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

Motivation - Local Failover Routing

- Mission-critical networks require fast reaction to link failures
- Fast rerouting mechanisms executing in the *data plane*
- First line of defense
  - Routes refined later on as part of the control plane

Challenges

- Algorithmically difficult
- Forwarding rules depend on *local* information only
- Sometimes low congestion impossible to achieve for deterministic algorithms

Our Contribution

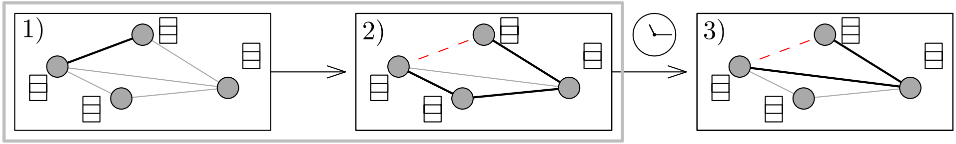
- Can we do better with randomized algorithms?
- Consider resulting load and not only connectivity

Local Failover Routing - Description

Routing Problem

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

- 1) Pre-computed forwarding rules (data plane) depending on *local* information
  2) Fast reaction in case of failures 3) Improve routes via control plane



- For a node v with neighborhood $\Gamma(v)$ pre-compute function

$$f_v : (\underbrace{\Gamma(v)}_{\text{Set of unreachable neighbors}} \times \underbrace{\mathcal{P}}_{\text{Packet header information (e.g. source address)}}) \to \underbrace{\Gamma(v)}_{\text{Next hop}}$$

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
- Focus on resulting congestion

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
  - Some data center networks have 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*

- Traffic going to some destination node d
- Each node $V \setminus \{d\}$ sends out one flow targeted at d

Model and Setting ctd.

Performance Measures

- Required number of *hops* for the flows to reach d
- Maximum *Load:* Number of flows crossing any edge and node $v \in V \setminus \{d\}$
  - Congestion threatens dependability
  - Major concern of any traffic engineering algorithm

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.

Different Rulesets

- Borokhovich and Schmid also give a $\sqrt{\varphi}$ lower bound in case the failover ruleset includes the source address.
- 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 congestion significantly

Randomization

- Intuition: Adversary doesn't know which links are used by the protocol
- Protocol will achieve results *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

Challenges

1. Keeping the routing table (and the ruleset) small and simple
2. Dealing with cycles in the packets routing paths
3. Avoiding the disruption of flows (TCP re-ordering)

Our Results - Overview

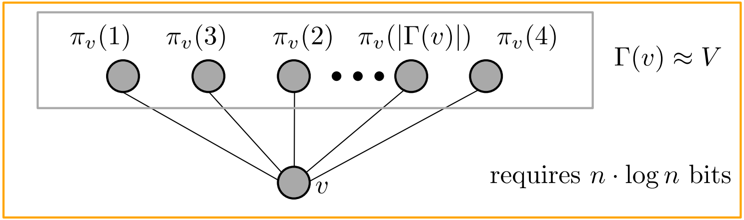
	3-Permutations	*Intervals*	*Shared-Permutations*
Rule Set	Destination + Hop	Destination	Destination + Hop [1]
Resilience	$\Theta(n)$	$\Theta(n/\log n)$	$\Theta(n)$
Congestion	$\mathcal{O}(\log^2 n \cdot \log\log n)$	$\mathcal{O}(\log n \cdot \log\log n)$	$\mathcal{O}(\sqrt{\log n})$
Hops	$\mathcal{O}(\log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

- All above results assume that up to $\mathcal{O}(n)$ or $\mathcal{O}(n/\log n)$ edges are failed by the adversary
- Deterministic protocols would allow the adversary to induce a load of $\Omega(\sqrt{n/\log n})$

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

Baseline Idea - Permutation Based Routing

- Failover function f_v grows exponential with $\Gamma(v)$
- Equip v with permutation π_v of neighbors $\Gamma(v)$



Basic Destination-Based Protocol (TODO reference where this idea comes from)

Input: A packet with destination d

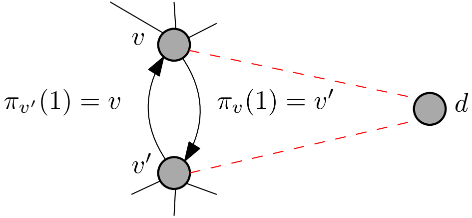
1: **if** (v, d) is intact **then** forward p to d and **return**

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

Observation 1: Forwarding Loops

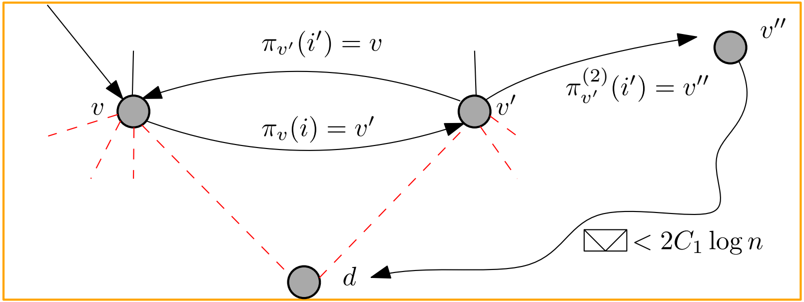


Observation 2: Failing "inner edges"

- *Intuition:* Failing edges not incident to d is not beneficial to the adversary.
- Permutations are unknown to the adversary
  - If (v, v') is failed: $P[\pi_v(1) = v'] = 1/(n-1)$
- All but $\mathcal{O}(\log n)$ off all nodes will forward via (v, d) or $(v, \pi_v(1))$ w.h.p.

3-Permutations Protocol

- Extend the simple permutation-based approach
- **Idea:** If at most $\alpha \cdot n$ (constant $\alpha < 1$) edges are failed, all packets *not* trapped in a cycle will reach the destination within at most $C_1 \log n$ hops w.h.p.



- Swap permutations every $C_1 \log n$ hops to break out of cycles
- **Caveat:** Packet travels in the cycle for $\mathcal{O}(\log n)$ hops and accumulates load on nodes lying on the cycle

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** $i \leftarrow \arg\max_{j \in \{1,2,3\}} \{h \geq (j-1) C_1 \log n\}$ and send p to first reachable node in $\pi_v^{(i)}$

3: increase $h \mathrel{+}= 1$

- Why only 3 permutations? No packet will get stuck in 3 cycles before hitting d w.h.p.

Theorem (3-Permutations (shortened))

Assume that the adversary fails at most $\alpha \cdot n$ edges. Then, if all nodes perform all-to-one routing to any destination d and follow the 3-Permutations protocol

1. *all but $\mathcal{O}(\log^2 n)$ nodes are passed by $\mathcal{O}(\log n \cdot \log \log n)$ flows, and*
2. *the remaining nodes receive $\mathcal{O}(\log^2 n \cdot \log \log n)$ load, and*
3. *no packet travels more than $\mathcal{O}(\log n)$ hops w.h.p.*

Intervals Protocol

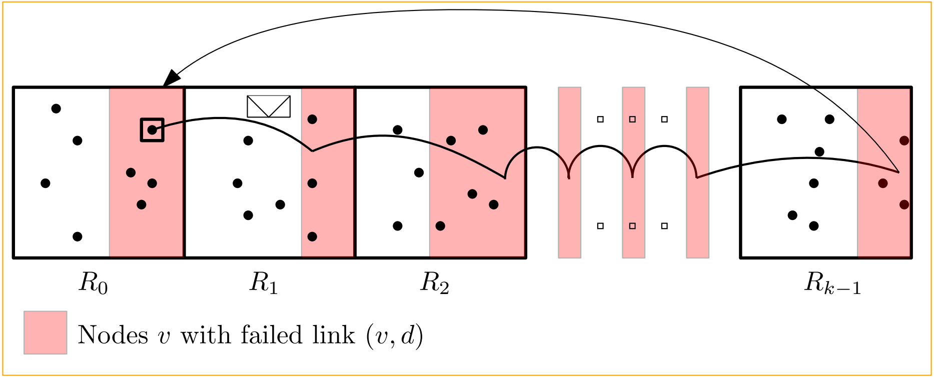
- Again extend upon simple permutation-based approach
- Avoid even temporary cycles w.h.p.
- Only relies on destination address

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
- Perform the Basic Permutation Protocol using this set of permutations π_v.

Intervals Protocol - Avoiding Temporary Cycles

- Assume adversary may destroy $\alpha \cdot I = \mathcal{O}(n/\log n)$ edges per partition $(0 < \alpha < 1)$.



- Packet located on a "bad" node in R_i will move to a "bad" node in R_{i+1} with probability at most α. Hence $\alpha^k < \mathcal{O}(1/n)$ prob. to traverse back to source.

Intervals Protocol (POV of node v)

Input: A packet 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

Theorem (Intervals Protocol)

Assume the adversary is allowed to fail up to $\alpha \cdot I$ many edges in every interval (constant $0 < \alpha < 1$ and $I = n/(4 \log_{1/\alpha} n)$). Then, when considering all-to-one routing to any destination d, the Intervals protocol guarantees

1. *that at most $\mathcal{O}(\log n \cdot \log \log n)$ flows pass any node $v \in V \setminus \{d\}$, and*
2. *every packet travels at most $\mathcal{O}(\log n)$ hops w.h.p.*

- Maximum resilience for $\alpha = 1/e$.
- Tradeoff of maximum resilience and load

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 $(0 \leq i \leq C_1 \log n)$

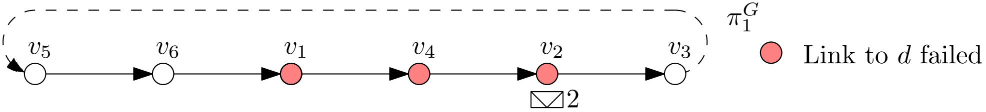
Input: A packet with destination d and hop h
1: **if** (v, d) is intact **then** forward p to d and **return**
2: **else** forward p to the successor w of v in π_h^G

- What if the edge (v, w) is failed?
- Raise hop count to $E_1 > C_1 \log n + 1$ and use different routing strategy for p.

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

Shared-Permutations - Key Concept

- Assume the adversary fails $\alpha \cdot n$ edges $(0 < \alpha < 1)$ of the form (v, d).
- Neglect any failed "inner edges"



- **Invariant:** Any node $v \in V \setminus \{d\}$ receives flow from at most 1 source per hop value.
- Fraction of nodes that host a packet with hop h is roughly α^h.
- Packet hits a fixed node $v \in V \setminus \{d\}$ with prob. $\alpha^h/n \to \mathcal{O}(\sqrt{\log n})$ hits w.h.p.

Theorem (Shared-Permutations Protocol)

Assume that the adversary is allowed to fail $\alpha \cdot n$ edges. When performing all-to-one routing to any destination d, the Shared-Permutations protocol guarantees that

1. *every node $v \in V \setminus \{d\}$ is passed by $\mathcal{O}(\sqrt{\log n})$ flows, and*
2. *no packet travels more than $\mathcal{O}(\log n)$ hops w.h.p.*

Further Remarks

Empowered Adversary

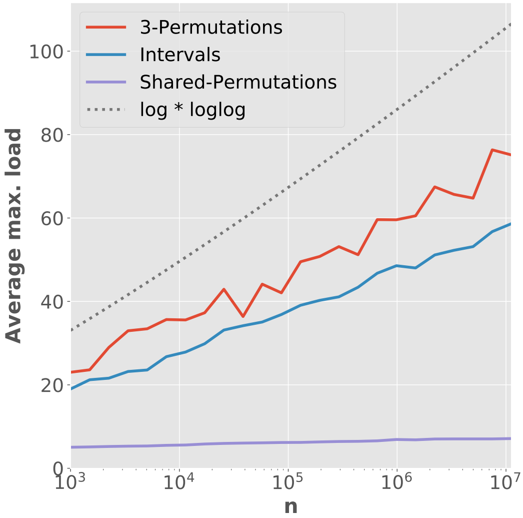
- Allow adversary to measure load
- Eventually even local permutations can be inferred
- **Solution:** Periodically regenerate random bits
- *3-Permutations* and *Intervals*: Nodes can re-compute the failover table *locally* and quickly.

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	$\mathcal{O}(1) \sim \mathcal{O}(\log n)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$

Simulation Results - Maximum Average Load



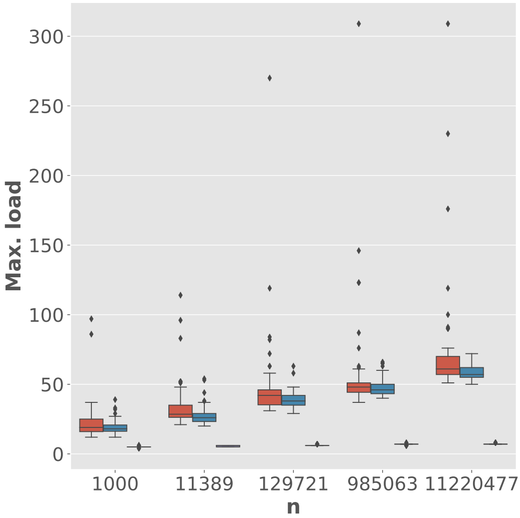
Setup

- 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

Simulation Results - Average Load Box Plot



Results

- Results of the *Shared-Permutations* and *Intervals* protocol tightly concentrated
- Observation: Not all *3-Permutations* runs contain temporary cycles
- Sometimes cycles exist and induce high load
- Still within the theoretical bound of $\mathcal{O}(\log^2 n \cdot \log\log n)$

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	$\times$	$\times$	$\checkmark$