EXPLICIT ROUTING for 802.1Qca

ciena

Background – recap of SPB SPF routing

SPB constructs Shortest Path source-rooted (multicast) trees :

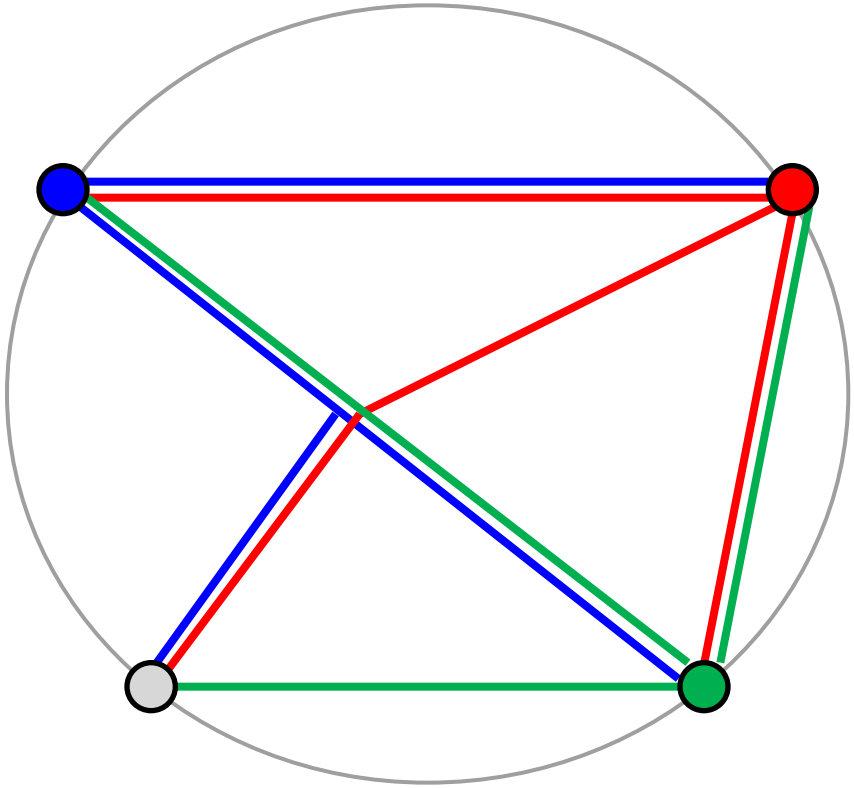
- → with a location-independent tie-breaker, go-return congruence "just happens" by construction
- → and destination-based forwarding follows that tree back to the root

Trees are identified by an SPVID (SPBV), or source MAC (SPBM) :

- → so SPBV uses **1** VID per edge node
- → and SPBM fully meshes the network on **1** VID per route set.

This analysis uses VID as a shorthand for "Tree" :

- → so any conclusions are valid both for FDBs populated by MAC learning or by ISIS-SPB.



ciena

Explicit Routes – first order consequences ?

The objective of 802.1Qca is to move traffic off the shortest path :

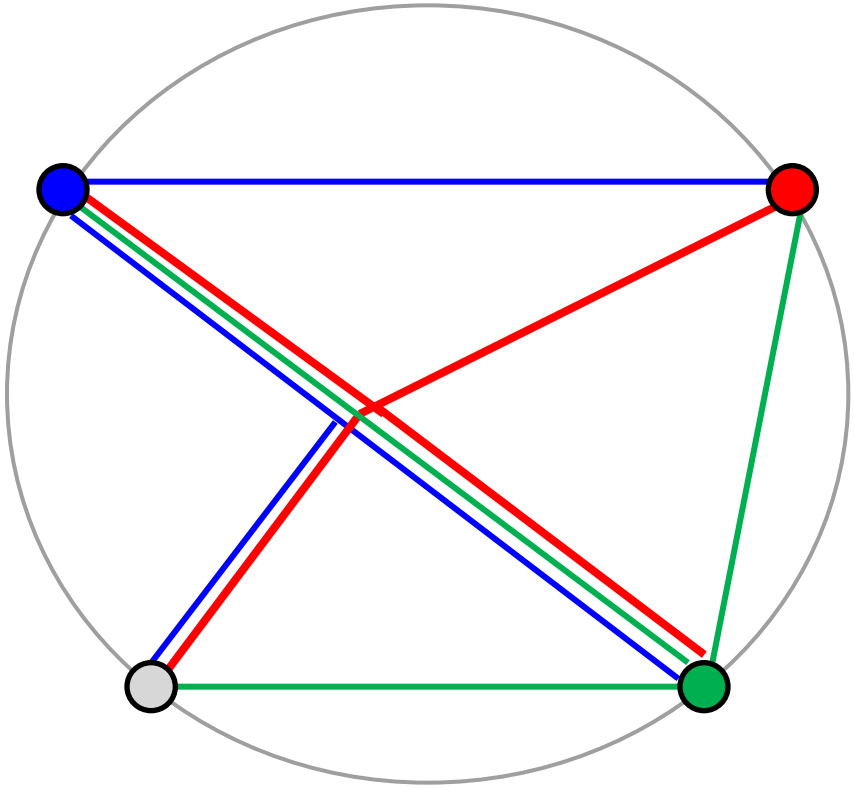
→ so how do we handle this ?

Moving to Explicit Routes means that go/return congruence is not an automatic consequence of the algorithm – it must be enforced by the PCE :

(the **red** tree ➔ has been rebuilt in this diagram)

→ look at the **blue**⇔**red** paths,

→ and at the **green**⇔**red** paths



ciena

Explicit Routes – first order consequences ?

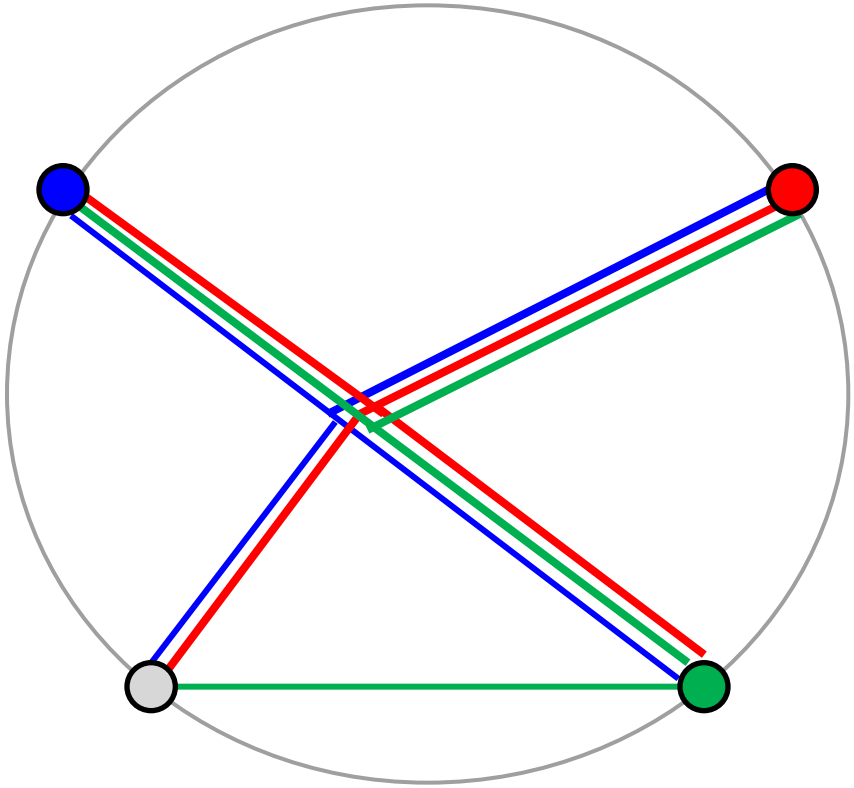
Moving to Explicit Routes means that congruence is not automatic

- it must be enforced by the PCE

Sometimes congruence can be restored by re-routing the same trees :

→ **blue** and **green** paths to **red**

But other routing choices can make this impossible …



ciena

Other routing choices make this impossible …

This is however a train-wreck :

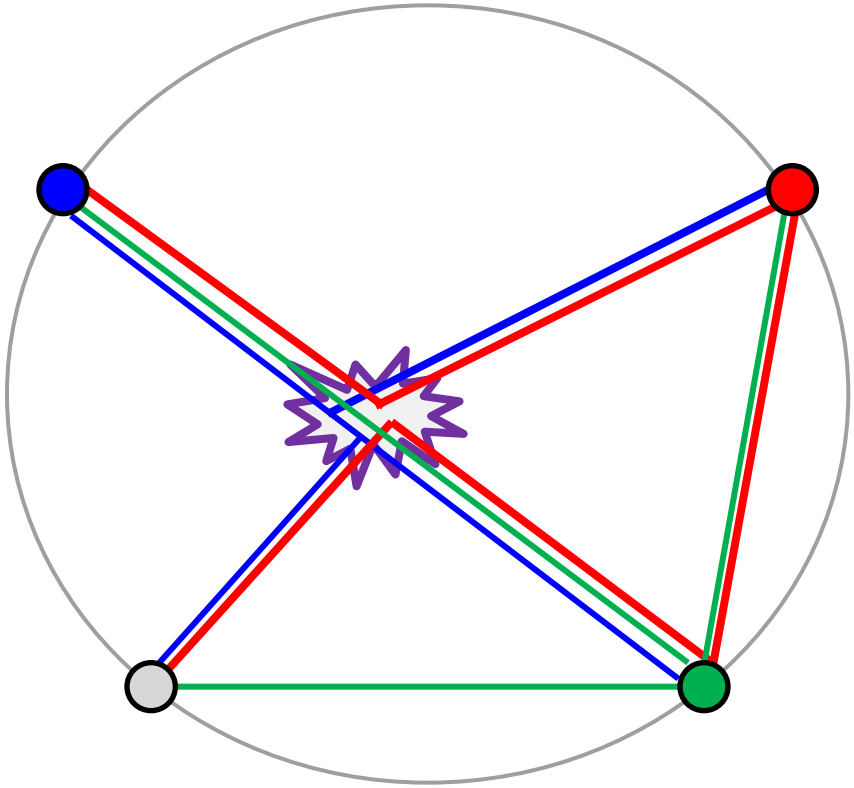
- → **two red paths intersect or cross**
- → **and cease to be simple trees.**

Viewed another way :

- → there are two conflicting routes (ports) by which unicast traffic should be forwarded towards the **red** node.

What rules can be formulated to avoid this problem :

- → **straightforwardly ?**
- → **incrementally** (to allow churn) ?



ciena

EXPLICIT ROUTING for 802.1Qca *(1)*

ciena

1st option : Bidirectional VIDs

This is really simple :

→ Off-SPF paths – every pair-wise path is explicitly routed, and assigned a single VID (⇔)

→ and this extends naturally to (unidirectional) multicast trees.

VIDs can be reused :

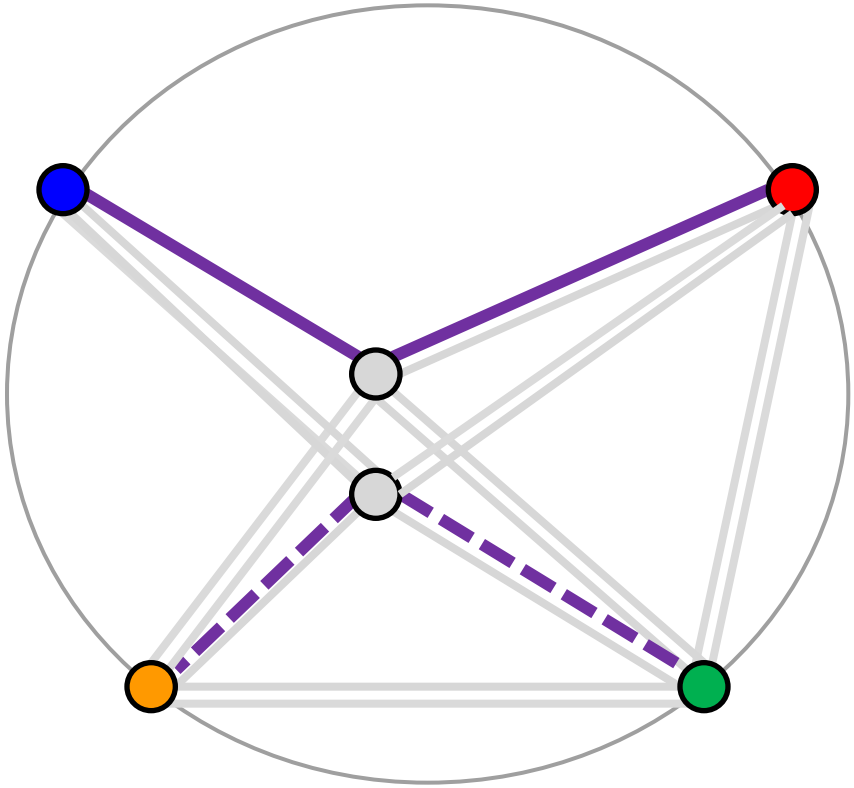
→ provided they never meet ➔

The only issue with this happy state is that the absolute limit of 4K VIDs :

→ a full mesh on only ~**90** nodes,

which may not scale adequately :

→ **so, what then ?**



ciena

EXPLICIT ROUTING for 802.1Qca *(2)*

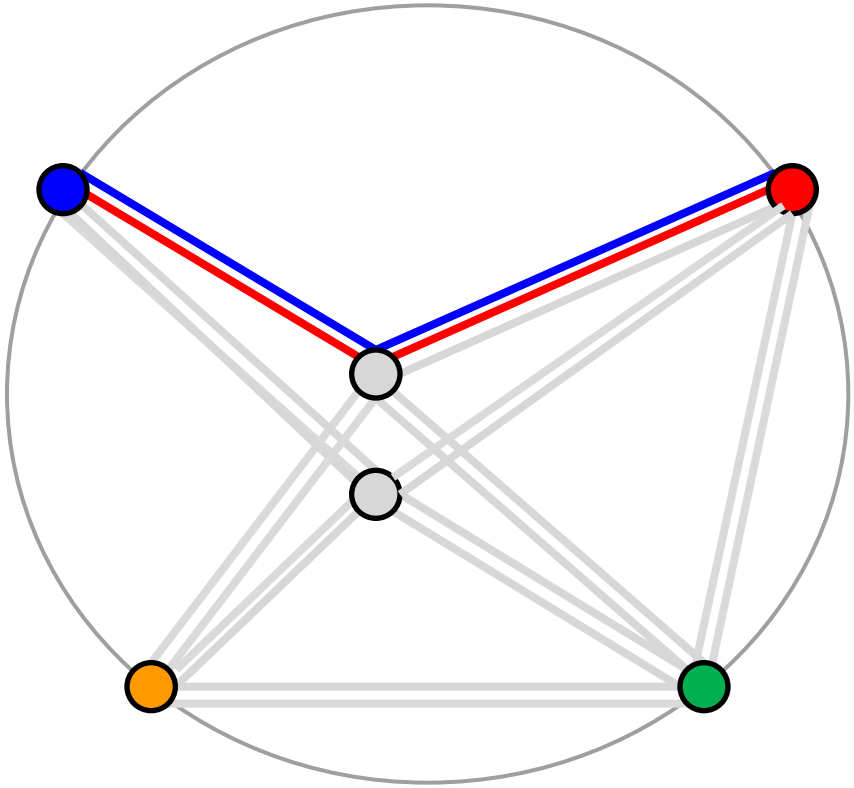
ciena

2nd option : Trees with unidirectional VIDs

This second approach <u>incrementally</u> builds off-SPF trees

- → **every pair-wise path must be co-routed for congruence, each root with its own VID**
  - → build the **red root** and **blue root** path elements,
  - → and that is the first p2p EP.

This directly follows SPBV practice, and is also very similar to unicast trunks running over PBB-TE.

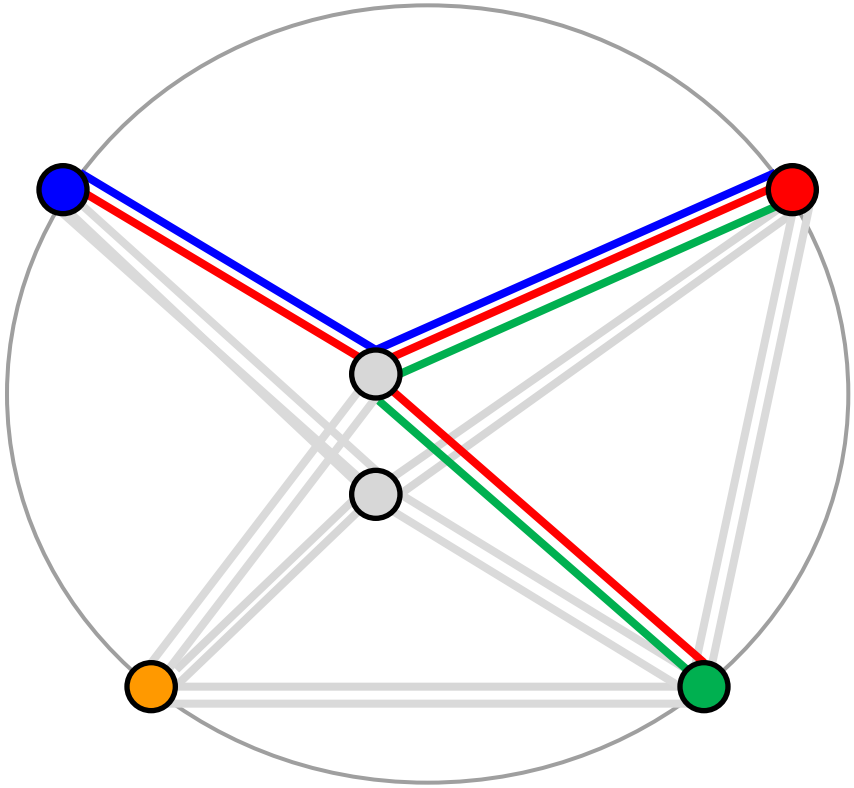


ciena

2nd option : Trees with unidirectional VIDs

This second approach incrementally builds off-SPF trees

- → **every pair-wise path must be co-routed for congruence, each root with its own VID**
  - → first build the red root and blue root path elements
  - → then extend an **existing** tree to add the **red**⇔**green** path
  - → which can reuse the previous **red root** VID for the return path



ciena

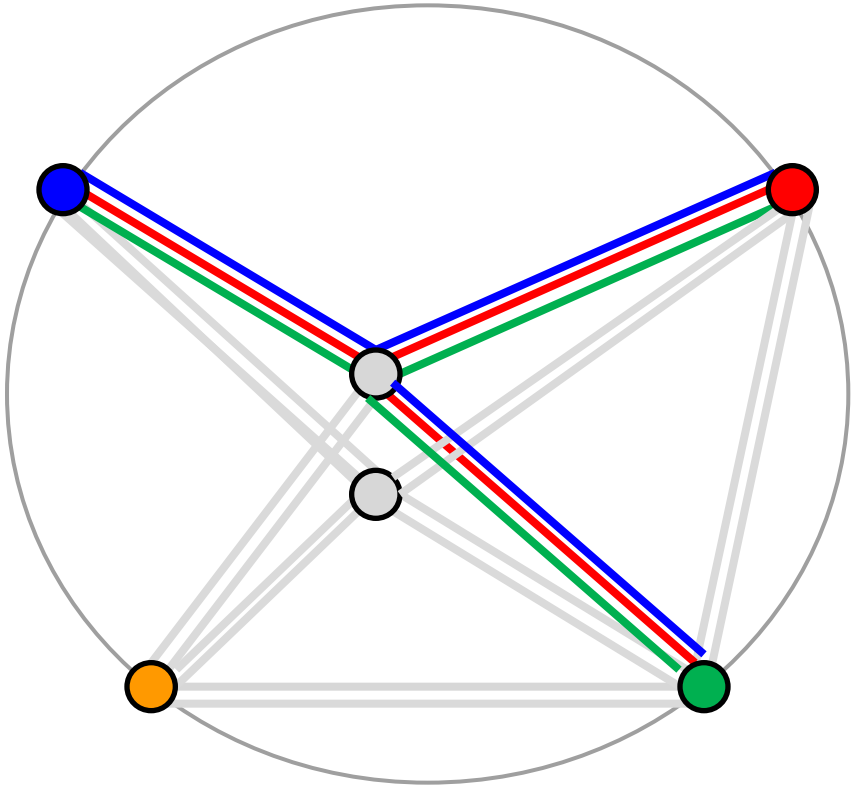
2nd option : Trees with unidirectional VIDs

This approach incrementally builds off-SPF trees – every pair-wise path must be co-routed :

- → first build the red root and blue root path elements
- → then extend an existing tree to add the red⇔green path
- → which can reuse the previous red root VID

Adding blue⇔green connectivity can reuse already assigned VIDs :

- → when using this routing ➔



ciena

2nd option : Trees with unidirectional VIDs

This approach incrementally builds off-SPF trees – every pair-wise path must be co-routed :

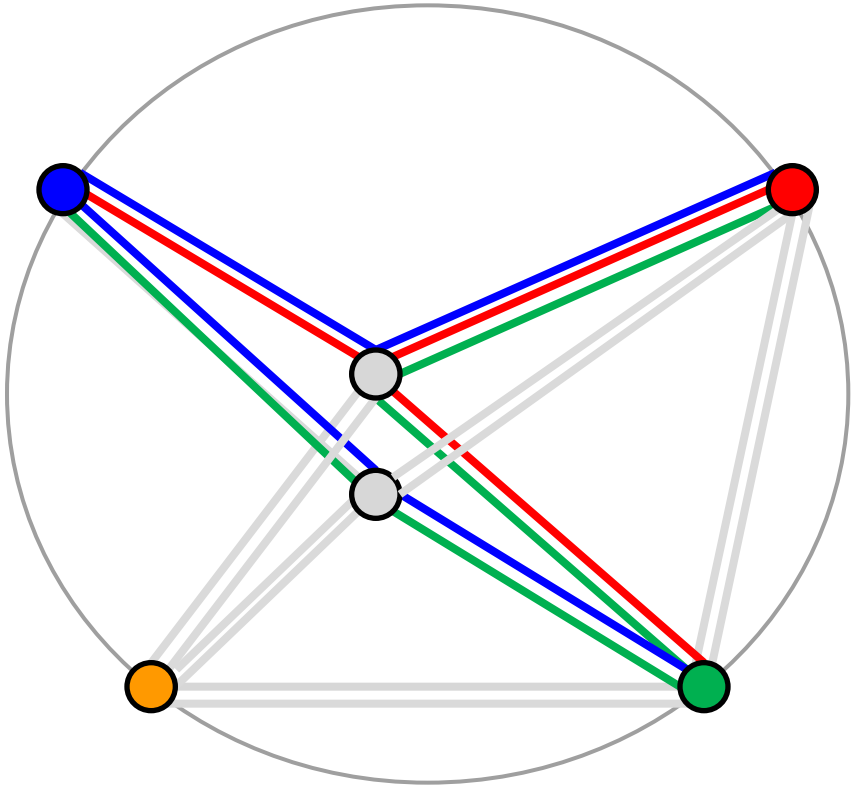
- → first build the red root and blue root path elements
- → then add the red⇔green path
- → which can reuse the previous red root VID

Adding blue⇔green connectivity can reuse already assigned VIDs :

- → or using this routing ➔

The trees remain simple trees in either case :

- → no meeting or crossing paths



ciena

2nd option : Trees with unidirectional VIDs

Adding blue⇔green connectivity reuses already assigned VIDs :

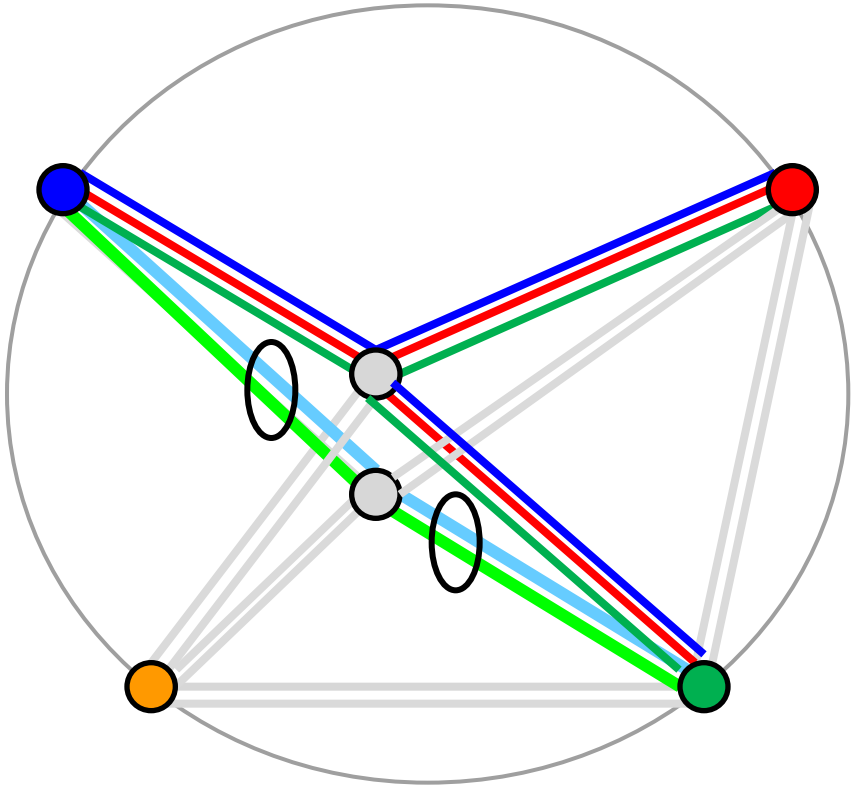
The trees have remained simple trees up to now.

When wishing to add a second blue⇔green route :

→ for example; ringed (➔)

new VIDs must be allocated at each end to prevent loops.

This process can be continued incrementally …



ciena

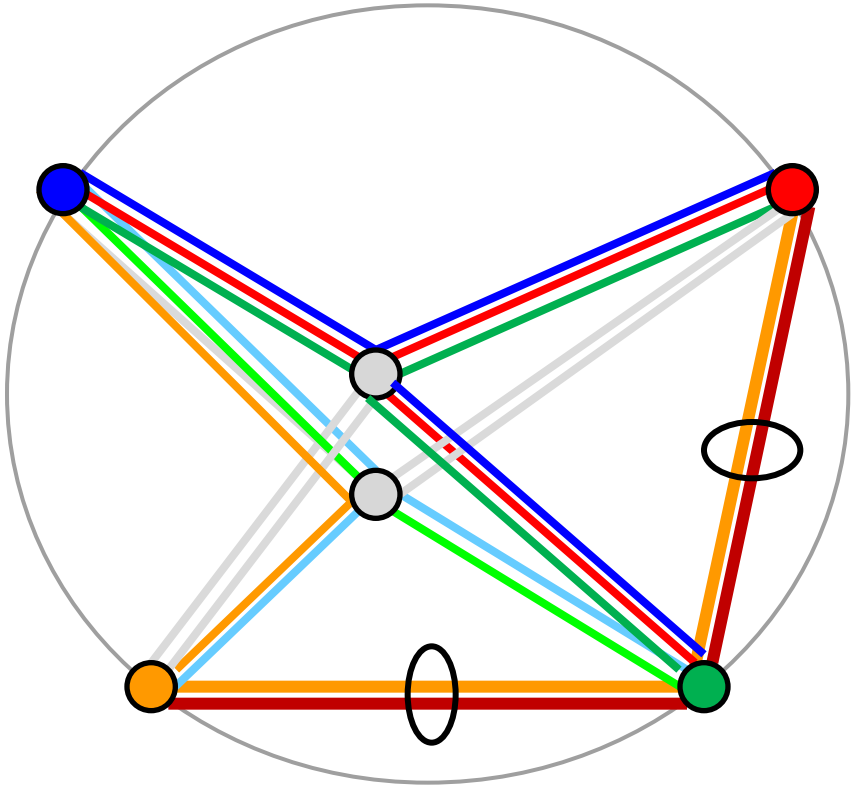
2nd option : Trees with unidirectional VIDs

This process can be continued incrementally :

- → for example, when adding the **orange**⇔**red** route (ringed), a new VID must be assigned at the **red** node because it already has a route to the **green** node
- → but the return path to the **orange** node is unconstrained unless it has already installed a tree to the **green** node via another path.

The lesson seems to be :

- → **First use an existing tree,**
- → **then extend an existing tree,**
- → **only then assign a new VID.**



ciena

A final thought :

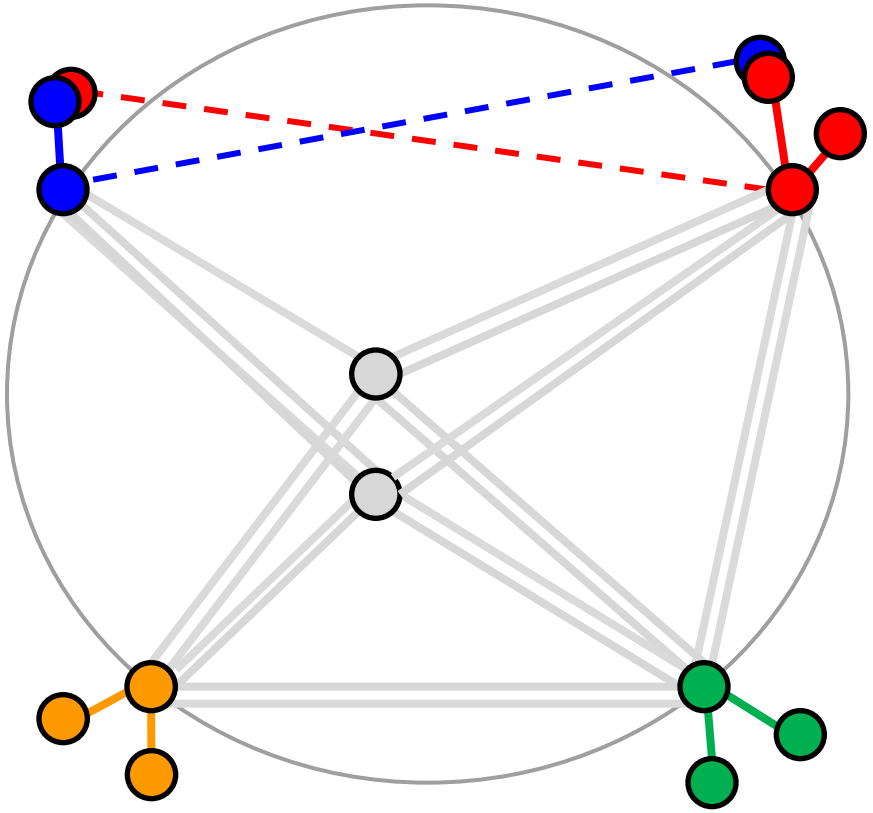
802.1Qca Edge Nodes do not need their "own" VIDs :

- → they can inherit the VID of their direct adjacency in the core,
- → because a loop or a path cross cannot be formed on a single Ethernet link (no 1 hop µloops)

This applies equally to multi-homed Edge Nodes :

- → provided that they never support **transit** connectivity,
- → as enforced by the PCE(s).

Maybe there is no practical scaling issue ?



ciena

Summary

When installing off-Shortest Path routes in an 802.1Qca environment :

- → **Use of a single bidirectional VID per p2p path / source-routed tree is a no-brainer up to a certain** (hardish)-**limited network scale :**
  - → scale is trivially guaranteed up to 4K such paths or trees;
  - → beyond that, VID reuse depends on the construct being formed :
    - → multicast trees offer lower reuse potential compared to p2p paths
- → **Use of a unidirectional VID per source-routed tree offers the potential of better incremental scaling properties :**
  - → a single VID can support connectivity from one root to all other nodes
    - → i.e. scaling is O(**N**), not O(**N^2**), in network nodes.
  - → VID reuse is not easy to predict, being very topology-dependant :
    - → however, in a typical (i.e. non-Fat-Tree) network, there are only a handful of useful routes across the network (~ # diverse paths through the core), and that small number of useful paths will be expected **# VIDs per node**.
- → **Enforcing "no Transit" at the network edge where possible has the potential to substantially reduce VID consumption.**

ciena