802.1Qbp
Shared Tree (*.G) Algorithms

Peter Ashwood-Smith
peter.ashwoodsmith@huawei.com

Motivation

- 802.1Qbp is introducing new ECMP behavior for unicast in an 802.1aq network.
- There is a desire to also do ECMP (head end) over SPBM multicast trees.
- So far we have only discussed the (S,G) trees (existing .1aq style and Ben's alternatives).
- I'd like to discuss some simple (*,G) options as state reduction likely more important than diversity.

N.B (S,G) is source/group specific tree, i.e. <SpSourceID>||<SID> in the DA
(*.G) is shared by all sources but one group i.e. <Constant>||<ISID> in the DA

Considerations

- What we really want is a minimum spanning tree that covers just a subset of the nodes (those in the ISID).
- This is referred to as a Steiner Tree.
- A Steiner Tree computation is **NP-complete**.
- “Non Polynomial” means its **>> O(N^c)** for any constant **c**.
- “Complete” is a fancy way of saying we ain’t gonna solve it here ..
- Basically its one of those problems that you have to enumerate all solutions and pick the best... And there are usually O(n!) solutions to pick from....

Solutions

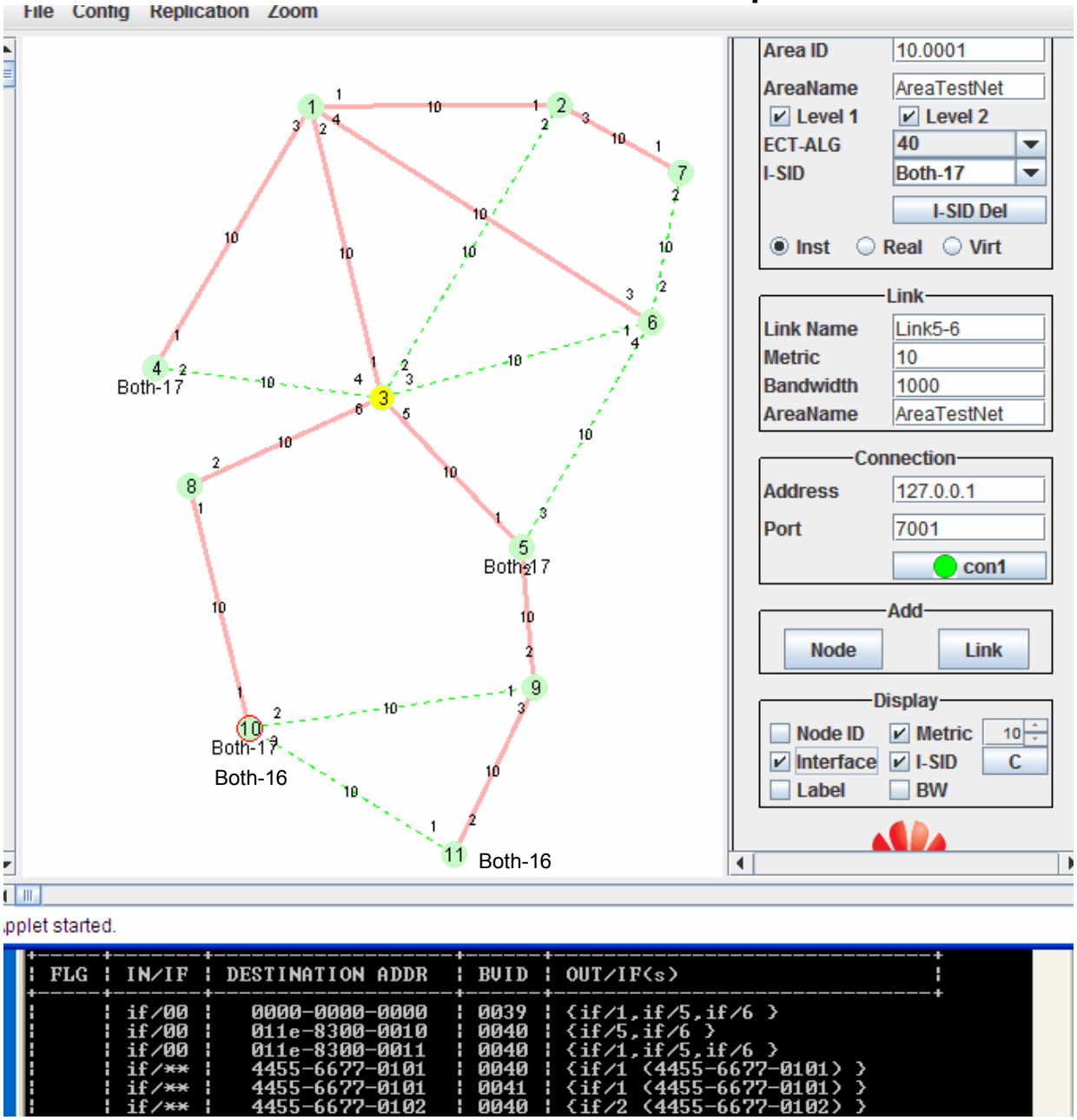
There are a few less optimal (*,G) solutions:

1. Pick some node as a root and use SPF from ‘it’ as tree.
   - This is O(n*logN) but sends traffic everywhere!!!
2. Modify above by pruning per ISID.
   - This is O(N*logN + I*logI)
   - Still non shortest path routing but state is minimal
3. Other solutions aimed at reducing non shortest path routing issues but increase CPU.. these are FFS.

Detailed Look at Option #2

- The 802.1aq CIST algorithm (which is just the STP algorithm done as a computation), can be reused for per ISID (*,G) trees in .1Qbp
- The multicast address format can be the existing PBB format i.e: ***00-1e-83-xx-xx-xx*** (where x.. Is the ISID)
- 16 different shared trees can be computed by finding the lowest BridgeIdentifier under the 16 .1aq ECT masks i.e. 0x00, 0xff, 0x11, 0x22 ... 0xee.
- These shared trees produce almost symmetric congruent results to the .1aq (S,G) trees in fat tree networks.
- Root selection automatic based on algorithm, auto recovery to new root etc. No explicit encoding of root in DA required.
- Can use F-TAG with TTL, or can rely on digest for loop prevention, or both....
- Can use same B-VID as unicast (no SVL), or different (with SVL) or even no B-VID.

Example #1



FLG	IN/IF	DESTINATION ADDR	BVID	OUT/IF(s)
	if/00	0000-0000-0000	0039	{if/1,if/5,if/6 }
	if/00	011e-8300-0010	0040	{if/5,if/6 }
	if/00	011e-8300-0011	0040	{if/1,if/5,if/6 }
	if/**	4455-6677-0101	0040	{if/1 (4455-6677-0101) }
	if/**	4455-6677-0101	0041	{if/1 (4455-6677-0101) }
	if/**	4455-6677-0102	0040	{if/2 (4455-6677-0102) }

Real CIST
Pruned for ISID 16
Pruned for ISID 17
Unicast entries ...

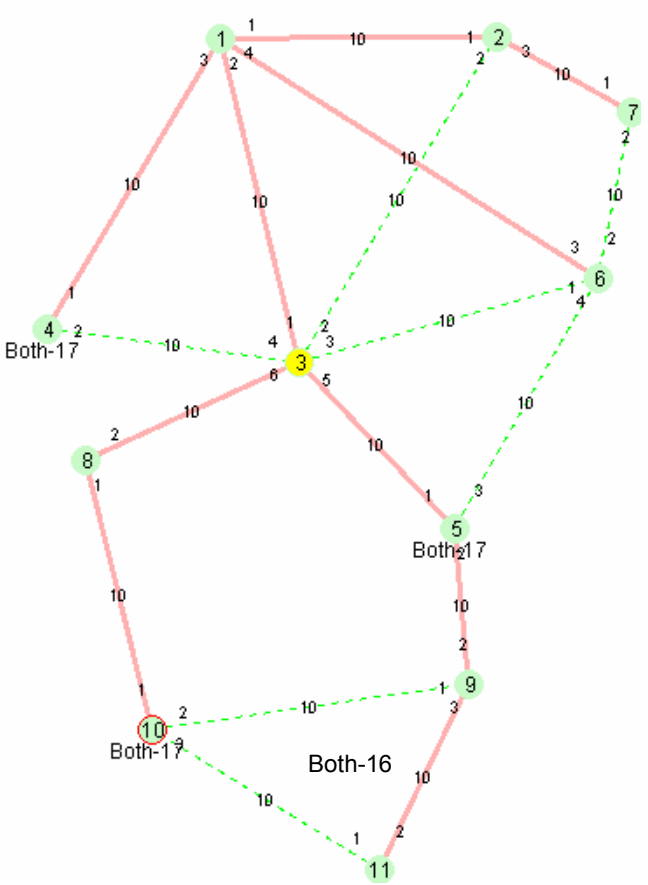
A (*,G) is computed using the Lowest Bridge Identifier (node 1) CIST algorithm.

The full tree is shown in pink.

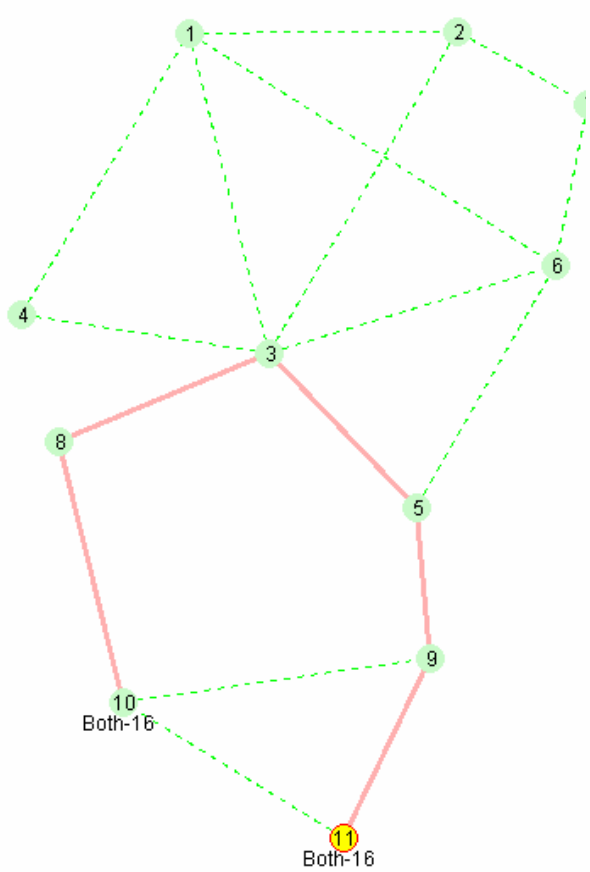
Two ISIDs are pruned against this tree for Multicast, sub trees below: ISID 17 and ISID 16.

We show the Mcast state at node 3 for Each ISID.

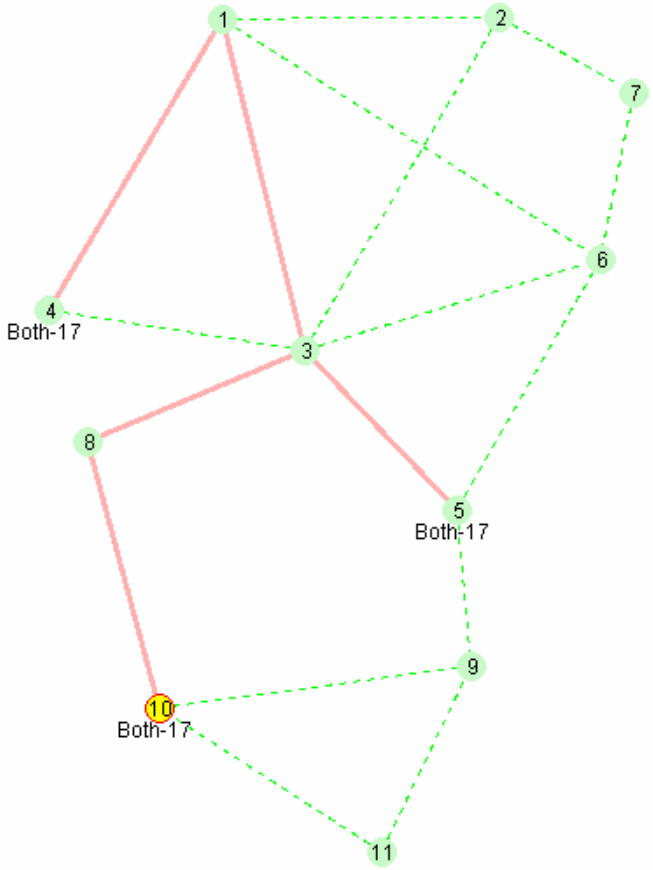
Example #1 – pruning



FULL MASK0x00
(ROOT=1) TREE

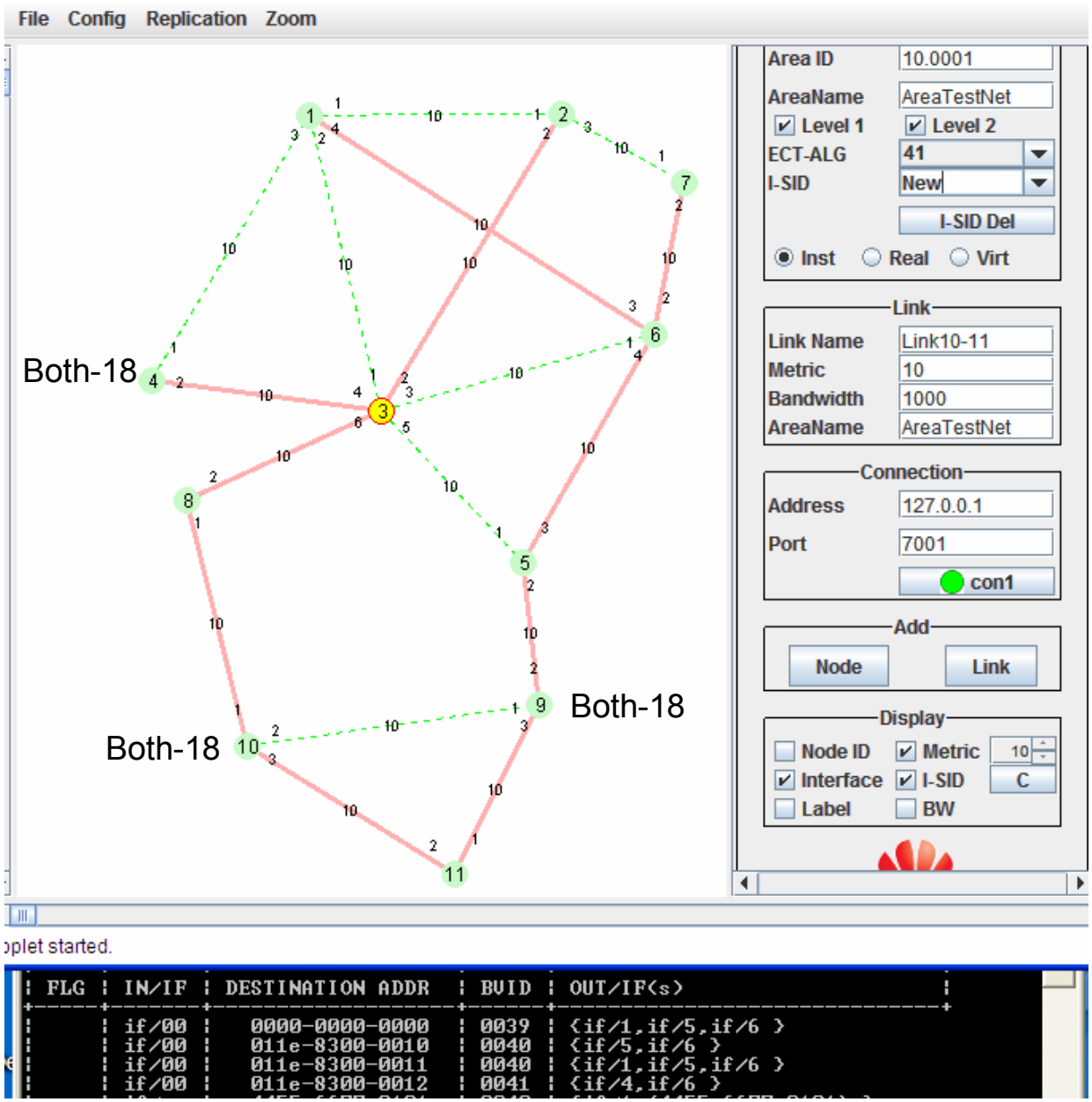


ISID 16
PRUNED



ISID 17
PRUNED

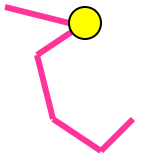
Example#2



A (*,G) is computed using the **highest** Bridge Identifier (node 11) i.e. CIST algorithm XOR 0xff.

The full tree is shown in pink.

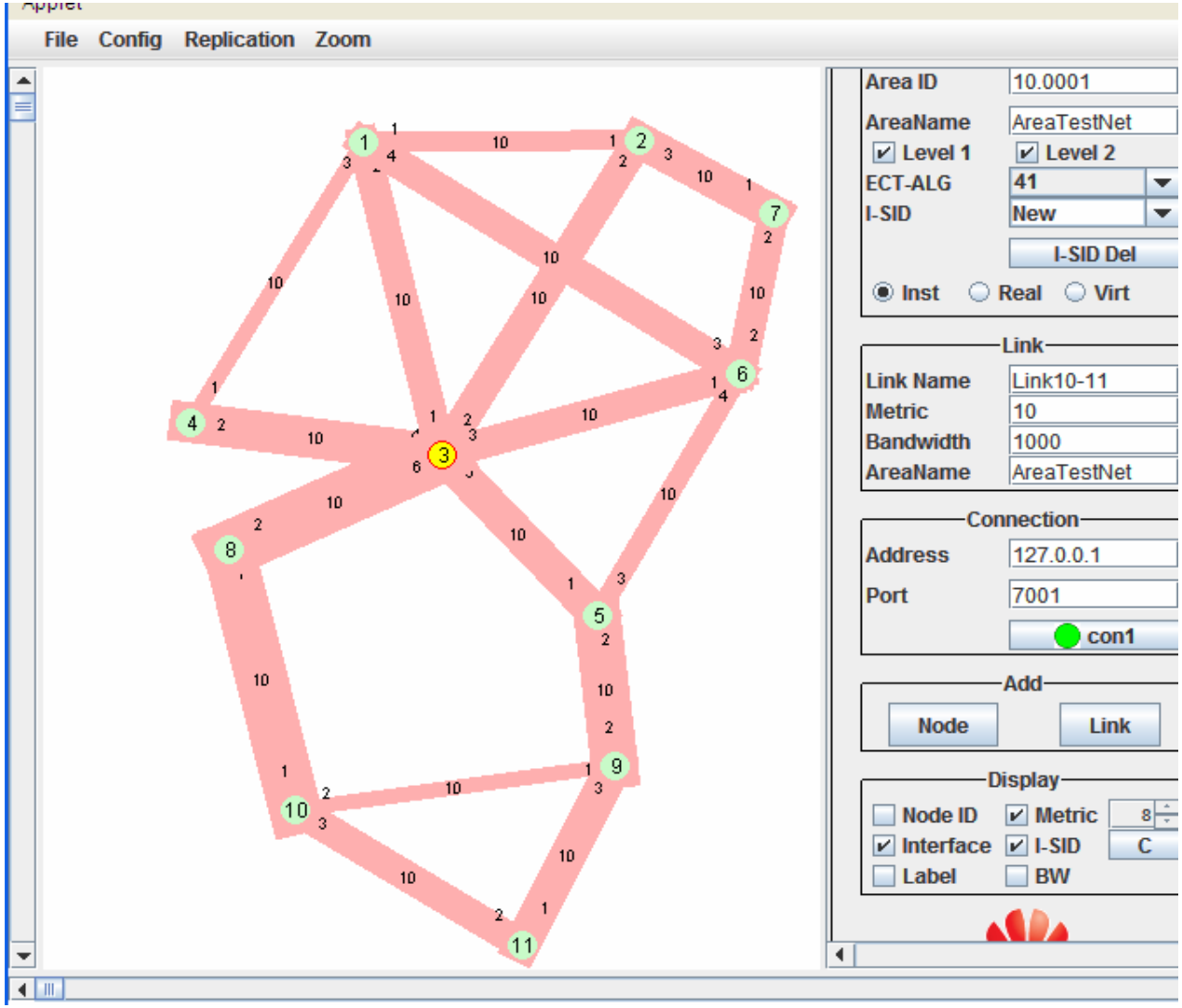
One ISIDs is pruned against this tree for Multicast, sub trees below: ISID 18



We show the Mcast state at node 3 for Each ISID.

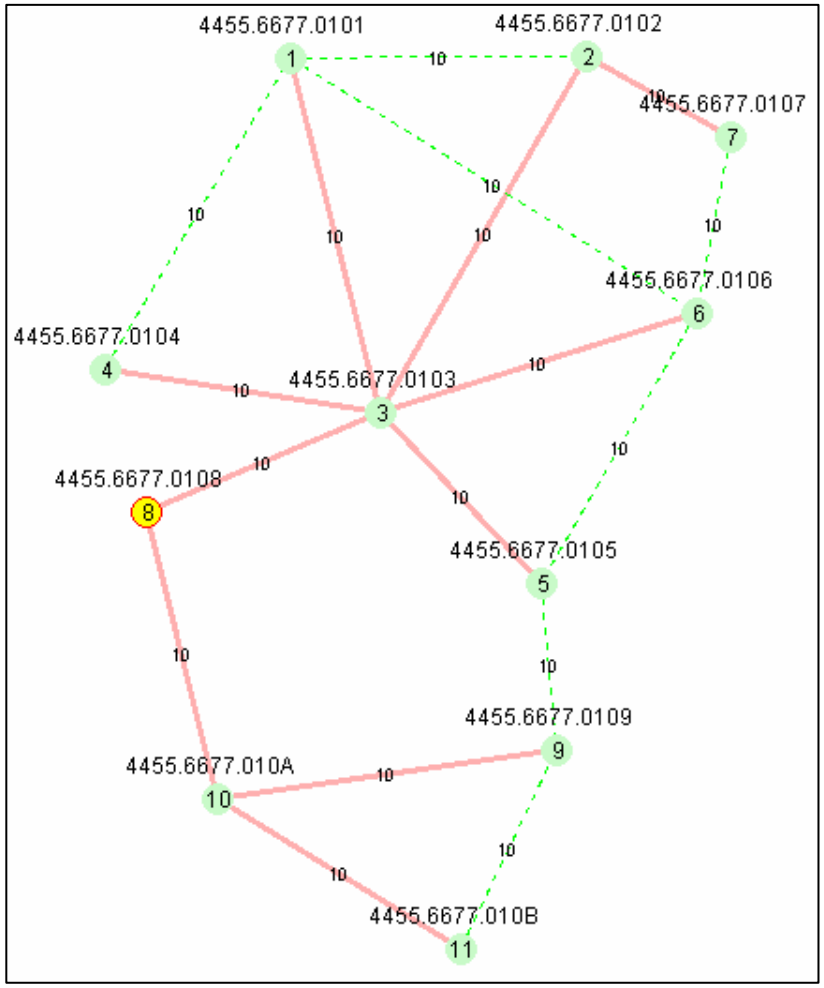
Real CIST
Pruned for ISID 16
Pruned for ISID 17
Pruned for ISID 18
Unicast entries ...

Example#3- Coverage is not bad

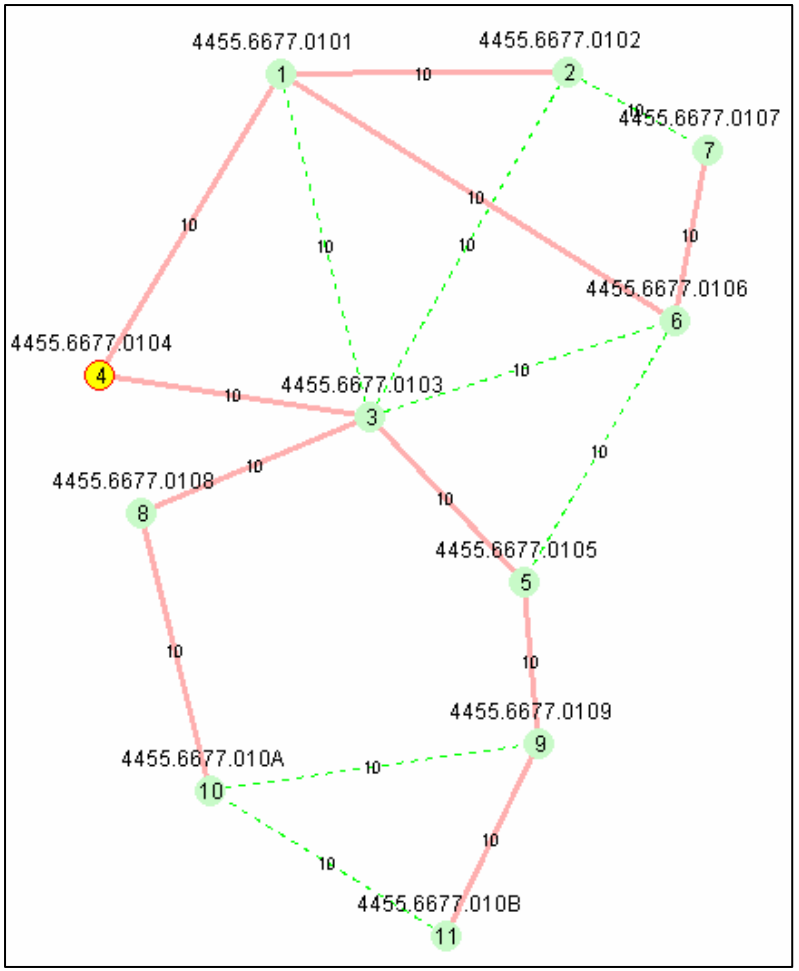


ALL 16 (*,G) Trees shown superimposed. Basically the CIST algorithm
16 times but with different root choices based on BridgeIdentifier XOR Mask[i]

Example#3- Some of the individual trees

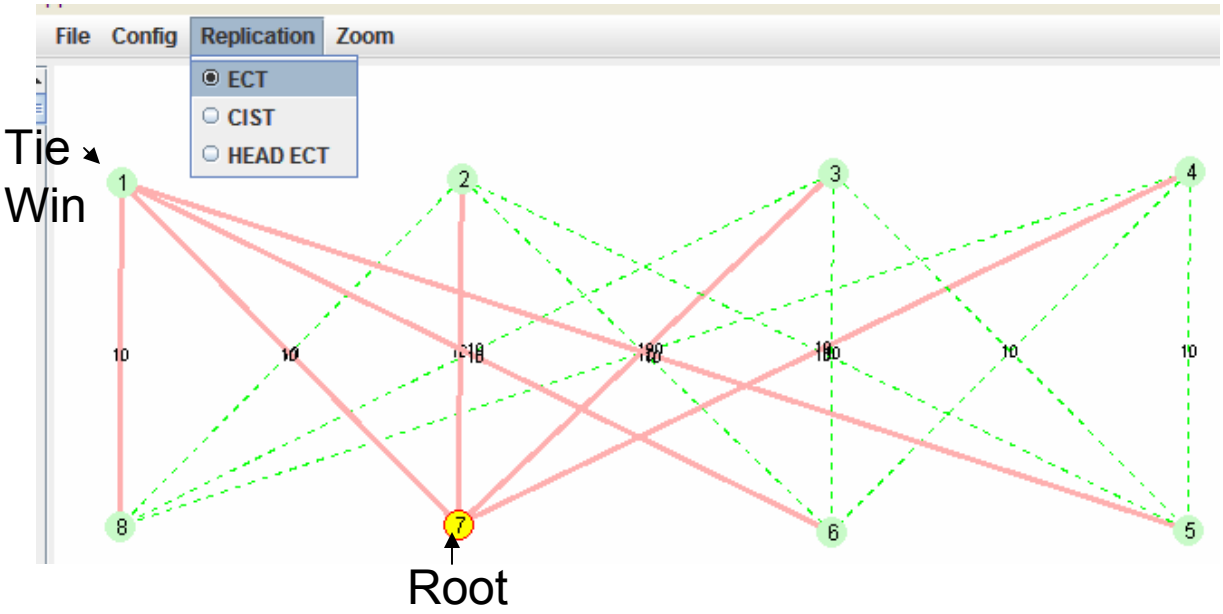


ALG MASK=0x8888.
So node .. 108 is root.



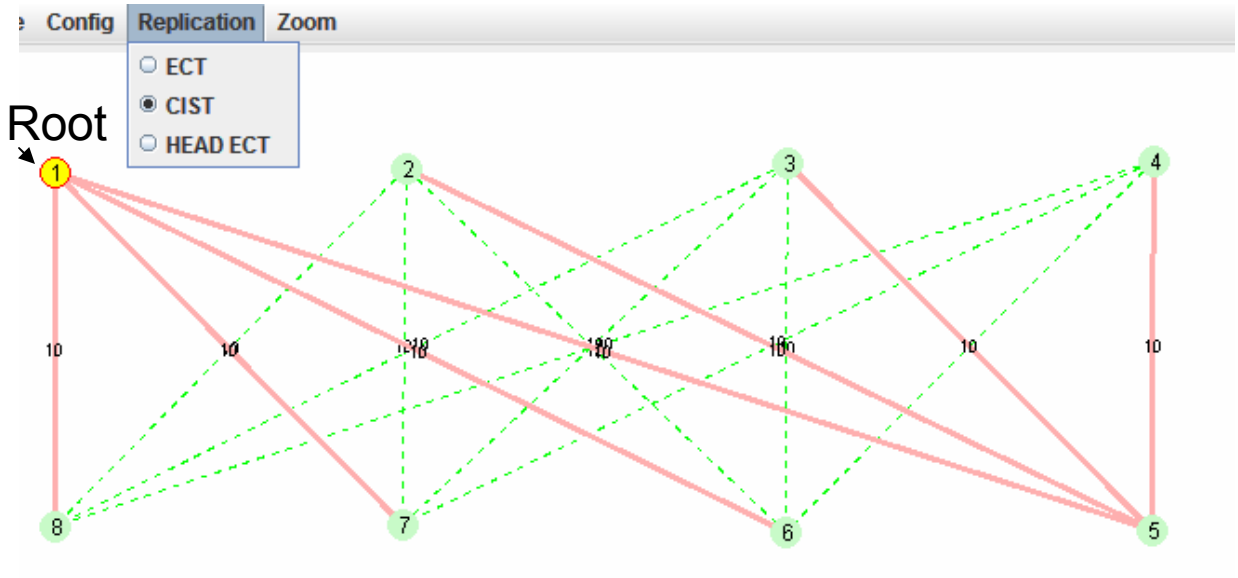
ALG MASK=0x444444.
so node 104 is root.

Example#3- Comparison to ECT source tree in Fat Tree



(S,G) Tree

Unicast and Mcast Routes from Node 7 to all other nodes.



(*,G) TREE

Multicast Shared Tree Routes from node 7.

Note routes to all other leaves 8,6,5 is identical To (S,G) tree above.

Option #2 Basic Algorithm

Compute Shared Tree (alg, self) { // alg==0 => .1aq CIST

root = find lowest BridgeIdentifier **XOR Mask[alg]**

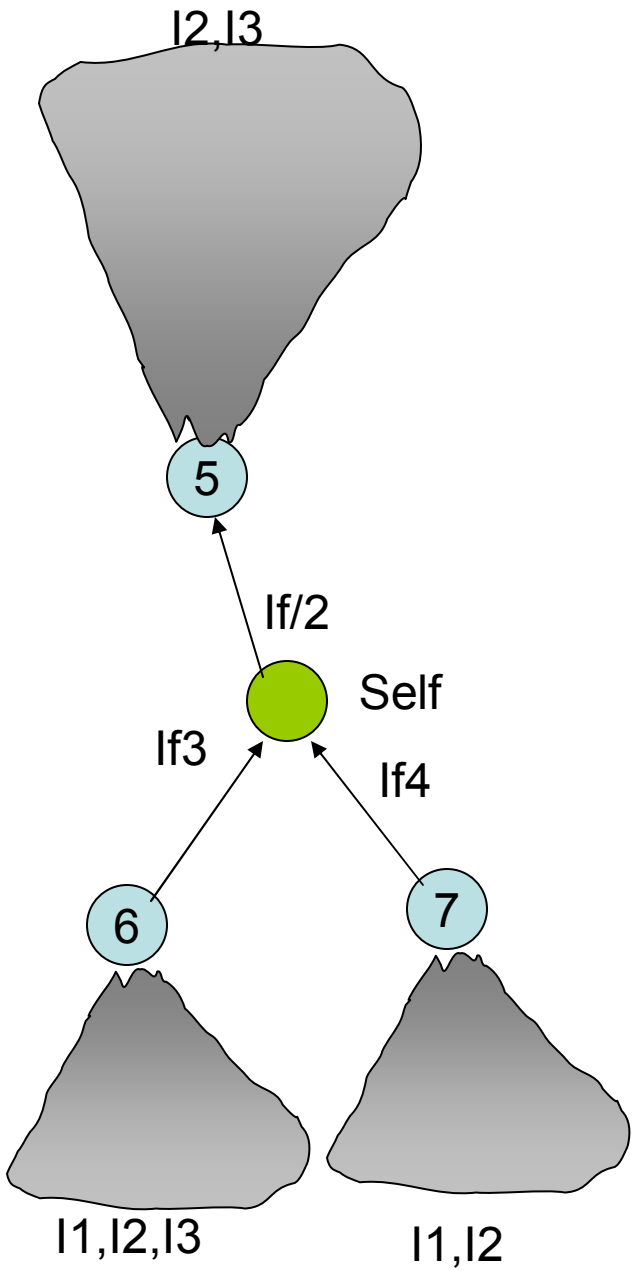
run SPF from root where

tie break on equal cost winner =
lowestBridgeIdentifier **XOR Mask[alg]**

}

Multicast DA per ISID can then easily be generated by sorting the set of all ISIDs and the interface to reach that that ISID.

So total run time is O(16 x [(NxLog(N)) + (I x Log(I))])



Option #2 More detail

1. At self do the SPF from selected root.
   Result is upward pointing parent pointers to root.

2. For each node in network assign it the local interface that reaches it. Eg: 5 and everything above it via if/2; 7 and everything below it by if/4 etc.

3. Then traverse network and generate a list of.
   <ISID, IF/#> records ..will have lots of duplicates.

I2,if/2
I3,if/2
I1,if/3
I2,if/3
I3,If/3
I1,if/4
I2,if/4

4
SORT →

I1,if/3
I1,if/4
i2,If/2
i2,if/3
i2,If/4
i3,if/2
i3,if/3

Traverse and generate Mcast

DA=00-1e-83-00-00-02
OutIF = {if/2, if/3, if/4]

Ignore if only reachable via one interface ..

108 node example – ISID 100 with 4 attachment points