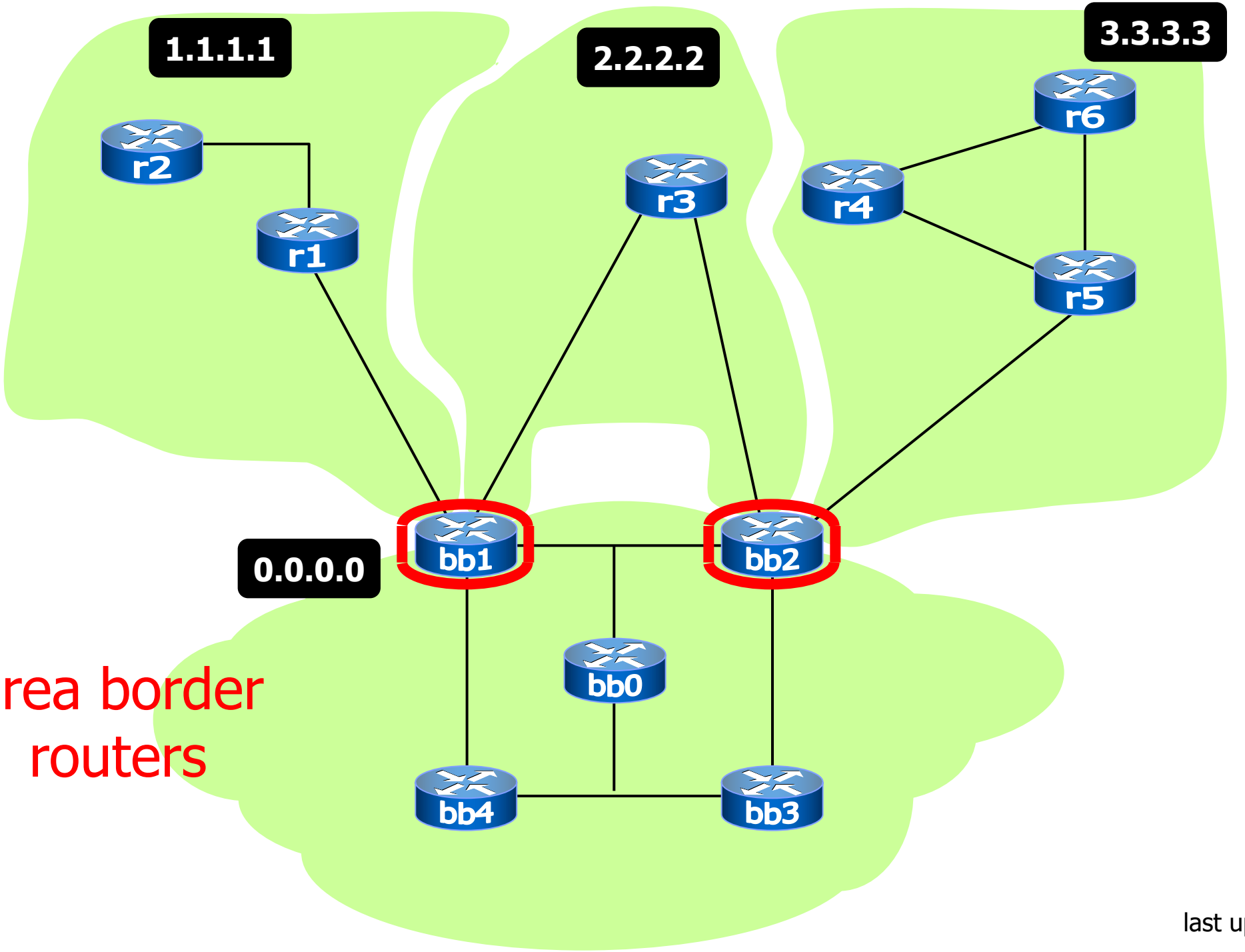
last update: Oct 2021





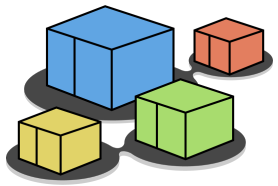
last update: Oct 2021

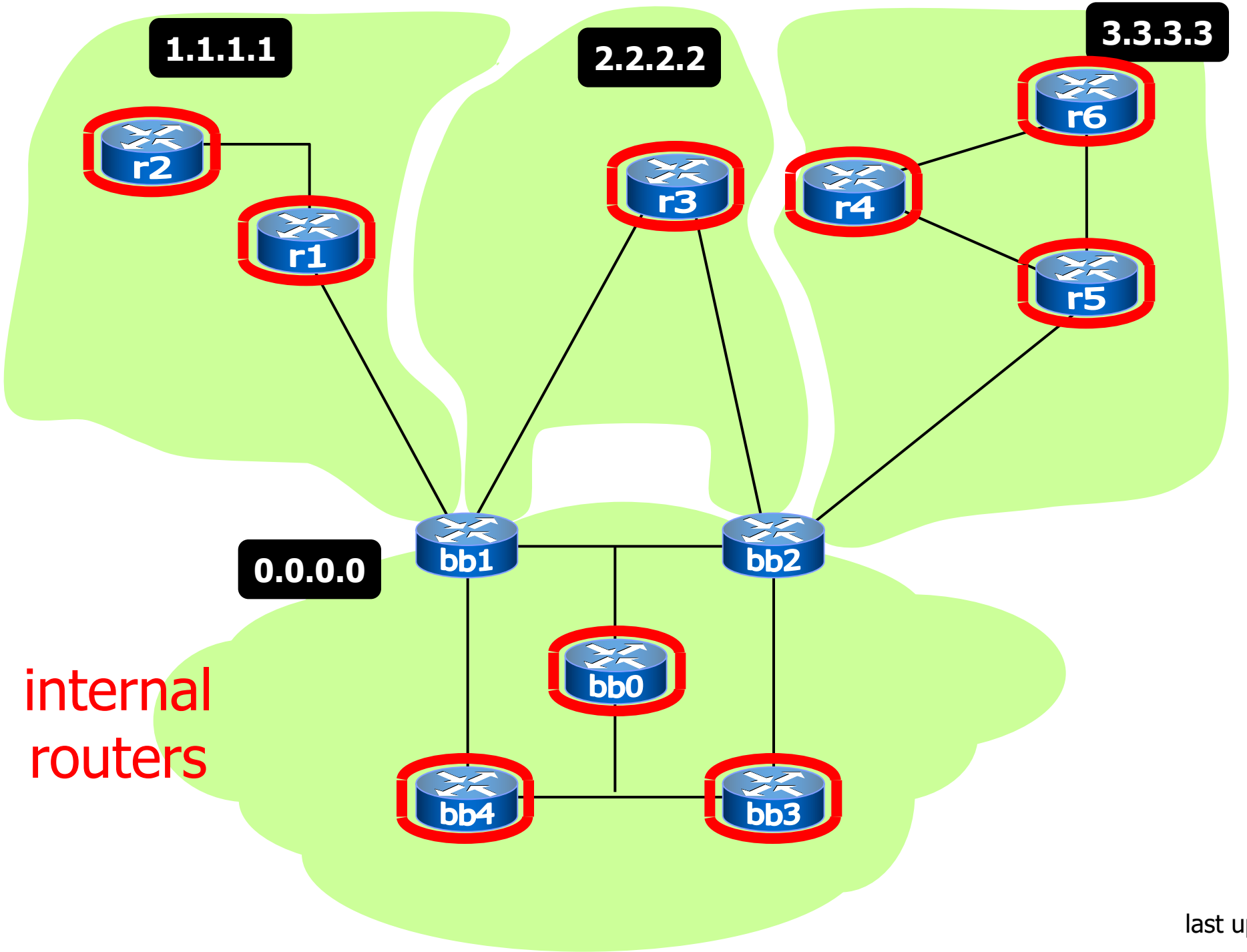


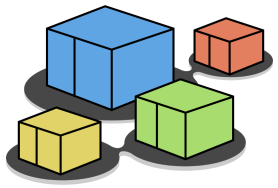


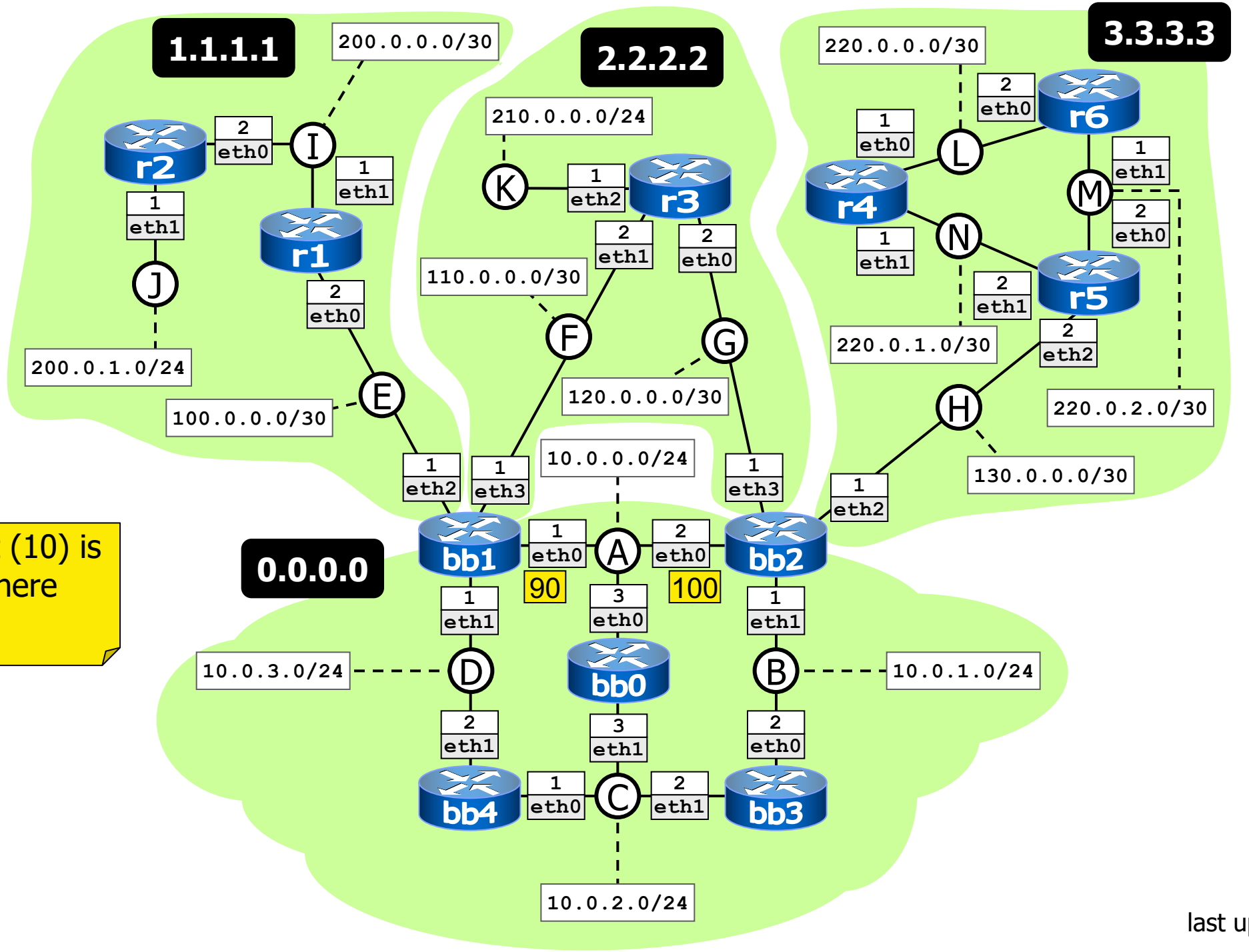
area border routers

last update: Oct 2021



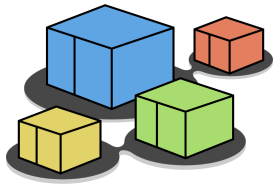






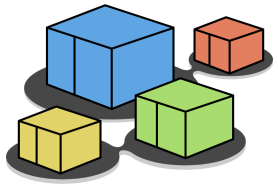
default cost (10) is assumed where unspecified

last update: Oct 2021



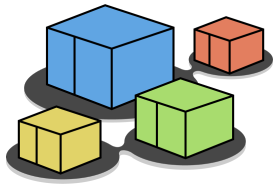
area configuration

- area information is found in two places
  - when enabling ospf on router interfaces
    `network 200.0.0.0/16 area 1.1.1.1`
  - when specifying the area type (not required for the backbone)
    `area 1.1.1.1 stub`



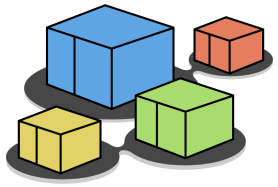
ospf path types

- there are 4 path types
  1. intra-area
  2. inter-area
  3. external type 1
  4. external type 2
- types can coexist in the same network
- each type is preferred over the following ones



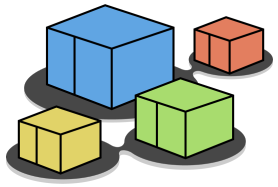
ospf path types

- intra-area paths
  - calculated using the shortest-path tree



ospf path types

- inter-area paths
  - abrs inject summary information inside each area, to make it aware of available destinations in other areas
    - such information includes the cost of the shortest path from the abr to the destination
    - if multiple subnets are summarized into a single network, the route cost will be the maximum cost to any of the component subnets
  - an inter-area path is always composed of:
    - an intra-area path from the source to the abr
    - a backbone path between the source and destination areas
    - an intra-area path to the destination



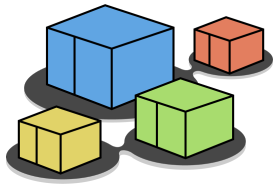
ospf path types

- external paths are learned from other routing protocols (e.g., bgp)
- type 1: the cost is expressed in terms of
  - the external (bgp) route cost* +
  - the ospf cost to the asbr

  example with bgp cost=495, ospf cost=10:

```
N E1 50.0.0.0/16        [505] tag: 0
                        via 10.0.1.2, eth1
```

* cost used when redistributing the protocol (bgp) into ospf; default for bgp=20; configurable by using `redistribute bgp metric` *`value`*

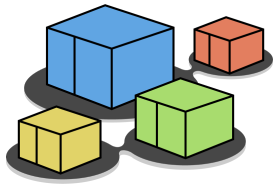


ospf path types

- external paths are learned from other routing protocols (e.g., bgp)
- type 1: the cost is expressed in terms of
  - the external (bgp) route cost* +
  - the ospf cost to the asbr
- type 2: the cost is expressed in terms of
  - the external (bgp) route cost* only (distance to the asbr is only used to break ties)

example with bgp cost=495, ospf cost=10:

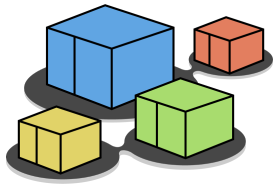
```
N E2 50.0.0.0/16      [10/495] tag: 0
                      via 10.0.1.2, eth1
```



ospf path types

- external paths are learned from other routing protocols (e.g., bgp)
- type 1: the cost is expressed in terms of
  - the external (bgp) route cost* +
  - the ospf cost to the asbr
- type 2: the cost is expressed in terms of
  - the external (bgp) route cost* only (distance to the asbr is only used to break ties)
- metric type is user-configurable

```
redistribute bgp metric-type 2 metric 495
```



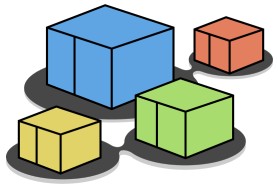
experiments

- check that routers know detailed topology information only about their own area

r2

```
r2# show ip ospf neighbor

Neighbor ID     Pri State           Up Time    Dead Time Address  Interface      RXmtL RqstL DBsmL
200.0.0.1         1 Full/Backup     1m17s      32.279s 200.0.0.1  eth0:200.0.0.2     0     0     0
```



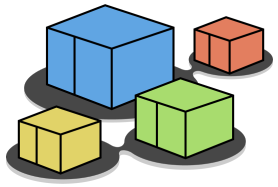
experiments

- check that routers know detailed topology information only about their own area

r2

```
r2# show ip ospf database router
 OSPF Router with ID (200.0.1.1)
                Router Link States (Area 1.1.1.1 [Stub])
  ...
  Link State ID: 110.0.0.1
  ...
   Number of Links: 1
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 100.0.0.2
     (Link Data) Router Interface address: 100.0.0.1
    ...
  Link State ID: 200.0.0.1
  ...
   Number of Links: 2
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 100.0.0.2
     (Link Data) Router Interface address: 100.0.0.2
     ...
    Link connected to: a Transit Network
     (Link ID) Designated Router address: 200.0.0.2
     (Link Data) Router Interface address: 200.0.0.1
     ...
```

note: the output has been summarized



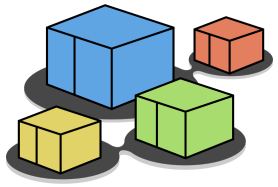
experiments

- check that routers know detailed topology information only about their own area

r2

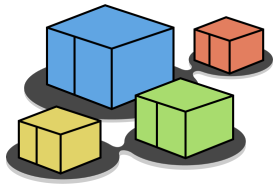
```
r2# show ip ospf database network
       OSPF Router with ID (200.0.1.1)
                Net Link States (Area 1.1.1.1 [Stub])
  LS age: 759
  Options: 0x0  : *|-|-|-|-|-|-|-
  LS Flags: 0x6
  LS Type: network-LSA
  Link State ID: 100.0.0.2 (address of Designated Router)
  Advertising Router: 200.0.0.1
  LS Seq Number: 80000001
  Checksum: 0x09ec
  Length: 32
  Network Mask: /30
        Attached Router: 110.0.0.1
        Attached Router: 200.0.0.1
  LS age: 763
  Options: 0x0  : *|-|-|-|-|-|-|-
  LS Flags: 0x3
  LS Type: network-LSA
  Link State ID: 200.0.0.2 (address of Designated Router)
  Advertising Router: 200.0.1.1
  LS Seq Number: 80000001
  ...
```

note: the output has been summarized



experiments

- check what routers know about the outside of the area, using the **`show ip ospf database summary`** command
  - in particular, check the **`Metric`** values, that show how far away the destination is from the advertising abr
- check that routers in stub areas are offered a default route, whereas routers in the backbone are not
  - also check what **`Metric`** is assigned to the default route



experiments

- experiment ospf's recovery capabilities
  - when multiple equal cost routes are available, ospf keeps all of them
  - check it by verifying what **`r3`** knows about the default route

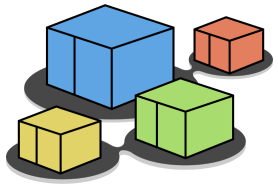
r3

```
r3# show ip ospf route
============ OSPF network routing table ============
N IA 0.0.0.0/0              [11] area: 2.2.2.2
                            via 110.0.0.1, eth1
                            via 120.0.0.1, eth0

...
```

inter-area path

equal cost routes



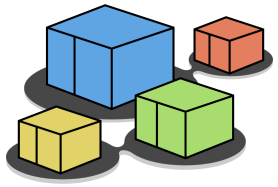
experiments

- experiment ospf's recovery capabilities
  - when multiple equal cost routes are available, ospf keeps all of them
  - check it by verifying what **`r3`** knows about the default route
  - zebra performs the actual selection

r3

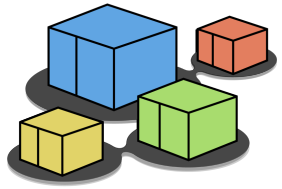
```
r3# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - IS-IS, B - BGP, E - EIGRP, N - NHRP, T - Table, v - VNC,
       V - VNC-Direct, A - Babel, F - PBR, f - OpenFabric, > - selected route,
       * - FIB route, q - queued, r - rejected, b - backup t - trapped, o - offload
failure

O>* 0.0.0.0/0 [110/11] via 110.0.0.1, eth1, weight 1, 00:18:15
  *                    via 120.0.0.1, eth0, weight 1, 00:18:15
```



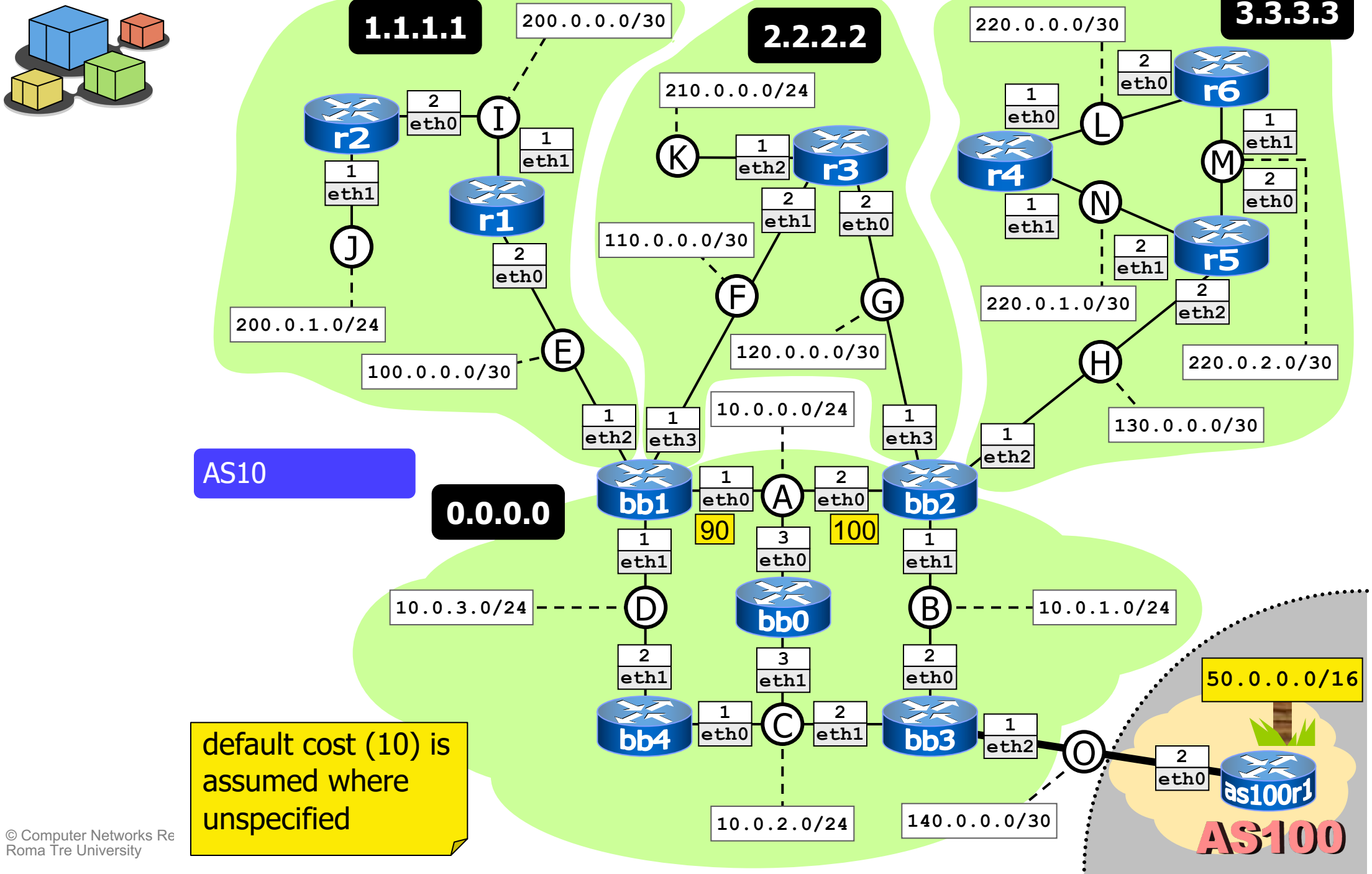
experiments

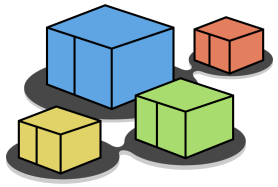
- experiment ospf's recovery capabilities
  - when multiple equal cost routes are available, ospf keeps all of them
  - check it by verifying what **`r3`** knows about the default route
  - zebra performs the actual selection
    - now bring **`bb1`**'s **`eth3`** down using **`ifconfig`**, wait a few seconds and check how the routing is changed
    - bring **`bb1`**'s **`eth3`** back up and check again how the routing is changed



Kathará

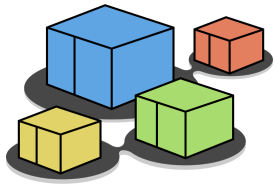
lab: ospf-complex





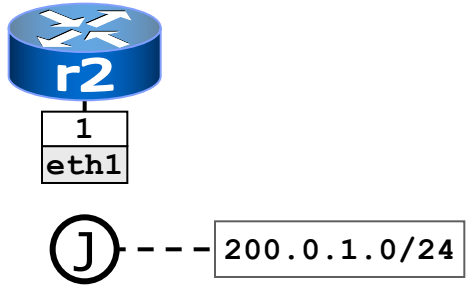
lab description

- same as multiarea + some information is injected via bgp from an external as
  - also, abrs are configured to just inject the default route
    `area 1.1.1.1 stub no-summary`
- perform the same experiments as for the multiarea lab
  - in addition, check asbr information using
    - **`show ip ospf database asbr-summary`**
  - also check that such information is not propagated inside stub areas



a quick note about stub networks

- "stub" = not used for transit
- three possible situations:



ospf is enabled also on the stub network's interface	then	the stub network is advertised in the entire ospf domain	but	interface uselessly sends hello packets
ospf is not enabled on the stub network's interface	then	the stub network is advertised	but	the stub network is advertised as an AS-external route (hence, only through the backbone)
ospf is enabled on the stub network's interface, which is configured as `passive-interface` ✓	then	the stub network is advertised in the entire ospf domain	but	N/A

last update: Oct 2021