Local Fast Rerouting with Low Congestion: A Randomized Approach

Gregor Bankhamer [1] Robert Elsässer [1] Stefan Schmid [2]

[1]Department of Computer Sciences
University of Salzburg
Austria

[2]Faculty of Computer Science
University of Vienna
Austria

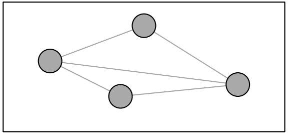
Introduction - Failover Routing

- Mission-critical networks require fast reaction to link failures
- Fast rerouting mechanisms executing in the *data plane*
- First line of defense

Introduction - Failover Routing

- Mission-critical networks require fast reaction to link failures
- Fast rerouting mechanisms executing in the *data plane*
- First line of defense

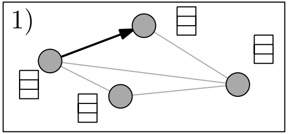
Concept



Introduction - Failover Routing

+ Mission-critical networks require fast reaction to link failures
+ Fast rerouting mechanisms executing in the *data plane*
+ First line of defense

Concept

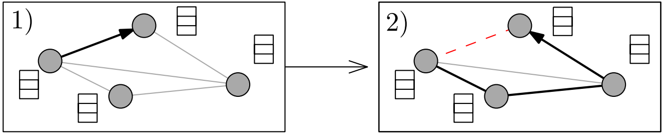


1. Incorporate failover paths (alternate paths) into forwarding table

Introduction - Failover Routing

- Mission-critical networks require fast reaction to link failures
- Fast rerouting mechanisms executing in the *data plane*
- First line of defense

Concept

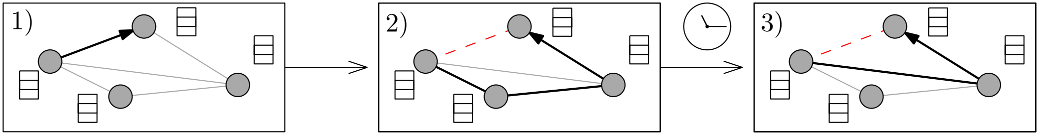


1. Incorporate failover paths (alternate paths) into forwarding table
2. Fast reaction in case of failures

Introduction - Failover Routing

- Mission-critical networks require fast reaction to link failures
- Fast rerouting mechanisms executing in the *data plane*
- First line of defense

Concept

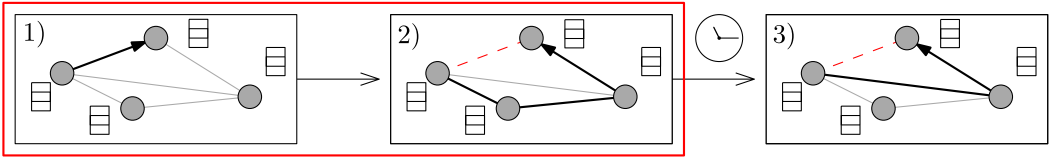


1. Incorporate failover paths (alternate paths) into forwarding table
2. Fast reaction in case of failures
3. Re-configure tables via control plane later on

Introduction - Failover Routing

+ Mission-critical networks require fast reaction to link failures
+ Fast rerouting mechanisms executing in the *data plane*
+ First line of defense

Concept



1. Incorporate failover paths (alternate paths) into forwarding table
2. Fast reaction in case of failures
3. Re-configure tables via control plane later on

Local Failover Routing - Description

Routing Problem

- Network of routers/switches. Deliver packets from source to destination
- **Desired:** Low amount of required hops and congestion

Local Failover Routing - Description

Routing Problem

- Network of routers/switches. Deliver packets from source to destination
- **Desired:** Low amount of required hops and congestion

Local Failover Protocol

- For each node v with neighborhood $\Gamma(v)$ pre-computable function

$$f_v : (2^{\Gamma(v)} \times \mathcal{P}) \to \Gamma(v)$$

Next hop

Set of unreachable neighbors

Packet header information (e.g. dest address)

Local Failover Routing - Description

Routing Problem

- Network of routers/switches. Deliver packets from source to destination
- **Desired:** Low amount of required hops and congestion

Local Failover Protocol

- For each node v with neighborhood $\Gamma(v)$ pre-computable function

$$f_v : (2^{\Gamma(v)} \times \mathcal{P}) \to \Gamma(v)$$

$2^{\Gamma(v)}$: Set of unreachable neighbors

$\mathcal{P}$: Packet header information (e.g. dest address)

$\Gamma(v)$: Next hop

Challenges

- Fast forwarding ruleset ; depending on *local* information only
- Low congestion hard (or impossible) to achieve under multiple link failures

Related Work

Existing Local Failover Protocols

- Multiple deterministic approaches
- Randomized protocol [Chiesa et al., ICALP 2016]
  - k-connected networks, arborescence cover, packet-based communication

Related Work

Existing Local Failover Protocols

- Multiple deterministic approaches
- Randomized protocol [Chiesa et al., ICALP 2016]
  - k-connected networks, arborescence cover, packet-based communication

- Existing results either do not account for load or are deterministic

Related Work

Existing Local Failover Protocols

- Multiple deterministic approaches
- Randomized protocol [Chiesa et al., ICALP 2016]
  - k-connected networks, arborescence cover, packet-based communication

- Existing results either do not account for load or are deterministic

Negative Result

- Congestion lower bound for deterministic local failover protocols [Borokhovich and Schmid, OPODIS 2013]

Model and Setting

Environment

- *Complete* undirected Graph $G = (V, E)$ with $|V| = n$.
  - May be generalized with arborescences or embedding
  - High degree and low diameter

Model and Setting

Environment

- *Complete* undirected Graph $G = (V, E)$ with $|V| = n$.
  - May be generalized with arborescences or embedding
  - High degree and low diameter

Communication Model

- Flow-based communication
- Consecutive stream of packets sent by source $s \in V$ to destination $d \in V$.

Model and Setting

Environment

- *Complete* undirected Graph $G = (V, E)$ with $|V| = n$.
  - May be generalized with arborescences or embedding
  - High degree and low diameter

Communication Model

- Flow-based communication
- Consecutive stream of packets sent by source $s \in V$ to destination $d \in V$.

Challenging Communication Pattern - *All-to-one Routing*

- Some destination node d
- Each node $V \setminus \{d\}$ sends out one flow targeted at d
- Commonly used in related work

Model and Setting ctd.

Powerful Adversary

+ Knows employed failover strategy
+ Knows destination d
+ Allowed to fail a high amount of edges – up to $\Omega(n)$.

Deterministic Case Lower Bound

Theorem (Borokhovich and Schmid, OPODIS 2013)

Consider any local destination-based failover scheme in a clique graph. There exists a set of φ (edge) failures ($0 < \varphi < n$) that results in a link load of at least φ.

Deterministic Case Lower Bound

Theorem (Borokhovich and Schmid, OPODIS 2013)

Consider any local destination-based failover scheme in a clique graph. There exists a set of φ *(edge) failures* $(0 < \varphi < n)$ *that results in a link load of at least* φ.

Different Rulesets

- Borokhovich and Schmid also give a $\sqrt{\varphi}$ lower bound if ruleset includes source adress.
- Can be extended to also account for *hop*-count
- Adversary can create a load of $\Omega(\sqrt{n})$ by destroying $\mathcal{O}(n)$ links.

Our Solution - Randomization

Goal: Break this bound and reduce the possible congestion significantly

Randomization

- **Observation:** Each failover protocol has bad failure scenarios (due *locality*)
- **Idea:** Make these scenarios *unlikely to occur*!
- Results achieved *with high probability* (w.h.p.; at least prob. $1 - n^{-1}$)

Our Solution - Randomization

Goal: Break this bound and reduce the possible congestion significantly

Randomization

- **Observation:** Each failover protocol has bad failure scenarios (due *locality*)
- **Idea:** Make these scenarios *unlikely to occur*!
- Results achieved *with high probability* (w.h.p.; at least prob. $1 - n^{-1}$)

Adapted (oblivious) Adversary

- May still know the protocol and *all-to-one* routing target d
- *Cannot* know the nodes generated random bits or measure the network load

Our Results - Overview

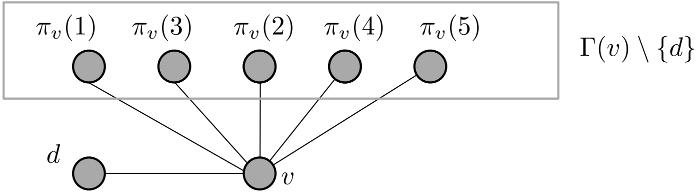
	3-Permutations	*Intervals*	*Shared-Permutations*
Rule Set	Destination + Hop	Destination	Destination + Hop [1]
Resilience	$\Theta(n)$	$\Theta(n/\log n)$	$\Theta(n)$
Congestion	$\mathcal{O}(\log^2 n \cdot \log\log n)$	$\mathcal{O}(\log n \cdot \log\log n)$	$\mathcal{O}(\sqrt{\log n})$
Hops	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$
Bits	$\mathcal{O}(\log^2 n)$	$\mathcal{O}(\log^2 n)$	$\mathcal{O}(\log^3 n)$
Shared Data	×	×	✓

+ Congestion: Maximum number of flows crossing any node $v \in V \setminus \{d\}$
+ Number of failed edges up to resilience
+ Deterministic protocols would allow the adversary to induce a load of $\Omega(n/\log n)$ or $\Omega(\sqrt{n})$ respectively.

[1] may be raised to some arbitrary value of $\mathcal{O}(\log\log n)$ bits

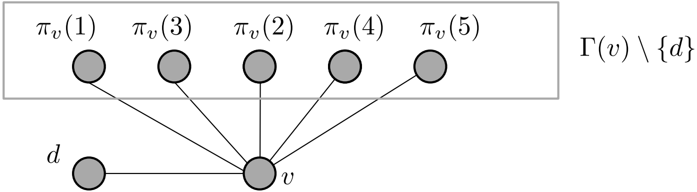
Baseline Idea - Permutation Based Failover Routing

- Domain of failover function f_v grows exponentially with $|\Gamma(v)|$
- Equip v with permutation π_v of neighbors $\Gamma(v) \setminus \{d\}$



Baseline Idea - Permutation Based Failover Routing

- Domain of failover function f_v grows exponentially with $|\Gamma(v)|$
- Equip v with permutation π_v of neighbors $\Gamma(v) \setminus \{d\}$



Basic Permutation-Based Protocol (POV of node v)

Input: A packet p with destination d

1: **if** (v, d) is intact **then** forward p to d and **return** ▷ Default route

2: **else** forward p over edge with smallest i s.t. $(v, \pi_v(i))$ is not failed

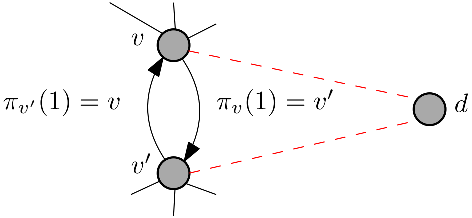
Permutation Based Routing - Observation

+ Randomized approach: Select π_v uniformly at random at each node v

Permutation Based Routing - Observation

+ Randomized approach: Select π_v uniformly at random at each node v

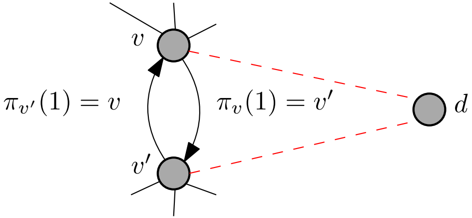
Bad News: Forwarding Loops



Permutation Based Routing - Observation

+ Randomized approach: Select π_v uniformly at random at each node v

Bad News: Forwarding Loops



Good News: All-to-one Routing

If adversary fails $\alpha \cdot n$ edges (for constant $0 < \alpha < 1$), then w.h.p.

+ All nodes *not* involved in a forwarding loop receive $\mathcal{O}(\log n \cdot \log\log n)$ flows
+ Packets *not* stuck in a loop reach d in $\mathcal{O}(\log n)$ hops

Analysis Idea – Describe Flows by Graph

- Assume only edges of the form (v, d) are failed (up to $\alpha \cdot n$)

Analysis Idea – Describe Flows by Graph

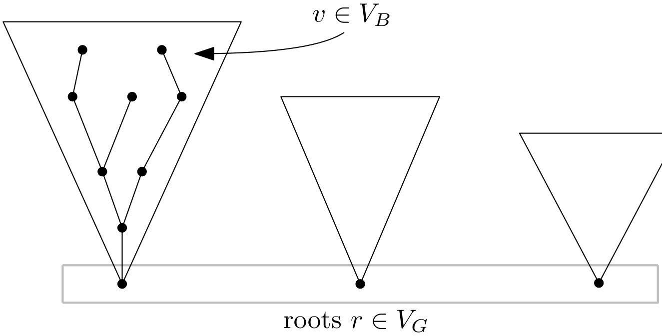
- Assume only edges of the form (v, d) are failed (up to $\alpha \cdot n$)
- **Remember:** If (v, d) failed first forwarding alternative $(v, \pi_v(1))$

Analysis Idea – Describe Flows by Graph

- Assume only edges of the form (v, d) are failed (up to $\alpha \cdot n$)
- **Remember:** If (v, d) failed first forwarding alternative $(v, \pi_v(1))$
- Di-Graph $P = (V', E')$ with $V' = V \setminus \{d\}$ and $E' = \{\ (v, \pi_v(1)) \mid v \in V_B\}$

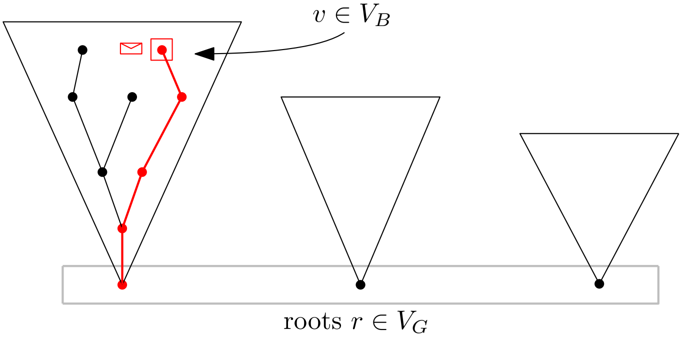
Analysis Idea – Describe Flows by Graph

- Assume only edges of the form (v, d) are failed (up to $\alpha \cdot n$)
- **Remember:** If (v, d) failed first forwarding alternative $(v, \pi_v(1))$
- Di-Graph $P = (V', E')$ with $V' = V \setminus \{d\}$ and $E' = \{\ (v, \pi_v(1)) \mid v \in V_B\}$



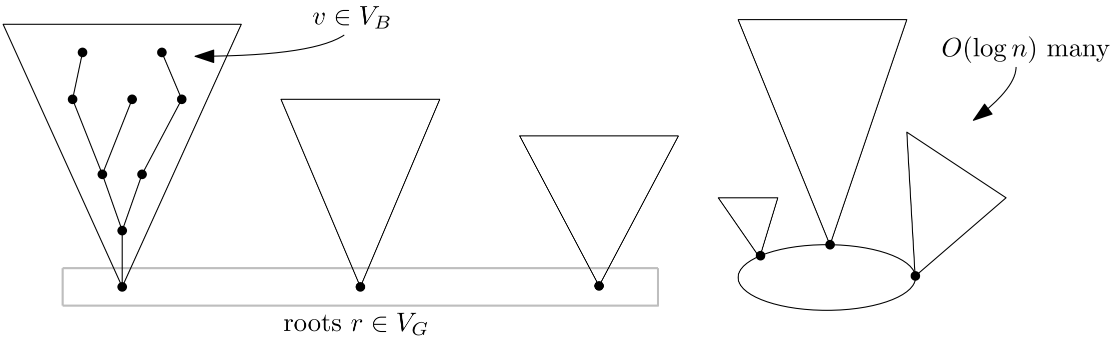
Analysis Idea – Describe Flows by Graph

- Assume only edges of the form (v, d) are failed (up to $\alpha \cdot n$)
- **Remember:** If (v, d) failed first forwarding alternative $(v, \pi_v(1))$
- Di-Graph $P = (V', E')$ with $V' = V \setminus \{d\}$ and $E' = \{\ (v, \pi_v(1)) \mid v \in V_B\}$



Analysis Idea – Describe Flows by Graph

- Assume only edges of the form (v, d) are failed (up to $\alpha \cdot n$)
- **Remember:** If (v, d) failed first forwarding alternative $(v, \pi_v(1))$
- Di-Graph $P = (V', E')$ with $V' = V \setminus \{d\}$ and $E' = \{\ (v, \pi_v(1)) \mid v \in V_B\}$



Analysis – Tree Size

+ *Size of the tree corresponds to received flows at root!*

Analysis – Tree Size

- *Size of the tree corresponds to received flows at root!*
- Look at one root $r \in V_G$ in isolation. Edges not drawn yet
- Tree is constructed layer-by-layer: Each $v \in V_B$ selects $\pi_v(1)$ u.a.r.

Analysis – Tree Size

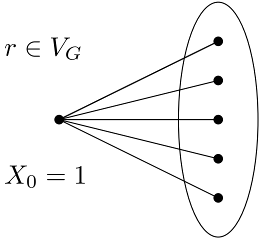
+ *Size of the tree corresponds to received flows at root!*
+ Look at one root $r \in V_G$ in isolation. Edges not drawn yet
+ Tree is constructed layer-by-layer: Each $v \in V_B$ selects $\pi_v(1)$ u.a.r.

$r \in V_G$

$X_0 = 1$

Analysis – Tree Size

- *Size of the tree corresponds to received flows at root!*
- Look at one root $r \in V_G$ in isolation. Edges not drawn yet
- Tree is constructed layer-by-layer: Each $v \in V_B$ selects $\pi_v(1)$ u.a.r.



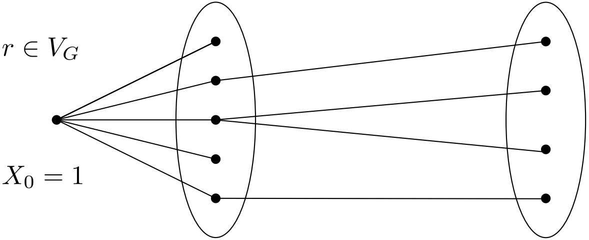
$$X_1 \sim \text{Binomial}\ (n_1, p_1)$$

$$n_1 = |V_B|$$

$$p_1 = \frac{X_1}{|V|}$$

Analysis – Tree Size

- *Size of the tree corresponds to received flows at root!*
- Look at one root $r \in V_G$ in isolation. Edges not drawn yet
- Tree is constructed layer-by-layer: Each $v \in V_B$ selects $\pi_v(1)$ u.a.r.



$X_1 \sim \text{Binomial}\ (n_1, p_1)$

$n_1 = |V_B|$

$p_1 = \frac{X_1}{|V|}$

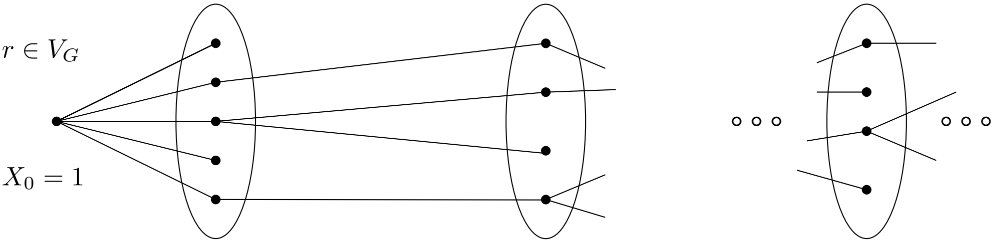
$X_2 \sim \text{Binomial}\ (n_2, p_2)$

$n_2 = |V_B| - X_1$

$p_2 = \frac{X_1}{|V| - X_0}$

Analysis – Tree Size

- *Size of the tree corresponds to received flows at root!*
- Look at one root $r \in V_G$ in isolation. Edges not drawn yet
- Tree is constructed layer-by-layer: Each $v \in V_B$ selects $\pi_v(1)$ u.a.r.



$X_1 \sim \text{Binomial}\,(n_1, p_1)$

$n_1 = |V_B|$

$p_1 = \frac{X_1}{|V|}$

$X_2 \sim \text{Binomial}\,(n_2, p_2)$

$n_2 = |V_B| - X_1$

$p_2 = \frac{X_1}{|V| - X_0}$

$X_i \sim \text{Binomial}\,(n_i, p_i)$

$n_i = |V_B| - \sum_{j=1}^{i-1} X_j$

$p_i = \frac{X_{i-1}}{|V| - \sum_{j=1}^{i-2} X_j}$

Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

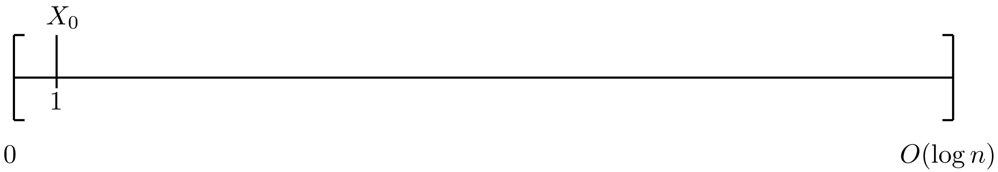
+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.

Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.

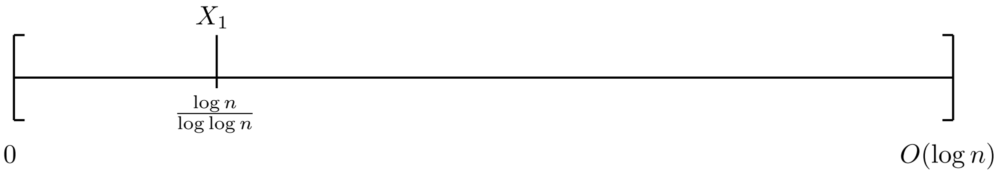


Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.

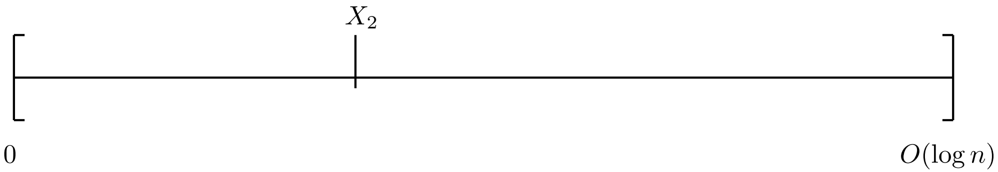


Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.

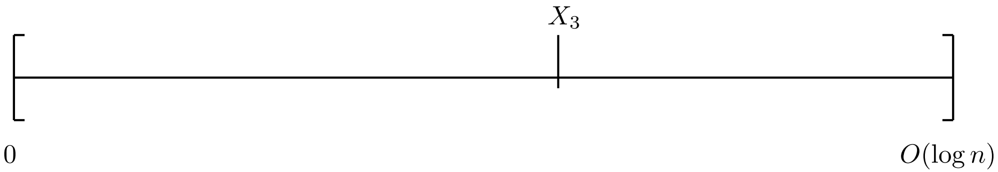


Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.

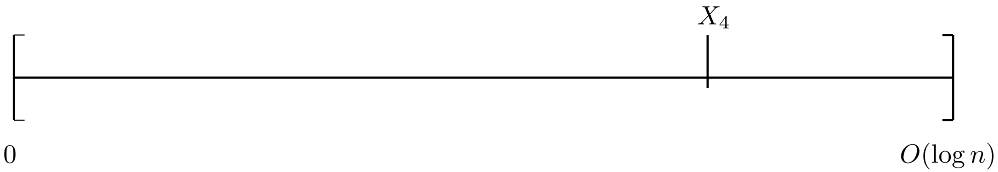


Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.

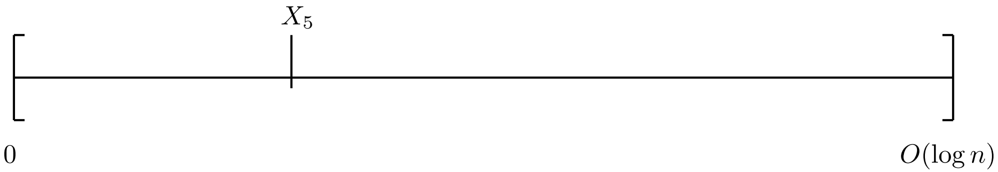


Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.

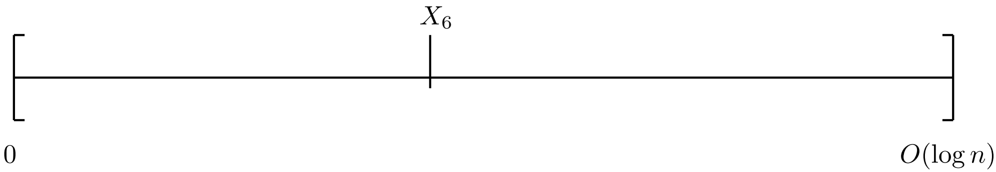


Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.

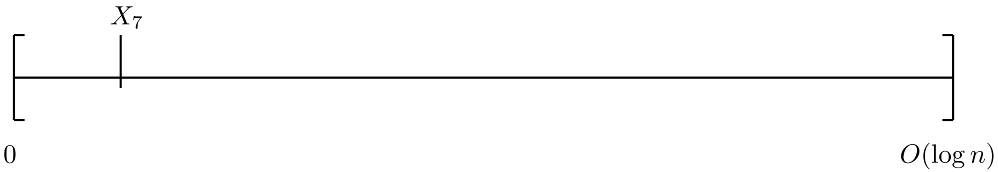


Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.

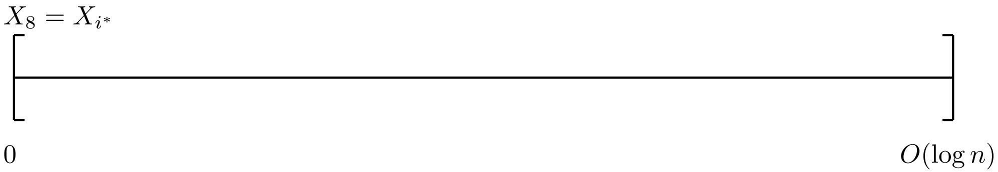


Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.

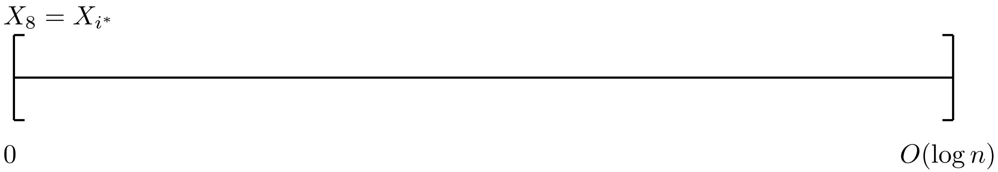


Analysis – Tree Size ctd.

For a fixed tree, we know:

1. $X_i \sim$ Binomial distribution. Depending on previously uncovered layers
2. $\text{Exp}[X_i] = n_i \cdot p_i \approx |V_B| \cdot \frac{X_{i-1}}{|V|} \leq \alpha \cdot X_{i-1}$

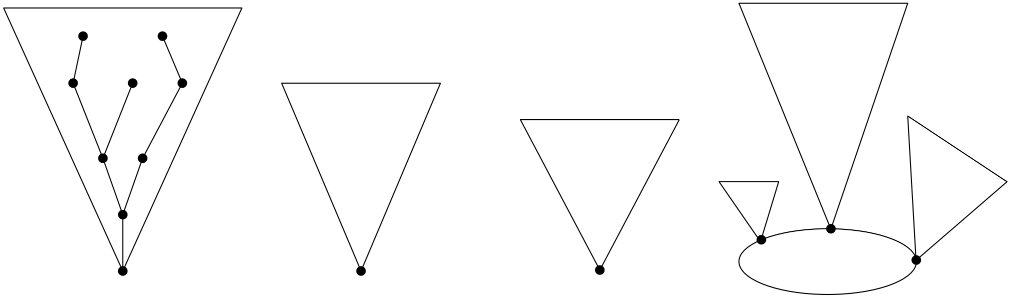
+ Sequence $\{X_i\}$ can be seen as Markov Chain or Random Walk
+ Drifts towards 0 and does not exceed $\mathcal{O}(\log n)$ w.h.p.



+ X_{i^*} hits 0 for $i^* < C_1 \log n$
+ **Technical Result:** $\sum_{i=0}^{i^*} X_i = \mathcal{O}(\log n \cdot \log\log n)$ w.h.p.

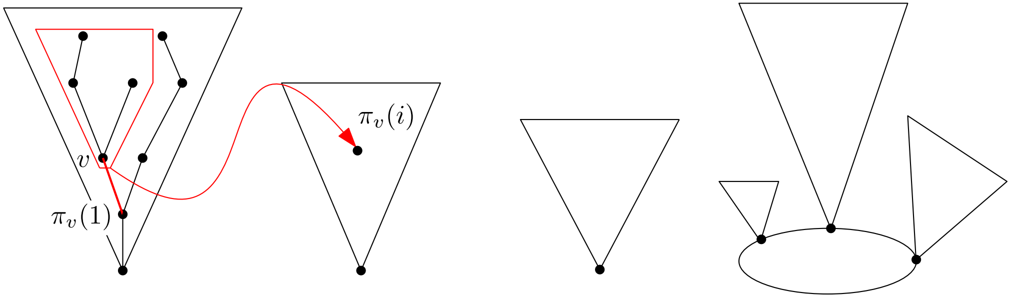
Analysis – Accounting For All Failed Edges

+ Until now: Neglected failed edges of the form $(v, \pi_v(1))$
+ Modify the graph P into P_m as follows



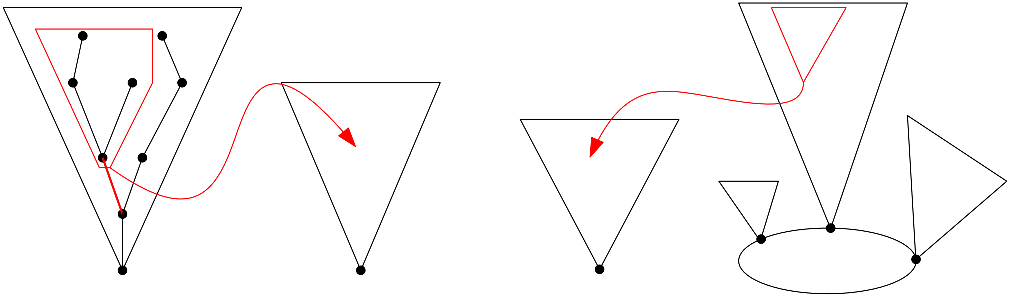
Analysis – Accounting For All Failed Edges

- Until now: Neglected failed edges of the form $(v, \pi_v(1))$
- Modify the graph P into P_m as follows



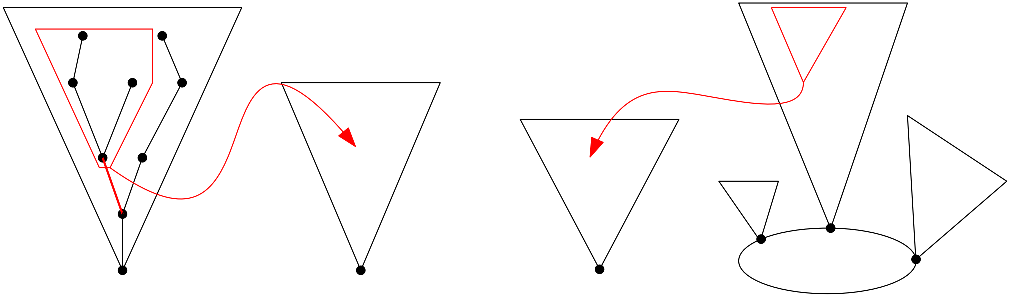
Analysis – Accounting For All Failed Edges

- Until now: Neglected failed edges of the form $(v, \pi_v(1))$
- Modify the graph P into P_m as follows



Analysis – Accounting For All Failed Edges

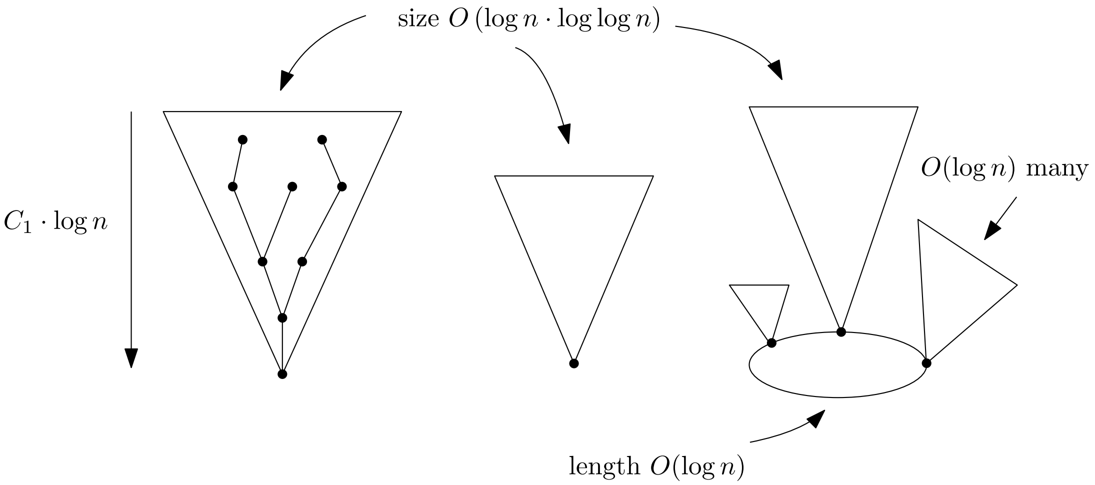
- Until now: Neglected failed edges of the form $(v, \pi_v(1))$
- Modify the graph P into P_m as follows



- Adversary does not know which edges are at pole-position of the permutations
- $\mathcal{O}(\log n)$ subtrees are relocated w.h.p. ($\mathcal{O}(1)$ to the same component)
- Height and size of trees does not change asymptotically

Analysis – Summary

- P_m describes packets' routes after $\alpha \cdot n$ edges are failed

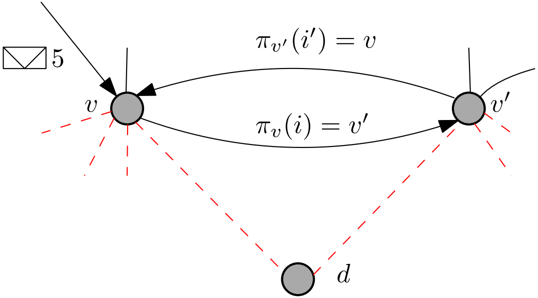


3-Permutations Protocol

- Extend the simple permutation based approach
- Deal with forwarding loops

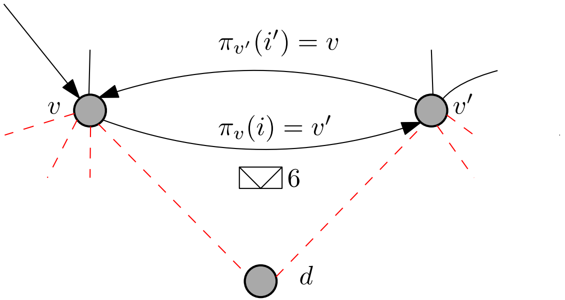
3-Permutations Protocol

- Extend the simple permutation based approach
- Deal with forwarding loops



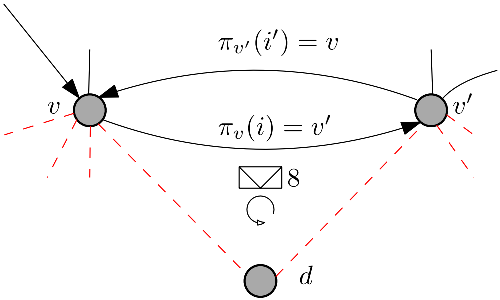
3-Permutations Protocol

- Extend the simple permutation based approach
- Deal with forwarding loops



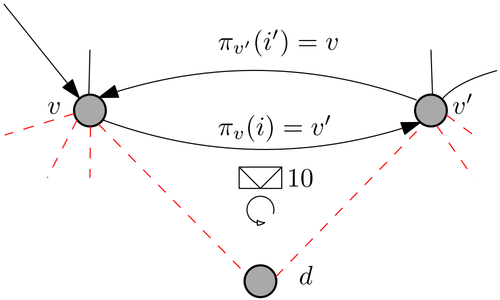
3-Permutations Protocol

+ Extend the simple permutation based approach
+ Deal with forwarding loops



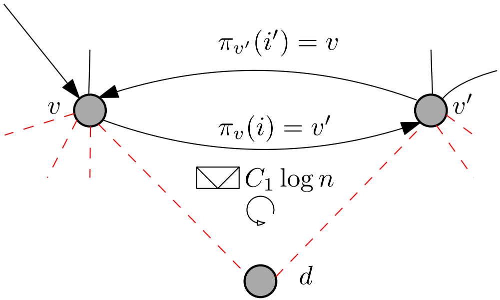
3-Permutations Protocol

- Extend the simple permutation based approach
- Deal with forwarding loops



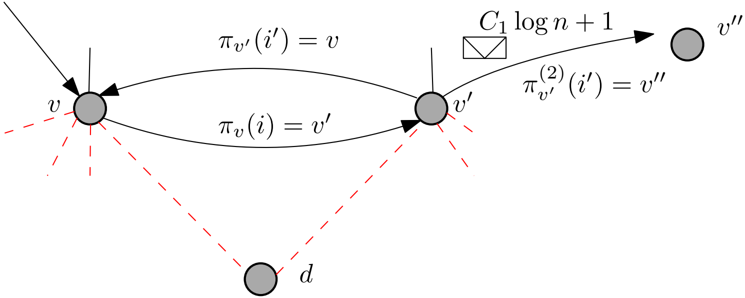
3-Permutations Protocol

+ Extend the simple permutation based approach
+ Deal with forwarding loops



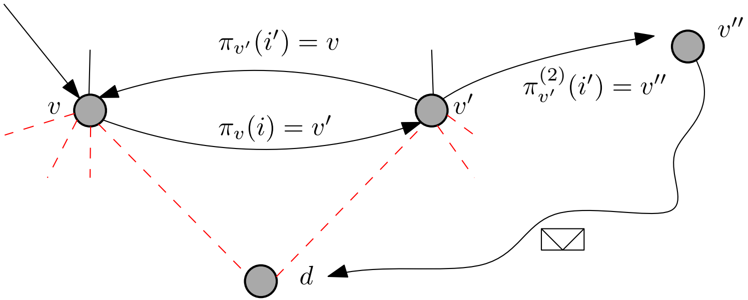
3-Permutations Protocol

- Extend the simple permutation based approach
- Deal with forwarding loops



3-Permutations Protocol

+ Extend the simple permutation based approach
+ Deal with forwarding loops



+ We know: Hop larger $C_1 \log n$ implies trapped in loop w.h.p.
+ **Caveat:** Flows travel in the cycle for $\mathcal{O}(\log n)$ hops and accumulate load

3-Permutations Protocol (POV of node v)

Input: A packet with destination d and hop count h

```
if (v, d) is intact then forward p to d and return
else if h ≤ C_1 log n then send p to first reachable node in π_v^(1)
else if h ≤ 2 · C_1 log n then send p to first reachable node in π_v^(2)
else send p to first reachable node in π_v^(3)
increase h += 1
```

3-Permutations Protocol (POV of node v)

Input: A packet with destination d and hop count h

1: **if** (v, d) is intact **then** forward p to d and **return**
2: **else if** $h \leq C_1 \log n$ **then** send p to first reachable node in $\pi_v^{(1)}$
3: **else if** $h \leq 2 \cdot C_1 \log n$ **then** send p to first reachable node in $\pi_v^{(2)}$
4: **else** send p to first reachable node in $\pi_v^{(3)}$
5: increase $h \mathrel{+}= 1$

Result (3-Permutations)

+ Adversary fails up to $\alpha \cdot n$ edges (any constant $0 < \alpha < 1$)
+ All-to-one routing to any destination d.

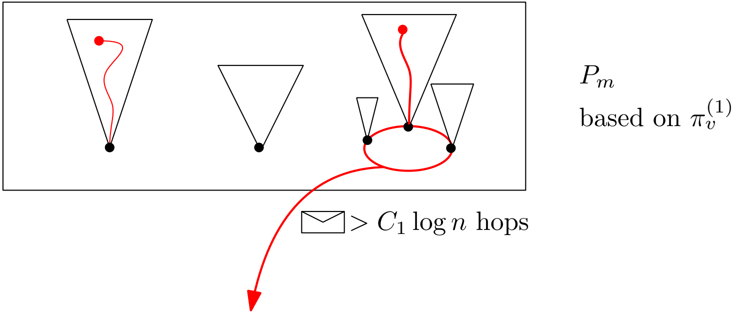
1. $\mathcal{O}(\log n)$ hops per packet
2. $\mathcal{O}(\log \cdot \log \log n)$ load at all but $\mathcal{O}(\log^2 n)$ nodes
3. $\mathcal{O}(\log^2 \cdot \log \log n)$ load at remaining nodes *w.h.p.*

3-Permutations – Analysis Sketch

+ For packets with $< C_1 \log n$ hops: Behavior same as Simple Permutation Based
+ Graph P_m based on $\pi_v^{(1)}$ describes first $C_1 \log n$ hops

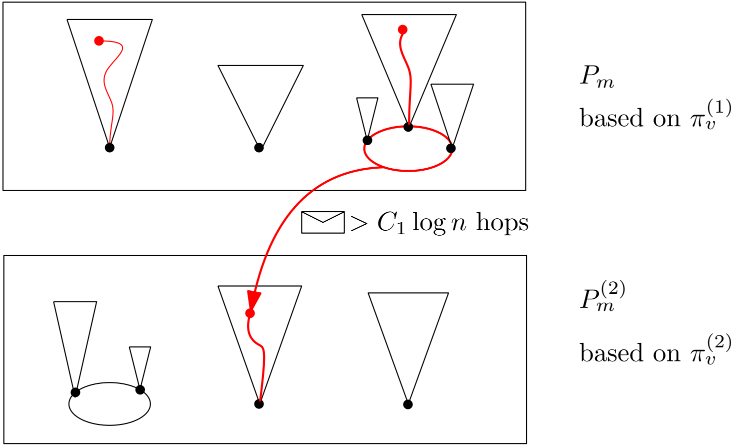
3-Permutations – Analysis Sketch

- For packets with $< C_1 \log n$ hops: Behavior same as Simple Permutation Based
- Graph P_m based on $\pi_v^{(1)}$ describes first $C_1 \log n$ hops

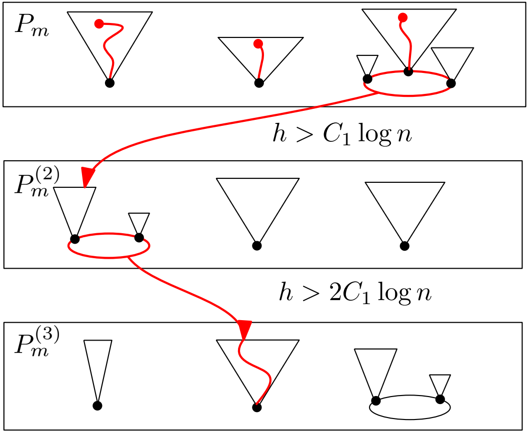


3-Permutations – Analysis Sketch

+ For packets with $< C_1 \log n$ hops: Behavior same as Simple Permutation Based
+ Graph P_m based on $\pi_v^{(1)}$ describes first $C_1 \log n$ hops



3-Permutations – Analysis Sketch ctd.



1. No packet stuck in loop in all 3 graphs $\Rightarrow$ 3 Permutations suffice
2. Every packet travels $< 3 \cdot C_1 \log n$ hops w.h.p.
3. Each node not on a cycle in any graph receives $\mathcal{O}(\log n \cdot \log \log n)$ load
4. Flow might spin $\Theta(\log n)$ times before "leaving" the loop $\Rightarrow$ $O(\log n)$ factor load amplification

Intervals Protocol

+ Again extend upon simple permutation-based approach
+ Avoid temporary cycles w.h.p.
+ Only relies on destination address

Intervals Protocol

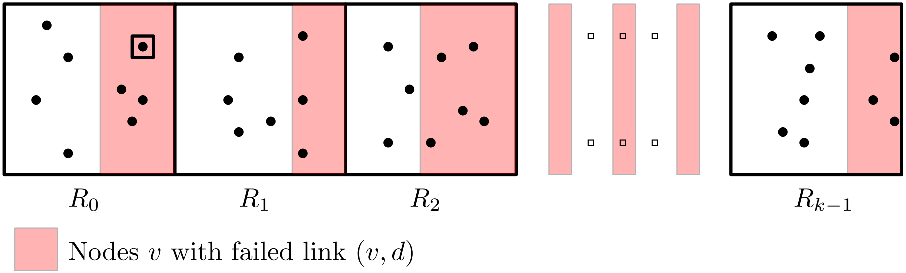
- Again extend upon simple permutation-based approach
- Avoid temporary cycles w.h.p.
- Only relies on destination address

Concept

- Partition the nodes V into $k = \mathcal{O}(\log n)$ sets $R_0, \ldots, R_{k-1} \subseteq V$
- Each $|R_i| \approx n/(4 \log_{1/\alpha} n) = \mathcal{O}(n/\log n)$ for constant $0 < \alpha < 1$.
- (Random) failover permutation π_v of $v \in R_i$ consists nodes in $R_{(i+1) \mod k}$ only
- Basic Permutation Routing Protocol using this set of permutations π_v.

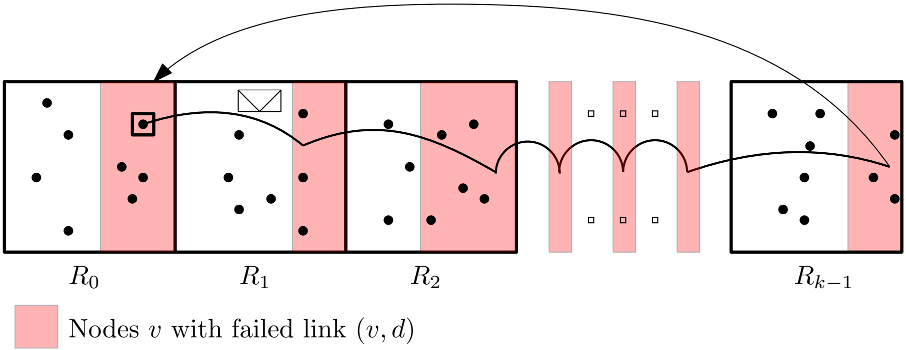
Intervals Protocol - Avoiding Temporary Cycles

+ Assume adversary may destroy $\alpha \cdot |R_i| = \mathcal{O}(n/\log n)$ edges per partition



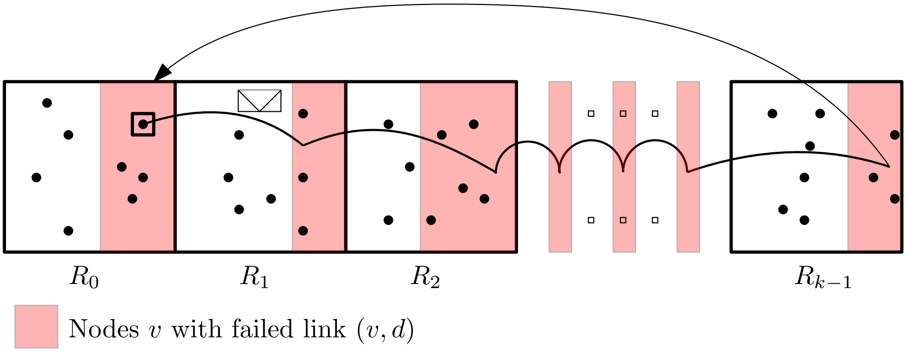
Intervals Protocol - Avoiding Temporary Cycles

+ Assume adversary may destroy $\alpha \cdot |R_i| = \mathcal{O}(n/\log n)$ edges per partition



Intervals Protocol - Avoiding Temporary Cycles

- Assume adversary may destroy $\alpha \cdot |R_i| = \mathcal{O}(n/\log n)$ edges per partition



- Packet moves to k consecutive "bad" nodes with probability $\alpha^k \ll \mathcal{O}(1/n)$

Intervals Protocol (POV of node v)

Input: A packet p with destination d

if (v, d) is intact **then** forward p to d and **return**
else send p to first directly reachable node in π_v

Intervals Protocol (POV of node v)

Input: A packet p with destination d

1: **if** (v, d) is intact **then** forward p to d and **return**

2: **else** send p to first directly reachable node in π_v

Result (Intervals)

+ Adversary fails up to $\alpha \cdot |R_i|$ edges in each partition R_i (const. $0 < \alpha < 1$)
+ All-to-one-routing to any destination d

1. $\mathcal{O}(\log n)$ hops
2. $\mathcal{O}(\log n \cdot \log\log n)$ load on all nodes *w.h.p.*

+ Maximum resilience of $(1/e) \cdot (n/\ln n)$ for $\alpha = 1/e$

Intervals Protocol – Analysis overview

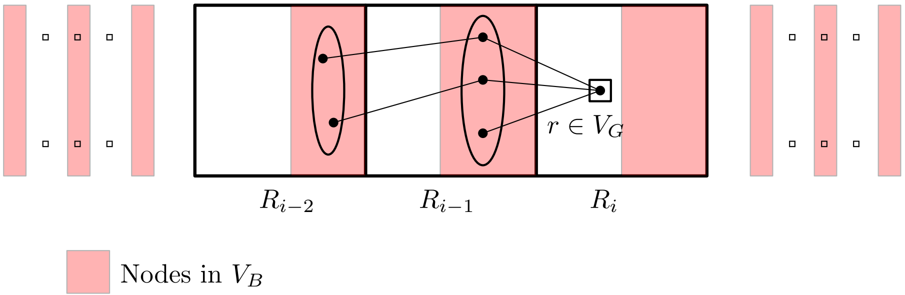
- **We know:** Forwarding loops are avoided by construction w.h.p.

Intervals Protocol – Analysis overview

+ **We know:** Forwarding loops are avoided by construction w.h.p.

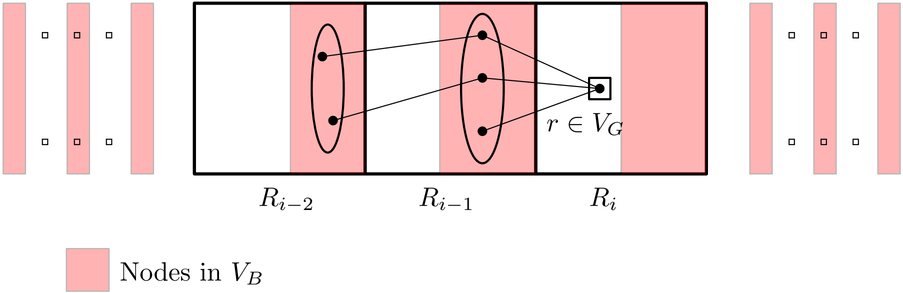
Intervals Protocol – Analysis overview

+ **We know:** Forwarding loops are avoided by construction w.h.p.



Intervals Protocol – Analysis overview

+ **We know:** Forwarding loops are avoided by construction w.h.p.



+ Size of tree $\mathcal{O}(\log n \cdot \log \log n)$
+ Height $\mathcal{O}(\log n)$

Shared-Permutations Protocol

- **Goal:** Further decrease maximum load
- Introduce additional type of permutation

Shared-Permutations Protocol

- **Goal:** Further decrease maximum load
- Introduce additional type of permutation

Concept

- *Globally shared* (random) permutations π_i^G of all nodes $V \setminus \{d\}$ $(0 \leq i \leq C_2 \log n)$

Input: A packet with destination d and hop h arriving at v

1: **if** (v, d) is intact **then** forward p to d and **return**

2: **else** forward p to the successor w of v in π_h^G

Shared-Permutations Protocol

- **Goal:** Further decrease maximum load
- Introduce additional type of permutation

Concept

- *Globally shared* (random) permutations π_i^G of all nodes $V \setminus \{d\}$ $(0 \le i \le C_2 \log n)$

Input: A packet with destination d and hop h arriving at v
1: **if** (v, d) is intact **then** forward p to d and **return**
2: **else** forward p to the successor w of v in π_h^G

- What if the edge (v, w) is failed?
- Raise hop count to $E_1 > C_2 \log n + 1$ and use different routing strategy for p.

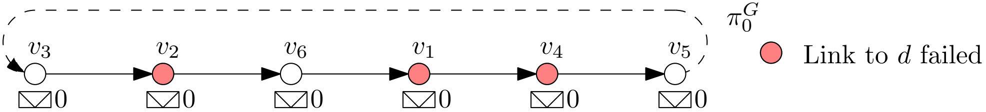
- **Assumption:** Adversary does not know the π_i^G.

Shared-Permutations - Key Concept

- Assume $\alpha \cdot n$ failed edges of the form (v, d) for constant $0 < \alpha < 1$

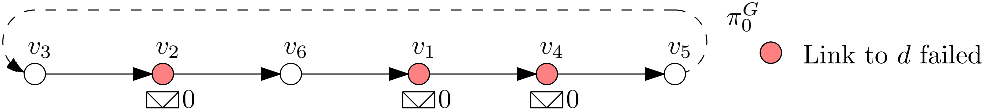
Shared-Permutations - Key Concept

- Assume $\alpha \cdot n$ failed edges of the form (v, d) for constant $0 < \alpha < 1$



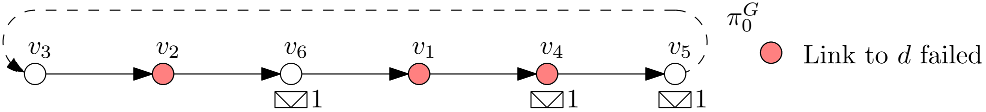
Shared-Permutations - Key Concept

+ Assume $\alpha \cdot n$ failed edges of the form (v, d) for constant $0 < \alpha < 1$



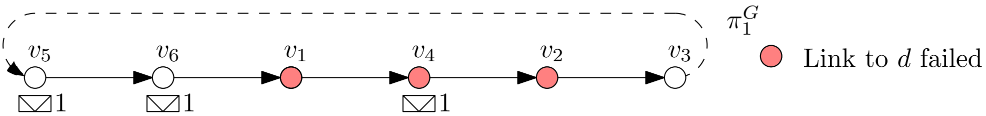
Shared-Permutations - Key Concept

+ Assume $\alpha \cdot n$ failed edges of the form (v, d) for constant $0 < \alpha < 1$



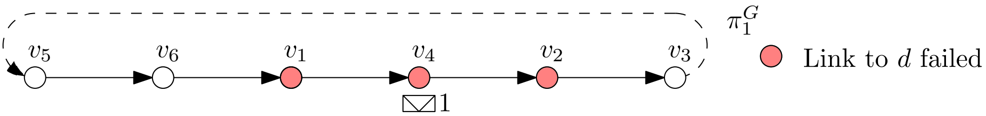
Shared-Permutations - Key Concept

+ Assume $\alpha \cdot n$ failed edges of the form (v, d) for constant $0 < \alpha < 1$



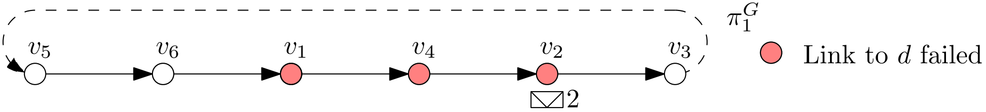
Shared-Permutations - Key Concept

- Assume $\alpha \cdot n$ failed edges of the form (v, d) for constant $0 < \alpha < 1$



Shared-Permutations - Key Concept

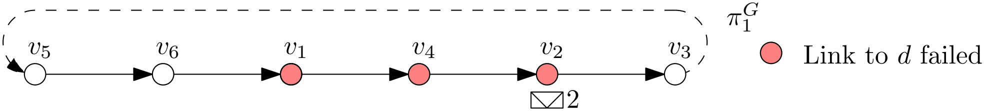
- Assume $\alpha \cdot n$ failed edges of the form (v, d) for constant $0 < \alpha < 1$



- **Invariant:** Any node $v \in V \setminus \{d\}$ receives flow from at most 1 source per hop value.

Shared-Permutations - Key Concept

+ Assume $\alpha \cdot n$ failed edges of the form (v, d) for constant $0 < \alpha < 1$



+ **Invariant:** Any node $v \in V \setminus \{d\}$ receives flow from at most 1 source per hop value.
+ Fraction of flows that *do not* reach d with h hops is roughly α^h
+ Results in a congestion of $\mathcal{O}(\sqrt{\log n})$ w.h.p.

Result (Shared-Permutations)

+ Adversary fails up to $\alpha \cdot n$ (const. $0 < \alpha < 1$)
+ All-to-one-routing to any destination d

1. $\mathcal{O}(\log n)$ hops
2. $\mathcal{O}(\sqrt{\log n})$ load on all nodes *w.h.p.*

Further Remarks

Empowered Adversary

- Allow adversary to measure load
- Eventually even local permutations can be inferred
- **Solution:** Periodically regenerate random bits
- *3-Permutations* and *Intervals*: Re-compute the failover table *locally* and quickly.

Further Remarks

Empowered Adversary

+ Allow adversary to measure load
+ Eventually even local permutations can be inferred
+ **Solution:** Periodically regenerate random bits
+ *3-Permutations* and *Intervals*: Re-compute the failover table *locally* and quickly.

+ Also: recover from bad low probability events

Further Remarks

Empowered Adversary

- Allow adversary to measure load
- Eventually even local permutations can be inferred
- **Solution:** Periodically regenerate random bits
- *3-Permutations* and *Intervals*: Re-compute the failover table *locally* and quickly.

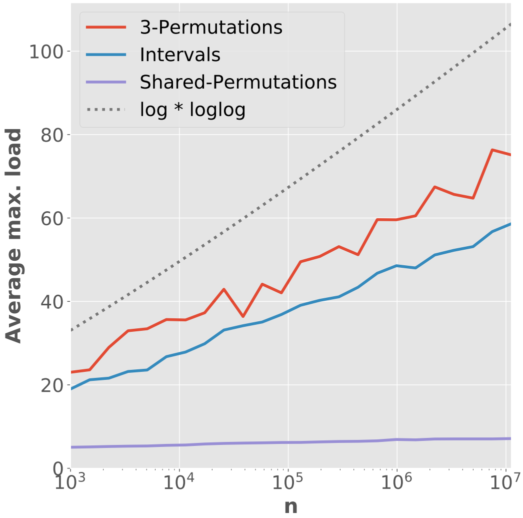
- Also: recover from bad low probability events

Reduced Amount of Failures

At most $n^{1-\delta}$ edge failures (any constant $\delta > 0$)

	3-Permutations	*Intervals*	*Shared-Permutations*
Load	$\mathcal{O}(1) \sim \mathcal{O}(\log n)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Hops	$\mathcal{O}(1) \sim \mathcal{O}(\log n)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$

Empirical Results - Average Maximum Load



Setup

- Complete graphs of increasing size
- All-to-one routing to random destination d
- Fail $\lceil 0.5 \cdot n \rceil$ edges of the form (v, d)

Results

- On average, no protocol induced load above $\log n \cdot \log \log n$
- *Shared-Permutation* load below 7 in all experiments
- *3-Permutations* lower than expected

Outlook – Possible Future Work

Improved Model

- Generalization to more realistic network models
- Data-centers have constant diameter, implying high degree
- Results should extend if degree at least polynomial in n

Simulations

- More in-depth simulations
- Different communication pattern
- Data-center topologies
- Comparison to deterministic schemes

Thank you very much for your attention!

	3-Permutations	*Intervals*	*Shared-Permutations*
Rule Set	Destination + Hop	Destination	Destination + Hop
Resilience	$\Theta(n)$	$\Theta(n/\log n)$	$\Theta(n)$
Congestion	$\mathcal{O}(\log^2 n \cdot \log\log n)$	$\mathcal{O}(\log n \cdot \log\log n)$	$\mathcal{O}(\sqrt{\log n})$
Hops	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$
Bits	$\mathcal{O}(\log^2 n)$	$\mathcal{O}(\log^2 n)$	$\mathcal{O}(\log^3 n)$
Shared Data	$\times$	$\times$	✓