RIFT: Routing In Fat Trees

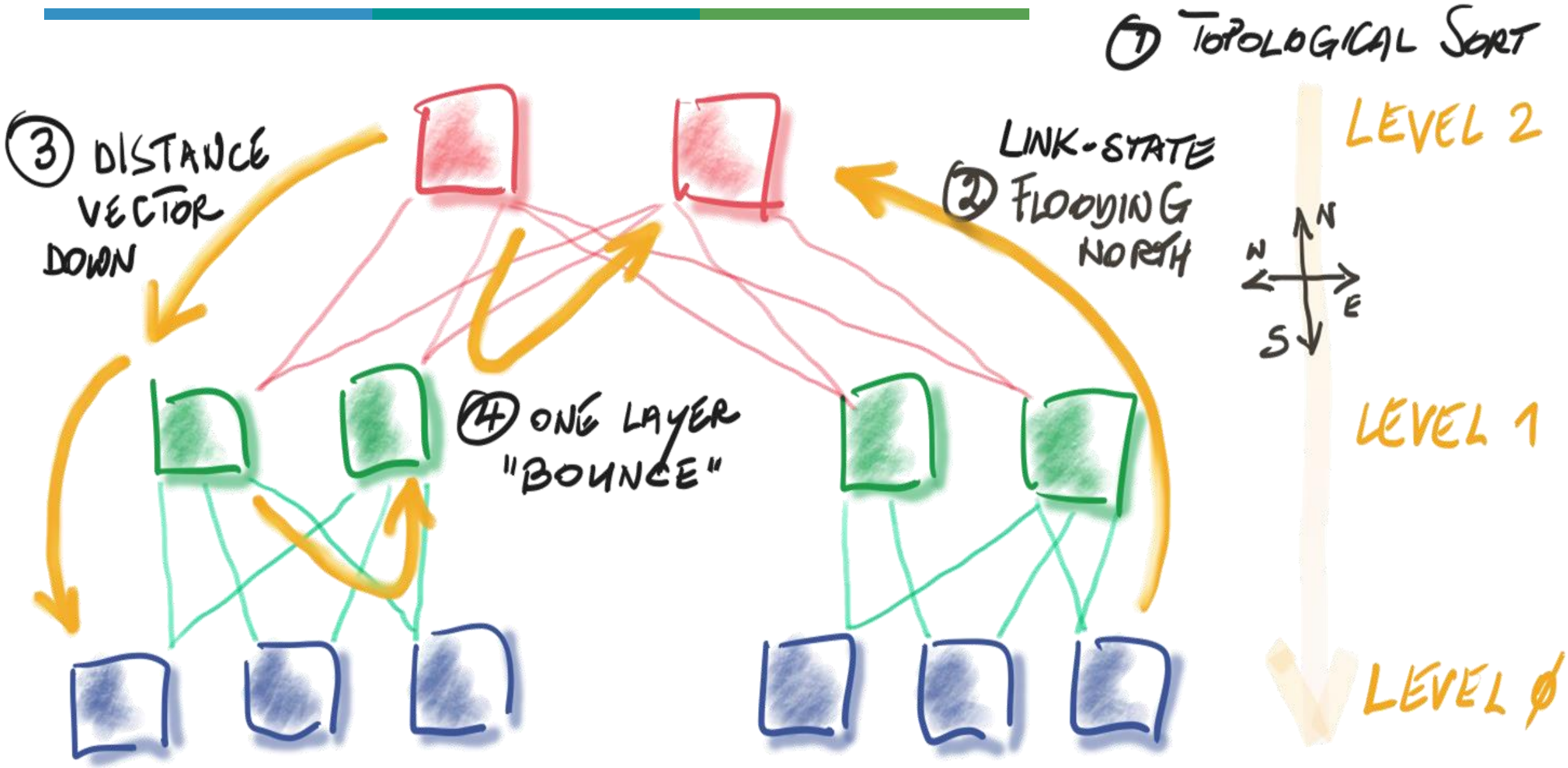
Jeffrey Zhang

Routing Area Open Meeting, IETF 103

Why RIFT?

- DC Underlay routing evolution: IGP → eBGP → RIFT
  - For scaling, convergence and Opex considerations
- Issues with IGP
  - Failure Impact Scope (aka Blast Radius)
    - A small change (e.g. a single link up/down on a leaf) is flooded everywhere, triggering SPF recalculation on every node
  - Rich connections make flooding unnecessarily redundant and inefficient
- Issues with eBGP
  - Cannot take advantages of well defined network topology
    - E.g. ideally a leaf (tier-3) node only needs a default route, and a tier-2 node only needs a default route and routes for destinations south of it
      - This cannot be done due to black-holing upon link failure
  - A node needs to keep all paths learnt from different peers
    - A leaf node connecting to 32 tier-2 nodes needs to keep 32 paths for each of all the prefixes in the DC

RIFT: Link-State Up, Distance Vector Down & Bounce



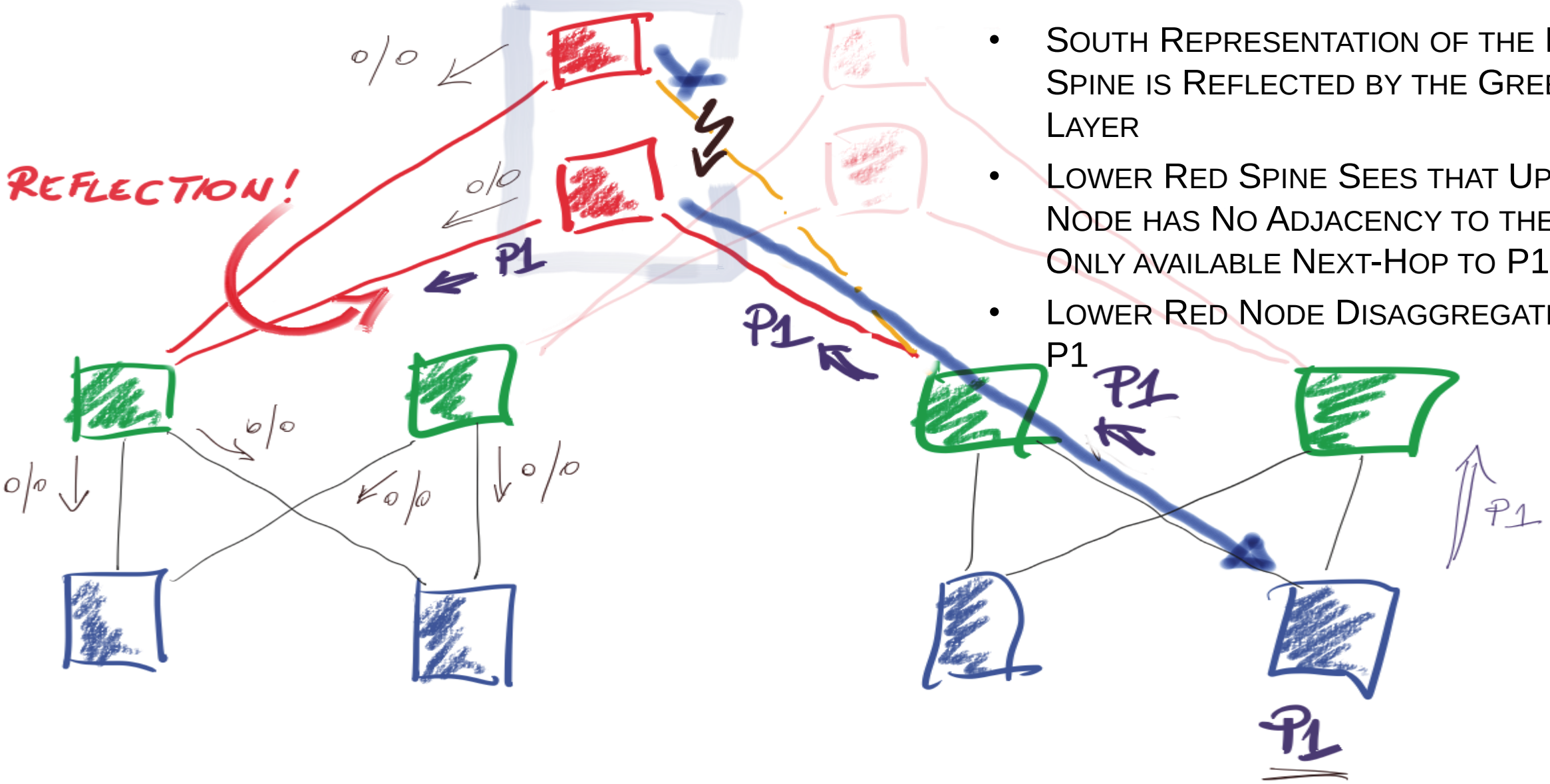
Northbound LSR

- Link State flooded northbound to the top tier
  - With flooding reduction
  - Each node has full view of the southbound topology
    - A top tier node has full set of prefixes from the SPF calculation
    - A middle tier node has only information necessary for its level
      - All destinations south of the node, from its SPF calculation
      - Default route (next slide)
      - Potential disaggregated routes (next slide)
- Fast convergence and ECMP benefits of LSR

Southbound Distance Vector Routing

- Default route and automatically disaggregated routes (when needed) advertised one-hop southbound
  - When a level-2 node A detects that another level-2 node B cannot reach one of A's south destinations P, it advertises P via southbound DVR
    - That way a south level-3 node will route P traffic only towards A (via the more specific route) not towards B (via the default route)
- A node's local link state is advertised one-hop southbound and then reflected one-hop northbound
  - So that node A can detect if node B can reach A's south destinations
  - Other than that, link state is not propagated south, greatly reducing impact scope

Automatic De-Aggregation



- South Representation of the Red Spine is Reflected by the Green Layer
- Lower Red Spine Sees that Upper Node has No Adjacency to the Only available Next-Hop to P1
- Lower Red Node Disaggregates P1

Zero Touch Provisioning

- Only top tier nodes need to be configured
  - Nodes that must be leaves or have leaf-leaf connection may be configured
  - Nodes with specific configuration can be mixed with others
- Upon connection nodes will fully auto-configure themselves and form adjacencies in a well defined north/south topology
  - With optional east-west connections
- ZTP makes DC fabric like RAM banks
  - No one configures RAM banks and CAS/RAS manually in a laptop
  - DC fabric HW is largely commodity already
  - DC fabric OPEX must and will commoditize
    - RIFT enables that

Other Features of RIFT

- Optimal Reduction and Load-Balancing of Flooding
- Mobility Support
  - Built-in support for rapid prefix moving from one leaf to another
- Key/Value Store
- Fabric Bandwidth Balancing
  - Northbound: modify the distance of default route received from a neighbor based on available BW through that neighbor
  - Southbound: during SPF consider available BW through lower level nodes
- Weighted all paths routing (RIFT is loop-free)
- Segment Routing Support
- Leaf-to-leaf Procedures
  - Allow E-W traffic strictly for local prefixes
- Policy Guided Prefixes
  - Moved to a separate draft

Summary of RIFT Advantages

- Advantages of Link-State and Distance Vector
  - Fastest Possible Convergence
  - Automatic Detection of Topology
  - Minimal Routes/Info on TORs
  - High Degree of ECMP
  - Fast De-comissioning of Nodes
  - Maximum Propagation Speed with Flexible # Prefixes in an Update
- No Disadvantages of Link-State or Distance Vector
  - Reduced and Balanced Flooding
  - Automatic Neighbor Detection
- Unique RIFT Advantages
  - True ZTP
  - Minimal Blast Radius on Failures
  - Can Utilize All Paths Through Fabric Without Looping
  - Automatic Disaggregation on Failures
  - Simple Leaf Implementation that Can Scale Down to Servers
  - Key-Value Store
  - Horizontal Links Used for Protection Only
  - Supports Non-Equal Cost Multipath and Can Replace MC-LAG
  - Optimal Flooding Reduction and Load-Balancing