IP Fast Reroute with Failure Inferencing

Junling Wang Srihari Nelakuditi
University of South Carolina, Columbia

Presenter: Sanghwan Lee
Kookmin University, Seoul, Korea

Outline

- IP Fast Reroute and Existing Approaches
- Our Approach
  - Interface Specific Forwarding → Failure Inferencing
  - Failure Inferencing based Fast Rerouting (FIFR)
- Applicability of FIFR
  - Both link and node failures
  - Symmetric and asymmetric link weights
  - Point-to-point and broadcast links
  - Intra-AS and Inter-AS failures

Fast Reroute

- Local rerouting by a node adjacent to a failure
  - Applications such as VoIP demand < 50 ms disruption
  - Global re-convergence process not fast enough
- MPLS fast reroute
  - Explicit routing of label switched paths
  - Label stacking facilitates local repair
  - Not quite scalable to configure backup paths
- IP fast reroute
  - Destination IP address based local rerouting
  - No explicit routing → more scalable

IP Fast Reroute Approaches

- Loop-free alternates
  - Select alternate next hops that do not loop back
  - May not find such a next hop even for a single failure
- Not-via addressing
  - Locally reroute a packet to a not-via address
  - Requires encapsulation and decapsulation of packets
- Multiple Routing Configurations
  - Determine a set of backup configurations/topologies
  - Route based on a different configuration upon a failure
  - Packets need to carry configuration information

Failure Inferencing based Fast Rerouting

- IP fast reroute without explicit routing/tunneling
- Employ Interface-specific forwarding
  - <incoming interface, destination> → next-hop
- Infer failures based on interface and destination
  - Find the farthest key link whose failure would cause a packet to arrive at the unusual interface along the reverse shortest path to the destination
- Precompute interface-specific forwarding tables
  - Failure inferencing is done in advance not per packet
    - Forwarding entries computed upon link state updates
  - Avoid the key link in choosing next hop for a destination

Illustration: No Failure Scenario

dest	next
A	A
C	AE
D	A
E	E
F	E

dest	next
B	B
C	C
D	D
E	B
F	B

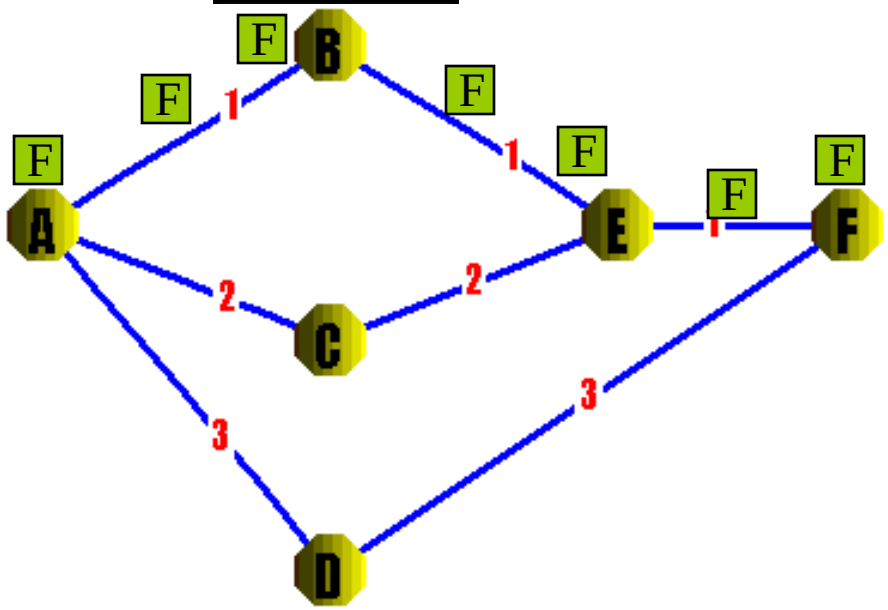


Illustration: Local Rerouting without FIFR

dest	next
A	A
C	A
D	A
E	A
F	A

dest	next
B	B
C	C
D	D
E	B
F	B

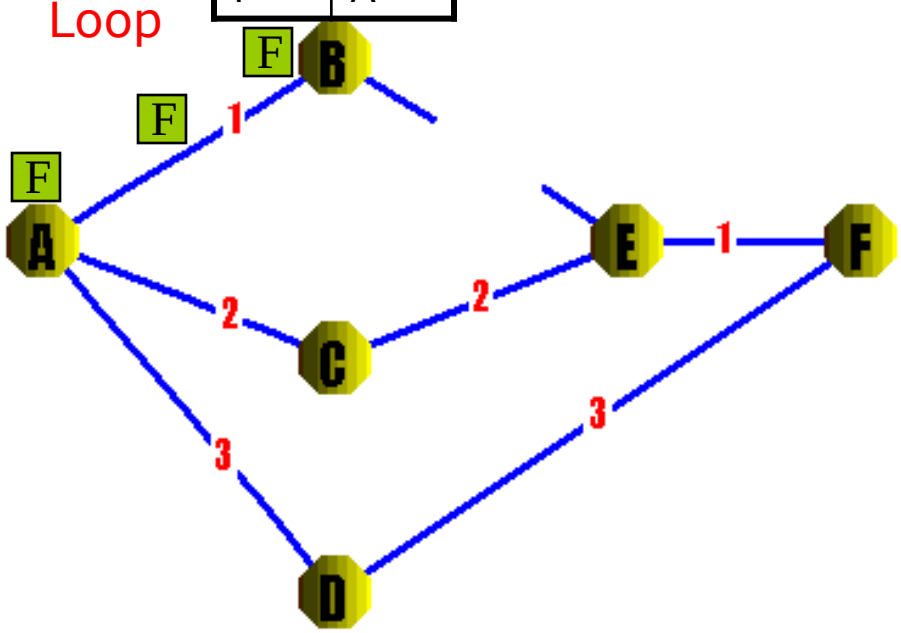
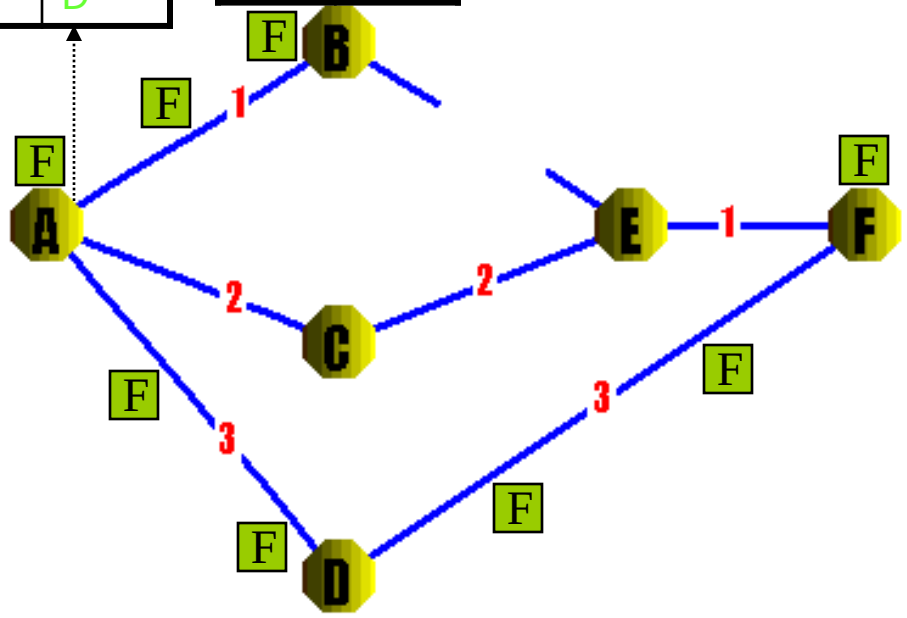


Illustration: Local Rerouting with FIFR

dest	next
B	-
C	C
D	D
E	C
F	D

dest	next
A	A
C	A
D	A
E	A
F	A

dest	next
B	B
C	C
D	D
E	B
F	B



Handling Link Failures with FIFR_L

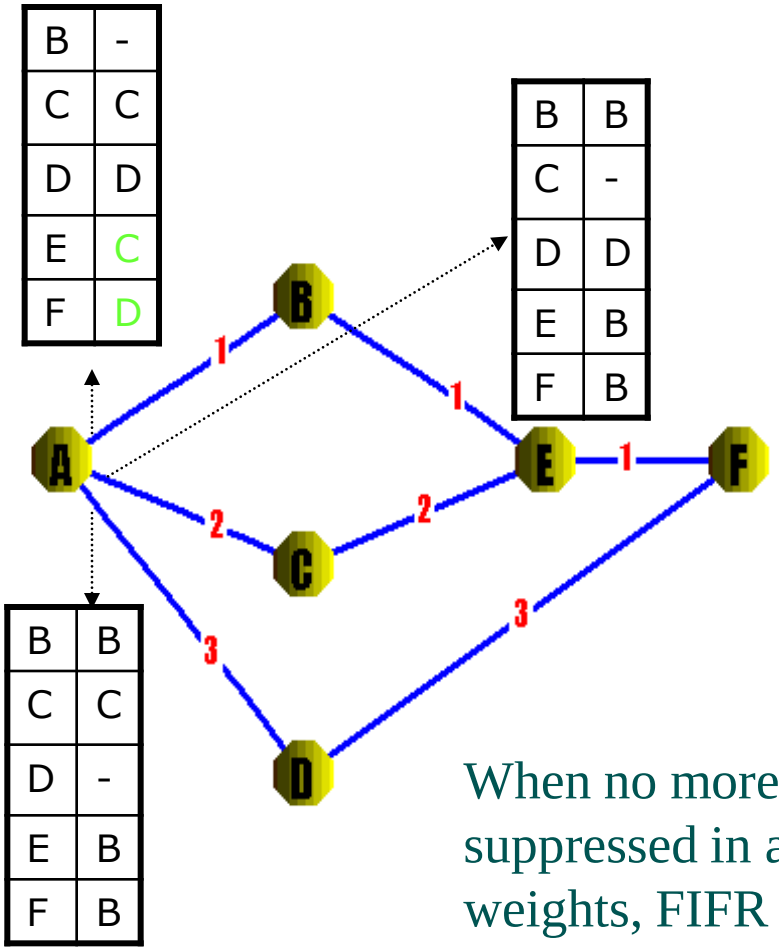
- Infer failed links from incoming interface and destination
  - $\mathcal{KL}^d_{j\to i}$: key link whose failure causes packet to d arrive at i from j
  - A link u→v is a candidate key link if
    - with u→v, j is a next hop from i to d
    - without u→v, edge j→i is along the shortest path from u to d
  - $\mathcal{KL}^d_{j\to i}$: is the farthest one from i among candidate key links
- Avoid key link in choosing the destination's next hop
  - $\mathcal{F}^d_{j\to i}$: next hops to d from i when packet arrives at i from j

$$\mathcal{F}^d_{j\to i} = N(d, \texttt{SPT}(i, \mathcal{V}, \mathcal{E} \setminus \{\mathcal{KL}^d_{j\to i}\}))$$

- Avoid the adjacent link for computing the destination's back hop
  - $\mathcal{B}^d_{i\to j}$: back hops to d from i when the link to next hop j is down

$$\mathcal{B}^d_{i\to j} = N(d, \texttt{SPT}(i, \mathcal{V}, \mathcal{E} \setminus \{i{-}j\}))$$

Illustration: Key Links Computation



$K^{C}_{B\to A} = \{\}$

$K^{D}_{B\to A} = \{\}$

$K^{E}_{B\to A} = \{\text{B-E}\}$

$K^{F}_{B\to A} = \{\text{E-F}\}$

When no more than one link failure is suppressed in a network with symmetric weights, FIFR always forwards successfully to a destination if a path to it exists

Handling Node Failures with $FIFR_N$

- Infer failed nodes from incoming interface and destination
  - $\mathcal{KN}^d_{j\to i}$: key node whose failure causes packet to d arrive at i from j
  - A node v is a candidate key node if
    - With v, j is a next hop from i to d
    - without v, edge j→i is along the shortest path from parent of v to d
  - $\mathcal{KN}^d_{j\to i}$ is the farthest one from i among candidate key nodes
- Avoid key node in choosing the destination's next hop
  - $\mathcal{F}^d_{j\to i}$: next hops to d from i when packet arrives at i from j

$$\mathcal{F}^d_{j\to i} = N(d, \mathtt{SPT}(i, \mathcal{V} \setminus \{\mathcal{KN}^d_{j\to i}\}, \mathcal{E} \setminus E(\mathcal{KN}^d_{j\to i})))$$

- Avoid the adjacent node choosing the destination's back hop
  - $\mathcal{B}^d_{i\to j}$: back hops to d from i when the next hop node j is down

$$\mathcal{B}^d_{i\to j} = N(d, \mathtt{SPT}(i, \mathcal{V} \setminus \{j\}, \mathcal{E} \setminus E(j)))$$

Handling Link and Node Failures with FIFR

- Both FIFR_L and FIFR_N have limitations
  - FIFR_L may cause forwarding loops when a node fails
  - FIFR_N may drop packets when a link fails
    - Adjacent to a partitioning node or destination
- Protection against any single failure without loops or drops
  - Treat an adjacent failure as a node failure in general

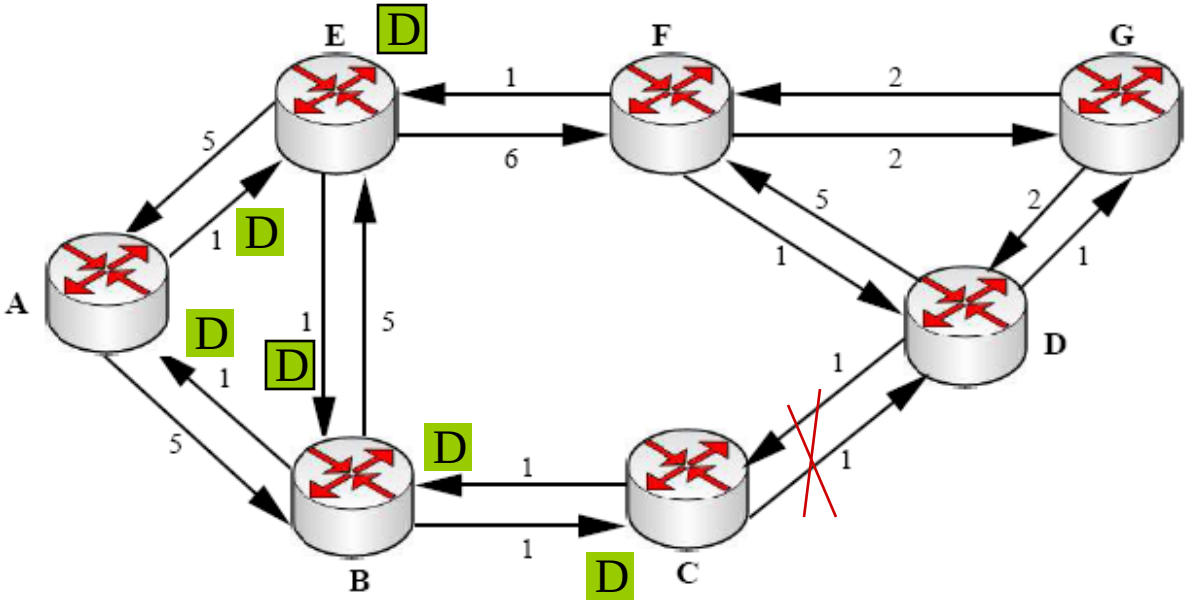
    $$\mathcal{B}^d_{i\rightarrow j} = N(d, \texttt{SPT}(i, \mathcal{V} \setminus \{j\}, \mathcal{E} \setminus E(j)))$$
  - If destination unreachable, treat it as a link failure

    $$\mathcal{B}^d_{i\rightarrow j} = N(d, \texttt{SPT}(i, \mathcal{V}, \mathcal{E} \setminus \{i\text{–}j\}))$$
    - Encapsulate the packet with next hop j as destination
      - Avoid forwarding loop in case j is indeed down
  - Non-adjacent routers infer both key nodes and key links
    - If $\mathcal{KN}^d_{j\rightarrow i}$ is empty $\quad \mathcal{F}^d_{j\rightarrow i} = N(d, \texttt{SPT}(i, \mathcal{V}, \mathcal{E} \setminus \{\mathcal{KL}^d_{j\rightarrow i}\}))$
    - Else $\quad \mathcal{F}^d_{j\rightarrow i} = N(d, \texttt{SPT}(i, \mathcal{V} \setminus \{\mathcal{KN}^u_{j\rightarrow i}\}, \mathcal{E} \setminus E(\mathcal{KN}^u_{j\rightarrow i})))$

Applicability of FIFR

- Assumptions so far
  - Links are point-to-point
  - Link weights are symmetric
  - Failures are within an AS
- This paper extends FIFR to
  - Asymmetric link weights
  - Multi-access links
  - Inter-AS failures
- FIFR still requires that
  - Links are bidirectional
  - At most a single failure is suppressed

FIFR with Asymmetric Link Weights

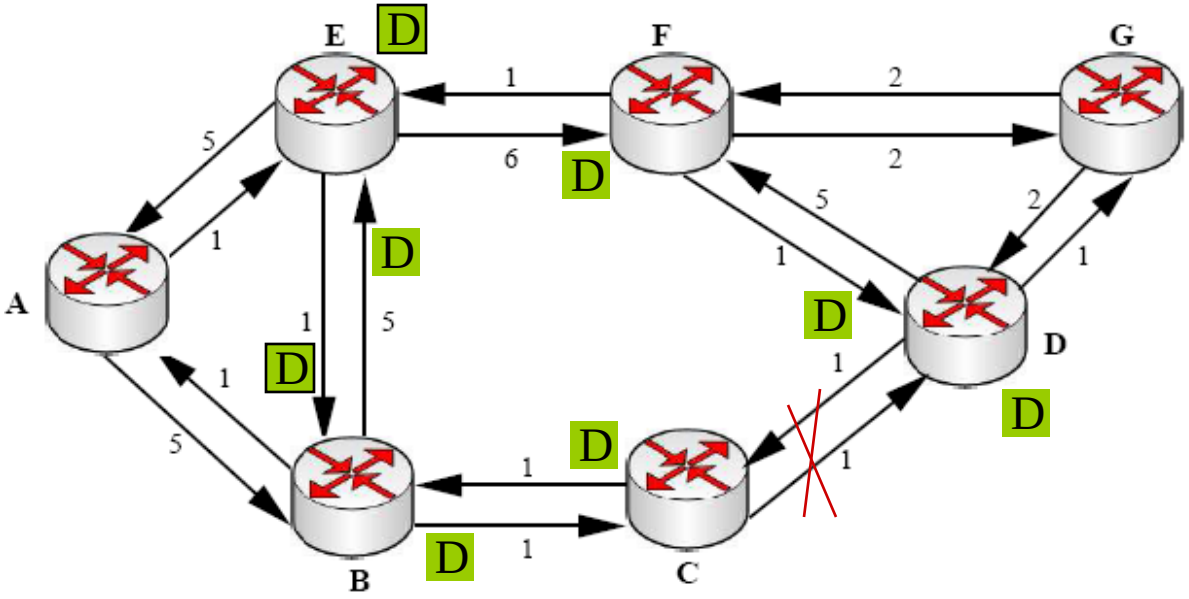


Forwarding Loop E→B→C→B→A→E→B→ …

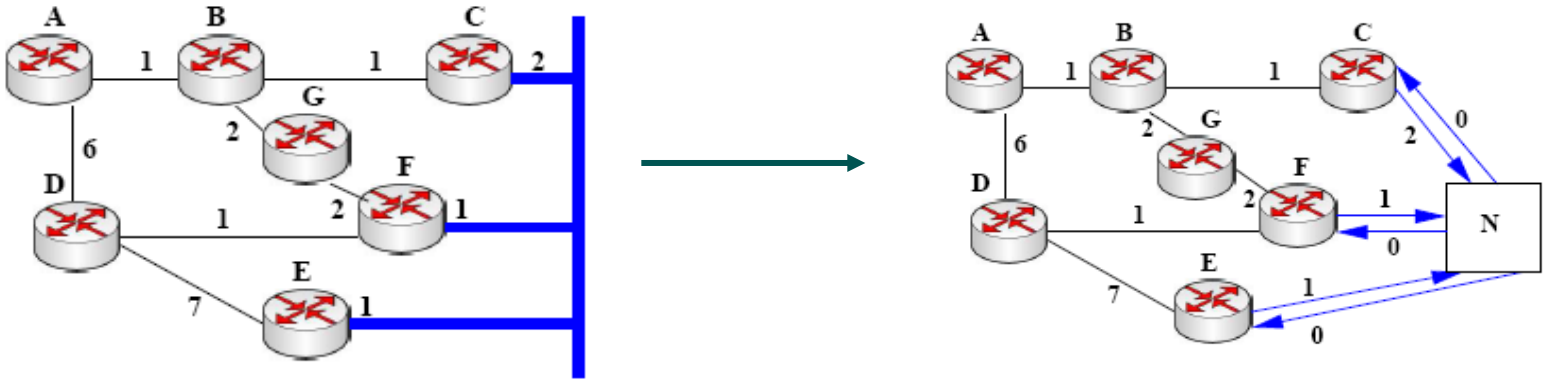
Handling Asymmetric Weights with FIFR

- When a link/node adjacent to s fails
  - Reroute a packet from s to d along rrSP(s,d)
    - rrSP(s,d): reverse of the shortest path from d to s
    - rrSP ≡ SP in networks with symmetric link weights
- Infer key nodes (similarly links) using rrSP
  - A node v is a candidate w.r.t. j→i and d if
    - With v, j is the next hop from i to d
    - Without v, rrSP(parent(v),d) contains edge j→i
  - Key node is still the candidate closest to d

Illustration: Handling Asymmetric Weights



Handling Multi-Access Links

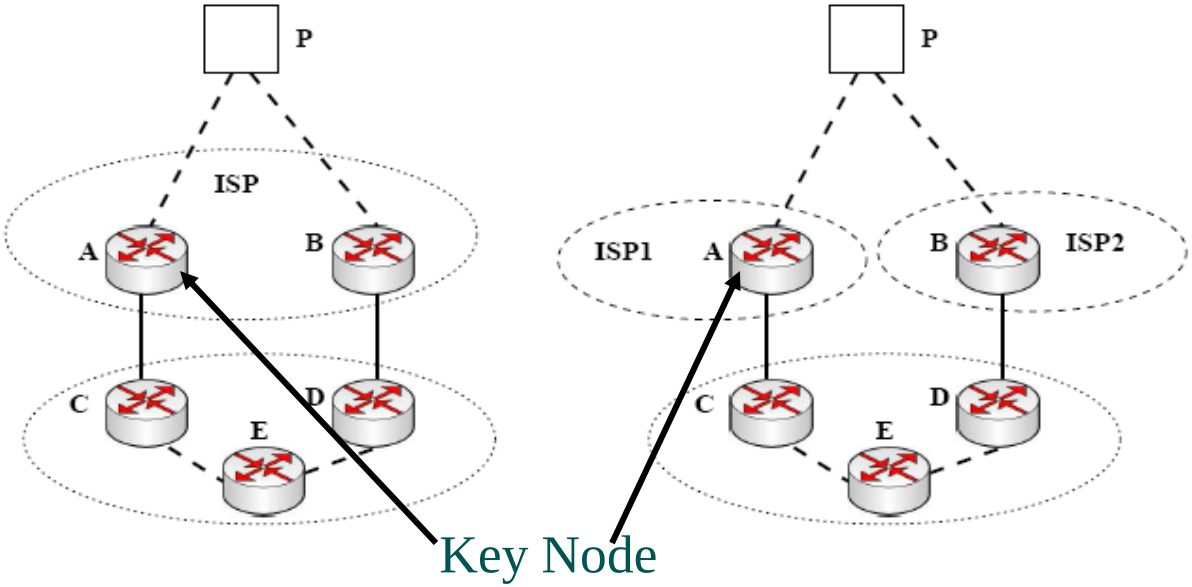


Model multi-access link as a virtual node

- Inference of a LAN failure
  - Treat it as the failure of the virtual node
- Inference of a LAN router failure
  - If parent of a node is virtual, consider grand parent as parent

Handling Inter-AS Failures

- Make FIFR aware of at least an egress apart from primary
- Assign IGP costs to virtual links from egresses to destination
- Apply FIFR approach on the resulting topological view



FIFR can automatically switch to backup egress

Summary of FIFR

- Protects against any single failures
  - Intra-AS or inter-AS
  - Link or node
- Suitable for networks with
  - Symmetric or asymmetric link weights
  - Point-to-point or multi-access links
- Requires interface-specific forwarding
  - Two forwarding entries per destination
  - $O(|E|\log^2|V|)$ to compute forwarding entries