Preferred Path Routing (**PPR**) **Graphs** – Beyond Signaling Of **Paths** To Networks

CNSM 2018 – HIPNET workshop

Toerless Eckert, Yingzhen Qu, Uma Chunduri

HUAWEI – *Future Networks*

Santa Clara, California, USA {firstname.lastname}@huawei.com

version 1.0

PPR Paths

BACKGROUND

Disclaimer **Why ?**

High Precision networked applications need High Precision network services

Many High Precision application still do not use Internet Technology (IP) networks.
Or are constrained to specially designed IP Networks. Not multi-services ISPs networks.
How about future High Precision applications ?

Contribution Video, Manufacturing Networks / Time Sensitive Networking (DetNet),
Tele-Surgery, orchestrated real-time transportation, Holography Teleportation/Avatars, …

Solution Overview **How?**

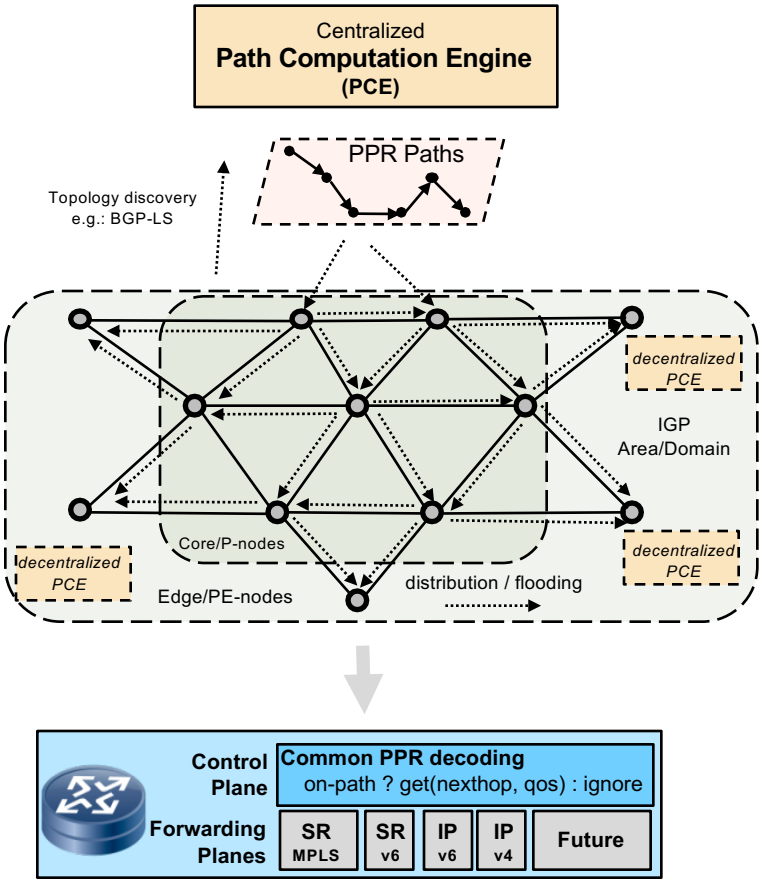
Compare with closest pre-existing technology

- RSVP-TE: signals path hop-by-hop sender<->receiver
- RSVP-TE IntServ: install per-hop QoS

PPR floods path/resource-parameters across network

- For example: LSP-IGP (IS-IS, OSPF)
- ***All nodes examine PPR path object***
- ***Nodes on-path establish forwarding/resource state***
- PPR is for IPv4 / IPv6 / SR-MPLS or future forwarding

  RSVP-TE only supports MPLS



This is crazy **Why ?**

- **This is a waste, not !???**
  - *Most nodes do not care about the path and have to examine PPR path information ???*

- **RSVP-TE complexity and Hop-by-hop serial processing is what impacts performance**
  - Per-hop/per-path delay due to per-hop state lookup/state-create sequentialisation
  - Even further delay when ASIC forwarding state creation is serialized with propagation

- **PPR:**
  - Parallel state building across all nodes (besides flooding propagation delay)
  - No per-path + propagation protocol state machineries -> easier to optimize implementations

- **We think PPR paths will perform better than RSVP-TE!**
  - Traffic flooding can be high performance / low overhead
    - IGP flooding already scales well, future accelerations/variations can do better
  - Examining and discarding irrelevant paths is single read over path information
    - Single CPU core could examine >> 100,000 of paths/sec (memory read access speed)

Broadcasting works

Why reinvent a dead horse **This will not scale !??**

- ***Full Mesh p2p paths between N nodes: up to O(N^2/k) in RSVP-TE / PPR Paths***
  - RSVP-TE suggested to be succeeded by SR (Segment Routing) because of scale ?!

- **PPR Paths does not have control-plane state overhead of RSVP-TE**
  - Just the same amount of forwarding entries (for multiple forwarding planes)

- **SR does not support high-precision per-hop path and QoS engineering:**
  - **Network Capacity Optimization**

    Core use case for SR: Lightweight path steering for Best Effort Internet Traffic
- **SR can not scale (arbitrary) either: Header size limit, Header-Tax !**

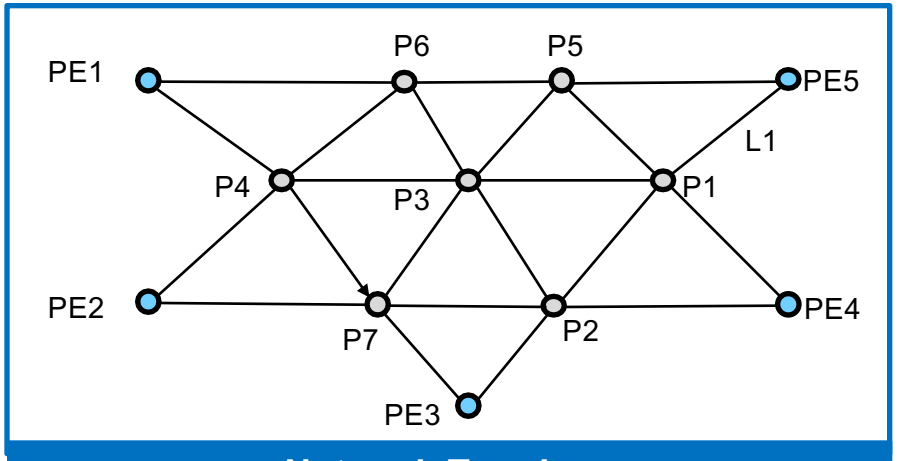
- **Scale PPR O(N^2/k) -> O(N) or even better with PPR Graphs**

PPR Graphs

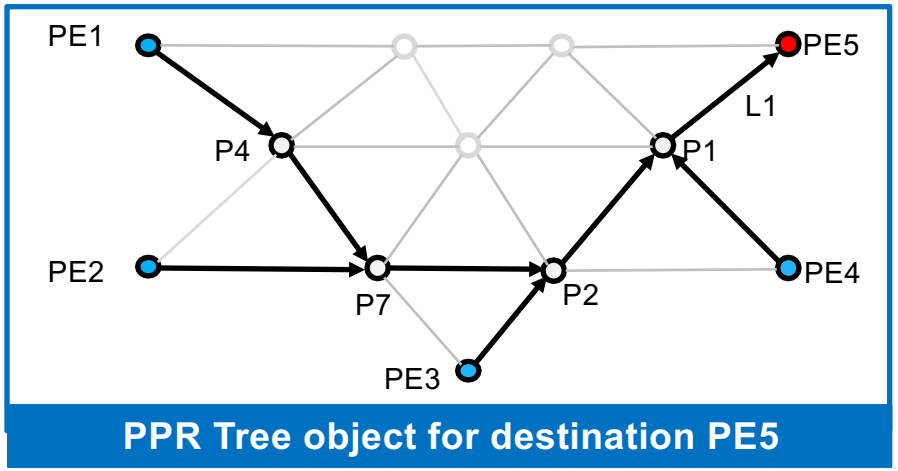
ARCHITECTURE

PPR Trees

- **PEx**: possible traffic source/destination,
  Px: only transit nodes
- **PPR Tree** = most fundamental PPR graph
  - Every PPR path is aaso a PPR Tree
- **PPR Object (Tree)** flooded into network
  - P3, P6, P5 ignore it (not on-tree)
  - PE1, PE2, PE3, PE4, P4, P7, P2, P1 : establish ONE forwarding entry for PE5 plus optional QoS parameters for this entry
- Ideal case: N destinations = N forwarding entries
  - Just like IGP routing table entries
    More efficient: Only entries in required nodes
  - **O(N) forwarding entries, PPR Trees**
  - **QoS entries depending on requirements**



Network Topology



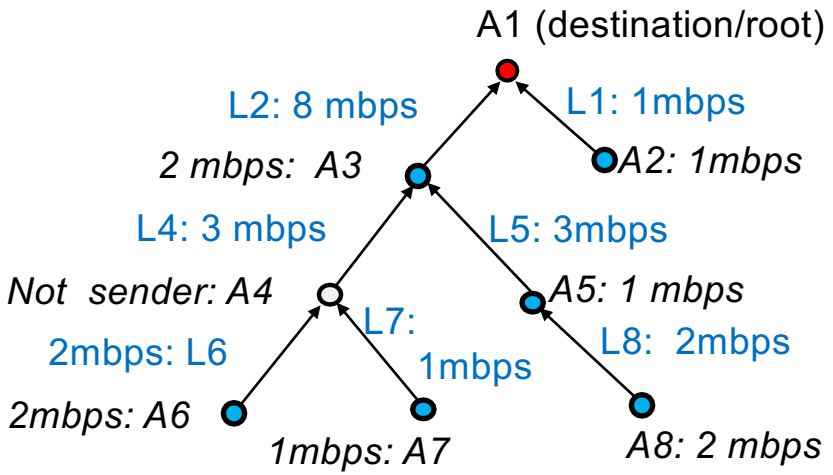
PPR Tree object for destination PE5

QoS for PPR Trees

- One encoding option for scalable forwarding/QoS
  - One Forwarding/QoS state to a destination
- For group traffic: TP/AR/VR participant->mixer
- Each participant can send to mixer/destination
  - Sender PDE encodes senders bandwidth
- Each node calculates sum of sender bandwidths passing through it. Including itself (if its sender)
  - Resulting bandwidth reserved for PPR-Tree forwarding entry on the outgoing interface

Compact encoding, O(N) QoS for multiple senders

- Easily extended. Often known maximum number of senders to limit max bandwidth requirements.



Group QoS for PPR Trees

Encoding

- **Branch: basic component of Graph**

  List of “Path Description Elements” (PDE)

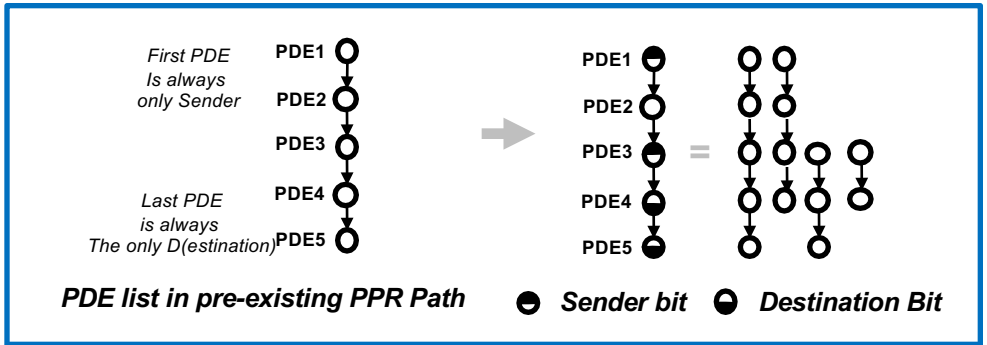
  Each transit or sender and/or destination

- **Graph composed from Branches**

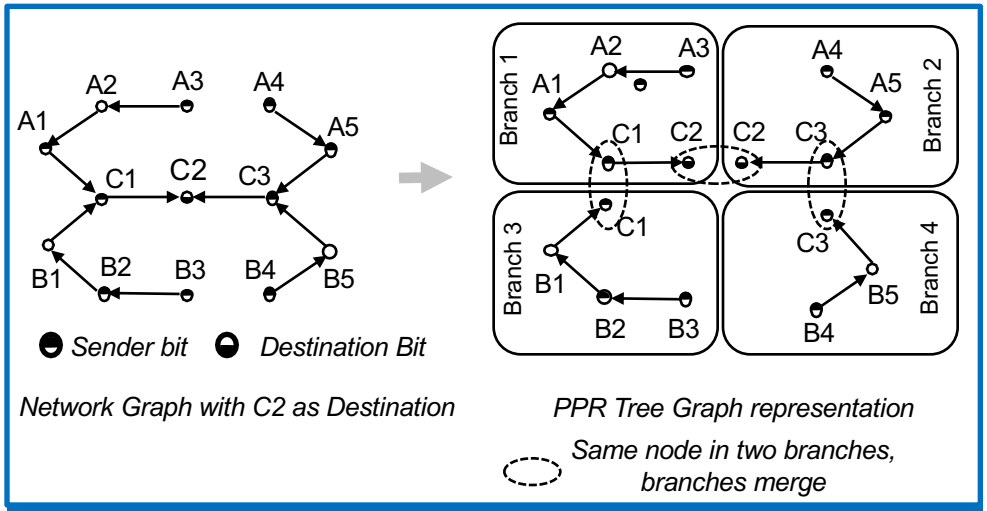
  Interconnected by PDE listed in both branches.

  Interconnect PDE must be first or last PDE in all but one branch

  Results in easier to parse Graphs



Branch with Sender and Destination bits



Composition of Graphs from Branches

Fragmentation

Message sizes in flooding mechanisms are limited. Logical fragmentation

Parameters: QoS and more

Also used to fragment really long paths

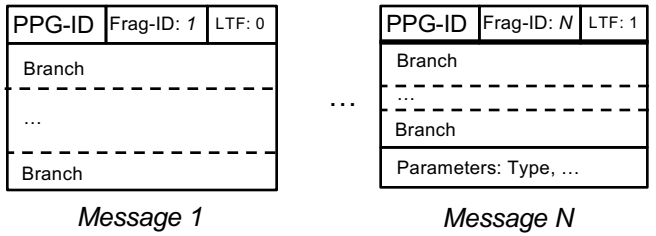


Figure 4: Fragmentation of PPR Graphs

Graph Types

Tree: Graph with one destination

Forest: Graph with multiple destinations

- Calculate separate forwarding entry for each destination reachable on graph!

Graphs have no forwarding ambiguity !!!

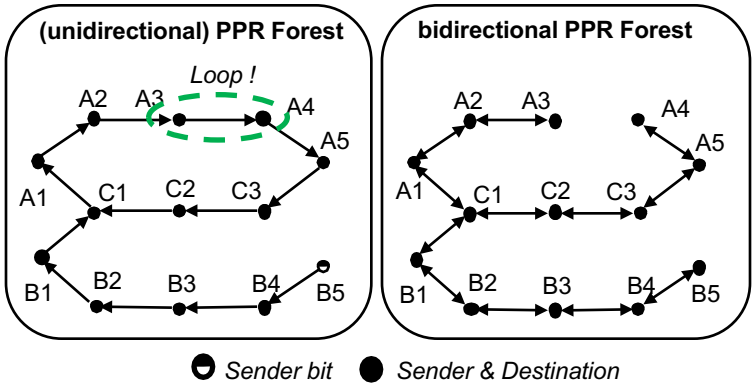


Figure 5: PPR Forest and Bidir-Forest

USE CASES / COMPARISONS

URLLC: Dual Transmission

- **Low Latency + Low Loss**

  Retransmissions/FEC adds delay, not redundancy !

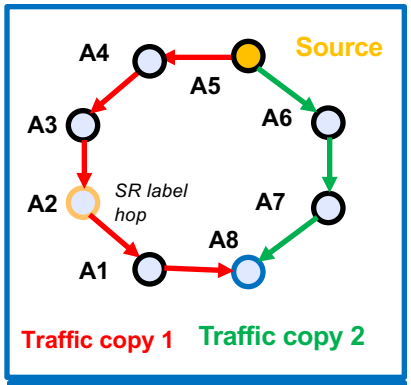
  Transfer traffic twice across different paths

  Disjoint paths: no single point of failure

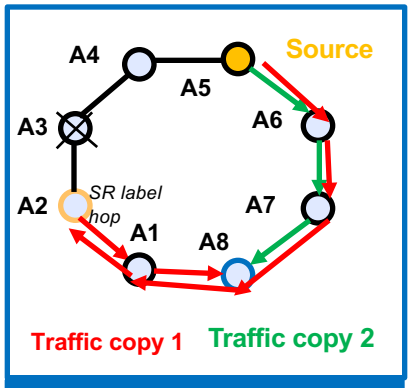
- **Loose path steering not good enough**

  Full hop-by-hop steering or constraint of reroutes

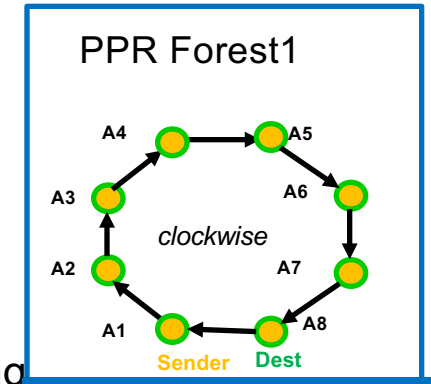
- **2 PPR Forests to represent any-2-any in ring**
  - 2 forwarding entry per destination:
    - clockwise / counterclockwise
  - Strict label stack headers would not only be too long but one order of magnitude more header memory: O(2*N^2/2) for SR vs O(2 * N) PPR nexhop entries

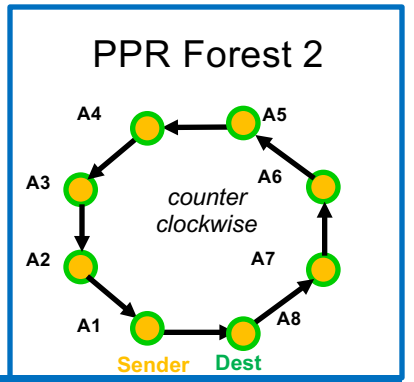


Live-live transmission In ring with SR



Rerouting under Failure





2 PPR Forests representing any path in ring topology

Real World Dual Transmission

”Subtended” rings

- Lowest cost redundant network topologies
- Popular in wide area networks with need for many breakouts
- Long paths / many hops

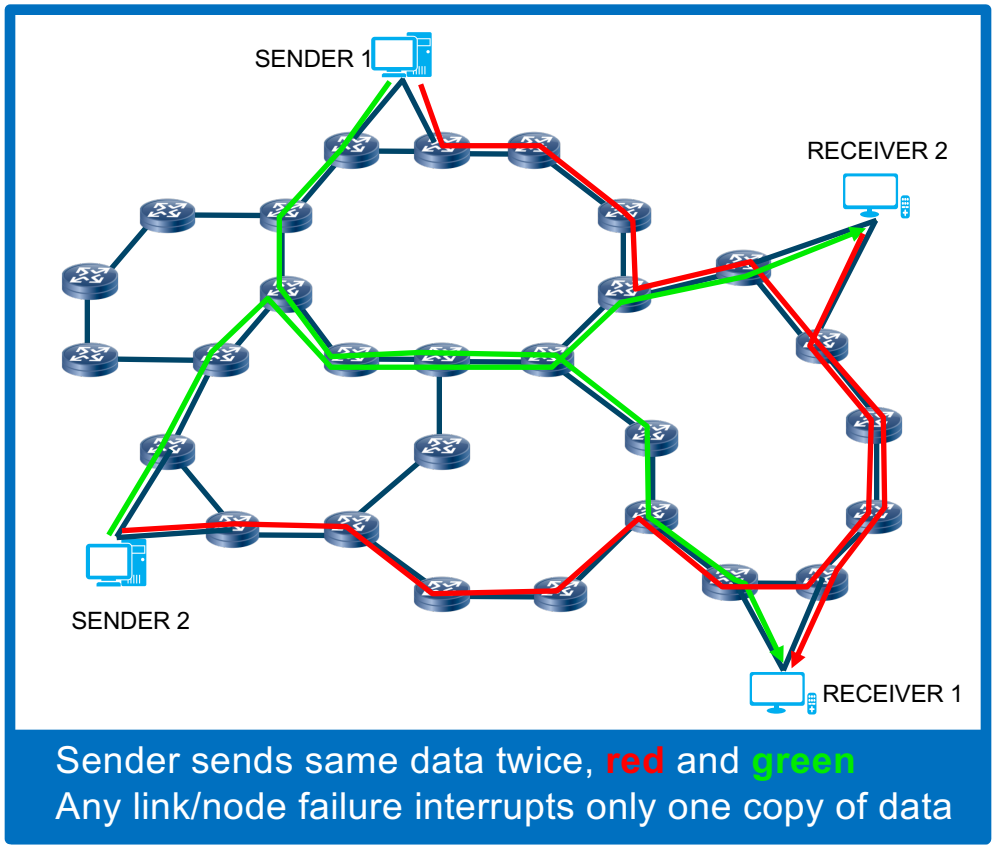
Dual transmission attractive

- Would need to have backup capacity anyhow

Can not avoid undesired reconvergence with simple ring specific solutions

- Other workaround (multiple topologies) also very difficult to use in these topologies

Strict hop-by-hop paths (via PPR)most ‘simple’ solution ?! (just one mechanism used).



Sender sends same data twice, **red** and **green**
Any link/node failure interrupts only one copy of data

Traffic Steering / capacity optimization

Strict / Loose p2p paths:

RSVP-TE: strict/loose hop-by-hop state

PPR paths share same amount of state, but different/better signaling

SR: strict/loose hop-by-hop state in First Hop Routers SID-stack

Same number of explicit next-hops in same set of paths SR vs. RSVP-TE, but state moved to FHR

Long comparison of benefits/downside RSVP-TE hop-by-hop state vs. SR in-packet-header/FHR state

IGP Metric Engineering (also called IGP Topology Engineering)

Lightweight in network and excellent scale: 1 forwarding state/destination

Can not optimize paths for different destinations independently

Metric changes impacts all/most destinations

PPR Trees ~"just" like IGP route (slide 8), can be optimized independently per destination !

And even more flexible

Multiple PPR Trees

1…N Trees to a destination – as necessary

Example: *too much traffic for L1 alone* 2 Trees

2 Forwarding entries / different nexthop in P1

Makes 2 trees necessary

Interesting open optimization research questions

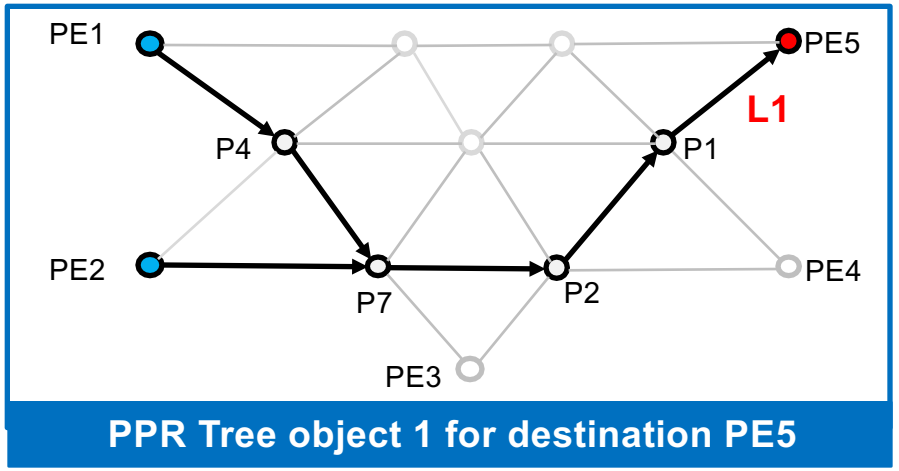
What is the number of trees required to achieve (roughly) same ideal capacity optimization as a full mesh of p2p paths ?

Very? depending on topology/traffic profile

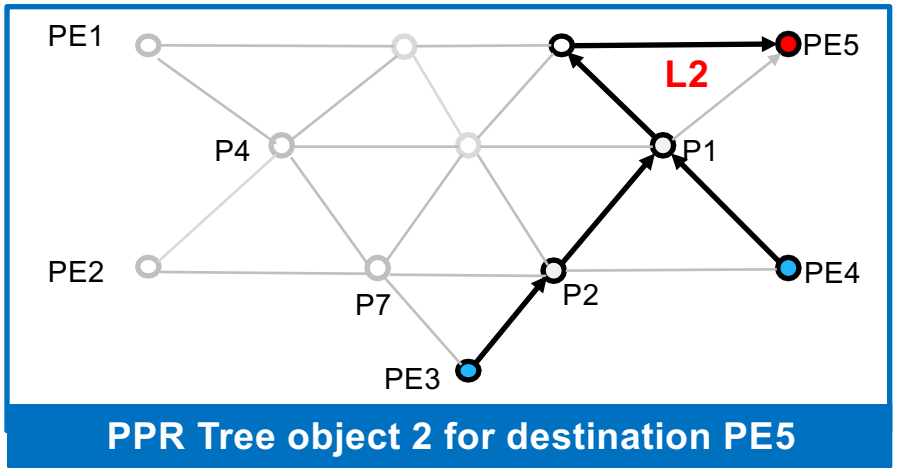
Merge p2p paths -> Trees: 100% performance

Find fewest trees for e.g.: 95% performance

Ideal algorithms for optimizing capacity vs. number of states ?



PPR Tree object 1 for destination PE5

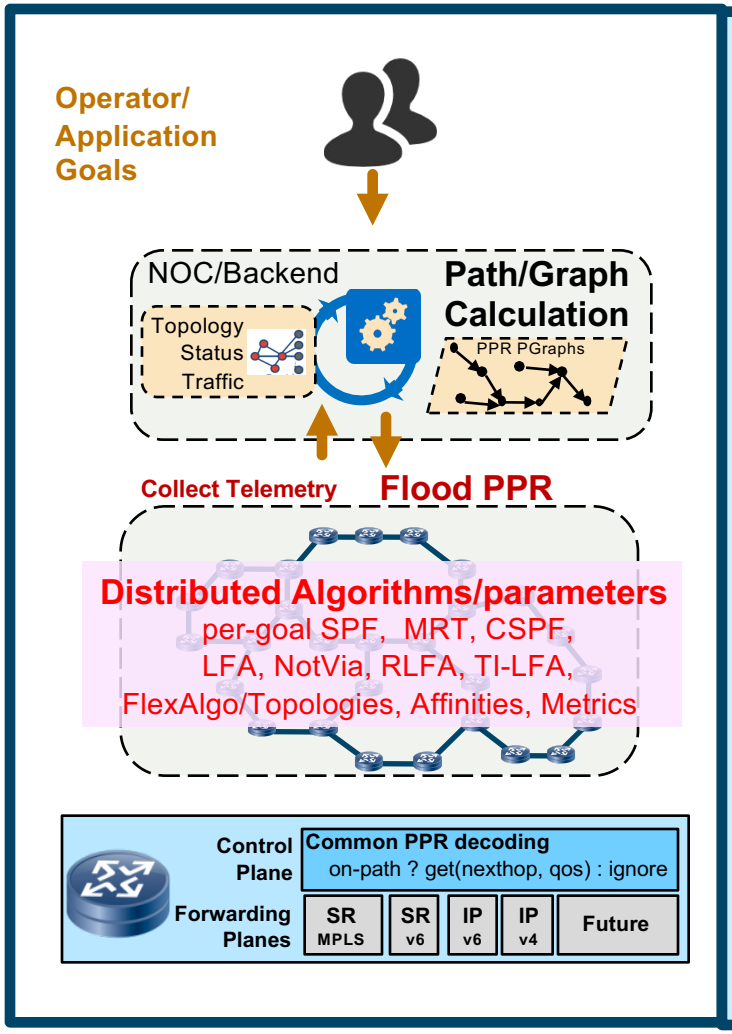


PPR Tree object 2 for destination PE5

SUMMARY AND NEXT STEPS

Distributed vs. Centralized

- Inventing, testing, deploying operating complex distributed algorithms/protocols is hard & risky.
- And costly: may need to perform complex calculation in all N nodes instead of just one central node.
- Define future minimum distributed routing/forwarding functions to enable central control for complex functions
  - SR + PPR + … ?



Operator/Application Goals

High-precision, **in-time** traffic (industrial)
Deadline, **on-time**
streaming, transportation
Minimum latency (financial market)
Lowest cost
cache preloading, "**scavenger** traffic"
High reliability, disjoined A/B services,
sub 50 msec FRR, DC apps,
media, emergency, finance ...
Time based, dynamic, temporary
Elastic Discard prioritized loss
Network capacity maximization

Applications

TP, VR, Hologram, TV-broadcast, …
Virtual Network
Internet, VNO, Slice, L2/L3VPN, ...
"Big Elephant" traffic flows

Summary and next steps

PPR Graphs is a forwarding plane agnostic routing architecture for Path+QoS engineering

PPR Graphs enables more **High Precision** centralized or decentralized Path+QoS calculations

Reduces in-network-device complexity, flexibly leverages available forwarding/QoS HW in network

Very early: many "how will it perform" questions to be simulated/measured

Wide range of interesting "how to adopt / optimize" research questions

Wide range of to be researched / solved future extensions to PPR architecture

Please get in contact with us for question and suggestions: ***Email Authors***

Thank you