On the Feasibility of Perfect Resilience with Local Fast Failover

Klaus-Tycho Foerster (ORCID)
Faculty of Computer Science, University of Vienna, Austria
klaus-tycho.foerster@univie.ac.at

Juho Hirvonen (ORCID)
Aalto University, Finland
juho.hirvonen@aalto.fi

Yvonne-Anne Pignolet (ORCID)
DFINITY, Switzerland
yvonneanne@dfinity.org

Stefan Schmid (ORCID)
Faculty of Computer Science, University of Vienna, Austria
stefan_schmid@univie.ac.at

Gilles Tredan
LAAS-CNRS, France
gtredan@laas.fr

─── **Abstract** ───

In order to provide a high resilience and to react quickly to link failures, modern computer networks support fully decentralized flow rerouting, also known as local fast failover. In a nutshell, the task of a local fast failover algorithm is to *pre-define* fast failover rules for each node using locally available information only. These rules determine for each incoming link from which a packet may arrive and the set of local link failures (i.e., the failed links *incident* to a node), on which outgoing link a packet should be forwarded. Ideally, such a local fast failover algorithm provides a *perfect resilience* deterministically: a packet emitted from any source can reach any target, as long as the underlying network remains connected. Feigenbaum et al. showed [19] that it is not always possible to provide perfect resilience; on the positive side, the authors also presented an efficient algorithm which achieves at least 1-resilience, tolerating a single failure in any network. Interestingly, not much more is known currently about the feasibility of perfect resilience.

This paper revisits perfect resilience with local fast failover, both in a model where the source can and cannot be used for forwarding decisions. We first derive several fairly general impossibility results: By establishing a connection between graph minors and resilience, we prove that it is impossible to achieve perfect resilience on *any non-planar graph*; furthermore, while planarity is necessary, it is also not sufficient for perfect resilience. In some scenarios, a local failover algorithm cannot even guarantee that a packet reaches its target, even if the source is still highly connected to the target after the failures.

On the positive side, we show that graph families which are closed under the subdivision of links, can allow for simple and efficient failover algorithms which simply *skip failed links*. We demonstrate this technique by deriving perfect resilience for outerplanar graphs and related scenarios, as well as for scenarios where the source and target are topologically close after failures.

1   Introduction

The dependability of distributed systems often critically depends on the underlying network, realized by a set of routers. To provide high availability, modern routers support local fast rerouting of flows: routers can be pre-configured with conditional failover rules which define, for each incoming port and desired target, to which port a packet arriving on this incoming port should be forwarded deterministically depending on the status of the *incident* links only: as routers need to react quickly, they do not have time to learn about remote failures.

This paper is motivated by the following fundamental question introduced by local fast rerouting mechanisms: *Is it possible to pre-define deterministic local failover rules which guarantee that packets reach their target, as long as the underlying network is connected?* This desired property is known as *perfect resilience.* The challenge of providing perfect resilience hence lies in the decentralized nature of the problem and the fact that routers only have local information about failed links; achieving perfect resilience is straightforward with global knowledge, as one could simply compute a shortest path.

Unfortunately, perfect resilience cannot be achieved in general: Feigenbaum et al. [19, 20] presented an example with 12 nodes for which, after certain failures, no forwarding pattern on the original network allows each surviving node in the target's connected component to reach the target. On the positive side, the authors showed that it is at least always possible to tolerate one link failure, i.e., to be 1-resilient. Interestingly, not much more is known today about when perfect resilience can be achieved, and when not.

1.1   Contributions

This paper studies the problem of providing perfect resilience in networks where nodes only have local information, considering both a model where nodes can and cannot match the packet source. On the negative side, we characterize network instances which do not allow for perfectly resilient solutions. From this characterization it follows that perfect resilience is impossible already on simple and small planar graphs, and even in scenarios where a source is in principle still highly connected to the target after the failures, by $\Omega(n)$ disjoint paths; however, it cannot route to it. We also derive the fairly general negative result that perfect resilience is impossible on any non-planar graph, and that already the complete graph K_5 and the complete bipartite graph $K_{3,3}$ do not permit for perfectly resilient solutions. To this end, we show an intriguing connection between graph minors and perfect resilience, and show that every graph minor retains the perfect resilience property.

On the positive side, we describe perfectly resilient algorithms for all outerplanar graphs and related scenarios (e.g., scenarios where the source and the destination are on the same face after failures), as well as for non-outerplanar scenarios where the destination is within two hops of the source. For our positive results, we establish the general insight that graph families which are closed under subdivision of links, allow for simple failover algorithms in which nodes can just skip locally failed ports, requiring very small forwarding tables.

Our results are summarized in Tables 1 to 3.

1.2   Related Work

Besides the paper by Feigenbaum et al. [19, 20], which is the closest work to ours, the design of local fast failover algorithms has already been studied intensively, see also the recent survey [13]. In the following, we will concentrate on related work which provides formal

Graph class	Without source matching	With source matching
Outerplanar	Perfect resilience: Thm 28	Perfect resilience (see left)
K_4	Perfect resilience: Thm 30	Perfect resilience (see left)
Planar graphs	Counterexample: Thm 23	Counterexample: Thm 24
Non-planar graphs	Perfect res. impossible: Thm 19	?

Table 1 Summary of perfect resilience results for specific graph classes.

Graph class	Without source matching	With source matching
Closed under link subdivision	Perfect resilience $\Rightarrow$ skipping perfect resilience, $f(m)$-resilience $\Rightarrow$ skipping $f(m)$-resilience: Thm 31	
General graphs	No perfect resilience [19]	No perfect resilience Thm 20
General graphs	No superconstant resilience: Thm 21	
General graphs	Impossible even if large connectivity under F: Thm 22	
Subgraph	Perfect resilience kept: Thm 8	Perfect resilience kept: Cor 9
Minor	Perfect resilience kept: Thm 15	Perfect resilience kept: Cor 16

Table 2 Characteristics of forwarding patterns and parametrized resilience results.

Context	Without source matching	With source matching
G planar and s, t on the same face	Perfect resilience: Cor 29	Perfect resilience (see left)
Small distance in general graphs	All nodes at most two hops from t in $G \setminus F$: perfect resilience: Thm 27	s and t at most two hops away in $G \setminus F$: perfect resilience: Thm 26

Table 3 Perfect resilience under specific settings.

resilience guarantees, however, we point out that there also exists much interesting applied work on the topic, e.g., [9, 16, 22, 31, 32, 37, 40].

A key property and challenge in the design of local fast failover algorithms is related to the fact that nodes need to react to failures fast, i.e., routers can only have local failure information and failover decisions are static, ruling out algorithms based on link reversal [17, 27] or reconvergence [10]. Furthermore, it is not possible to rewrite packet headers [11, 18], e.g., to carry failure information, which is often impractical; this rules out classic graph exploration algorithms such as [6, 35]. The fact that neither routing tables nor packet headers can be modified also means that it is not possible to adopt, e.g., rotor router approaches [4], which would provide connectivity but require state. We also note that the focus in this paper is on *deterministic* algorithms, which, in contrast to related work such as [2, 12], do not require random number generators at routers.

From a distributed computing perspective, the design of local fast failover algorithms also features an interesting connection to distributed algorithms "with disconnected cooperation" [33], a subfield of distributed computing where solve a problem in parallel without exchanging information among them. This was first pointed out by Pignolet et al. in [34], and used in several related works [7, 24], however, primarily to reduce load in dense networks, rather than to improve resilience.

In terms of local fast failover algorithms which provide a provably high resilience, there exist several interesting results by Chiesa et al. who introduced a powerful approach which

decomposes the network into arc-disjoint arborescence covers [12, 14, 15, 15]. Chiesa et al. are primarily interested in k-connected graphs, and leverage a well-known result from graph theory, which allows us to decompose any k-connected graph in a set of k directed spanning trees (rooted at the same vertex, the target) such that no pair of spanning trees shares a link in the same direction. Packets are routed along some arborescence until hitting a failure, and are then rerouted along a different arborescence. Chiesa et al. conjecture that for any k-connected graph, basic failover routing can be resilient to any $k-1$ failures, and show that this conjecture is true for a number of interesting graphs, including planar graphs.

However, the general conjecture remains to be proved or disproved, and also differs conceptually from the perfect resilience considered in our paper, which asks for connectivity as long as the underlying graph is connected. Indeed, as we show in this paper, perfect resilience is for example *not* achievable in planar graphs, even when matching on the packet source. Nevertheless, the arc-disjoint arborescence decomposition technique has been explored further, not to improve resilience but e.g., to reduce stretch [21–24].

2   Model

Let $G = (V, E)$ be a network represented by an undirected graph of nodes ("routers") V connected through undirected links E along which packets are exchanged. Initially, an arbitrary set $F \subset E$ of links fail (rendering them unusable in both directions). We write $G \setminus F$ to denote the graph G with the links in F removed. More generally, $G \setminus E'$ and $G \setminus V'$ represent the graph G after removing the set of links in $E' \subset E$ or the set of nodes $V' \subset V$ and their incident links respectively. We define $n = |V|$, $m = |E|$, and write $V_G(v)$ and $E_G(v)$ for the neighbors and incident links of node v, respectively. G is omitted and the degree $d_v = |V_G(v)|$ is shorthanded to d when the context is clear.

We study the class of local routing (forwarding) algorithms, in which every node $v \in V$ takes deterministic routing decisions based solely on

- the target t of the packet to route,
- the set of incident failed links $F \cap E(v)$, and
- the receiving or incoming port (*in-port*) of the packet at node i.[1]

In particular, this implies that neither the state of the packet nor the state of the node can be changed, e.g., by header rewriting or using dynamic routing tables. In other words, routing has be to purely local, which we formalize next.

Given a graph G and a target $t \in V(G)$, a local routing algorithm is modelled as a *forwarding function* $\pi_v^t \colon E(v) \cup \{\bot\} \times 2^{E(v)} \mapsto E(v)$ at each node $v \in V(G)$, where $\bot$ represents the empty in-port, i.e. the starting node of the packet. That is, given the set of failed links $F \cap E(v)$ incident to a node v, a forwarding function π_v^t maps each incoming port (link) to the outgoing port (link) $\pi_v^t(e, F)$. The tuple of forwarding functions $\pi^t = (\pi_v^t)_{v \in V}$ is called the *forwarding pattern*.

We say that a forwarding pattern is *k-resilient* if for all G and all F where $|F| \leq k$ the forwarding pattern routes the packet from all $v \in V$ to the target t when v and t are connected in $G \setminus F$. A forwarding pattern is *perfectly resilient* if it is ∞-resilient: the forwarding always succeeds in the connected component of the target. Let $A_p(G, t)$ be the set of such perfectly resilient patterns (algorithms), abbreviated by A_p when the context is clear.

---

[1] Note that without knowledge of the in-port, already very simple failure scenarios prevent resilience. For example, consider a packet reaching a node v from a node w, where all further links incident to v failed. Node v must return the packet to w, which forwards it back to v, resulting in a permanent forwarding loop.

We also consider the model where the forwarding functions depend on the source node s in addition to the in-port, incident failures and target. We denote the forwarding function and pattern by $\pi_v^{s,t}$ and $\pi^{s,t}$, respectively; it is perfectly resilient if forwarding always succeeds when s and t are connected in $G \setminus F$, where the set of such patterns is denoted by $A_p(G, s, t)$.

Note that resilient algorithms that do not match on the source also imply resilient algorithms in the model where we can match on the source (it can be ignored); similarly, impossibility results in the model where we can match on the source also imply impossibility in the model where we do not match on the source (extra information).

3 First Insights

Let us start by providing some basic insights and also derive a first impossibility result. In general, we observe that to achieve perfect routing, every possible route to the target must be explored. The challenge is that the nodes must ensure this property in a distributed fashion, while having no means to exchange their local views of the network. We will show that in order to achieve this, the forwarding functions must be coordinated for global coherence. We will then study the limits of such patterns. But first, we formalize the "local" perspective of a node, illustrated in Figure 1.

▶ **Definition 1** (Relevant node). *For any node $i \in G$, $i \neq t$, a neighboring node $j \in N_{G \setminus F}(i)$ is* relevant *for routing to t under the failure set F iff it there is a path from i to t in $G \setminus V'$ where $V' = V_{G \setminus F}(i) \setminus \{j\}$ is the set of all other nodes still connected to i. In other words, j is a potential relay to reach t from i's perspective if in addition to F all links incident to other neighbors of i have failed.*

Note that nodes are oblivious to non-incident link failures and relevance is defined by a path to the target which does not use further neighbors of the current node. Hence, if a neighbor v of some node i is relevant, then the failure of further links incident to i can only remove i's relevance if the link (v, i) fails.

▶ **Observation 2** (Nodes stay relevant). If a node v is relevant for a node i under a failure set F, v remains relevant for i under any failure superset F' with $F \subseteq F'$, if v remains a neighbor of i under F'.
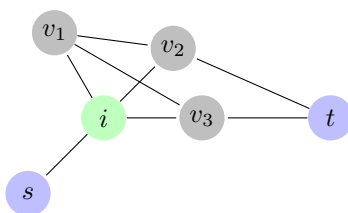
Figure 1 In this context, only nodes v_2 and v_3 are relevant for i: it is not useful for i to relay to v_1 as v_2 and v_3 constitute shorter alternatives for the same paths towards t. Note that relevance is lower bounded by connectivity in the failure-free case and that the number of relevant neighbors can increase with failures, e.g., if (i, v_2) or (i, v_3) fail, v_1 becomes relevant for i

▶ **Definition 3** (Orbit). *Let $\pi_v^t(\cdot, F)$ be the forwarding function of a node v for some set of failed links F. We say a set of neighbors $V' \subseteq V(v)$ is in the same orbit w.r.t. $\pi_v^t(\cdot, F)$, if for all pairs $v_1, v_2 \in V'$ it holds: there is some $k \in \mathbb{N}$ s.t. $\left(\pi_v^t(\cdot, F)\right)^k (v_1) = v_2$.*

▶ **Lemma 4.** *Let $G, A \in A_p(G, t)$. For all F and all $t, i \in V$ it holds: If i has at least two relevant neighbors under F, then all relevant neighbors of i must be part of the same orbit in A's forwarding function $\pi_v^t(\cdot, F)$ of i.*

Proof. Our proof will be by contradiction. Assume there is a failure set F and a node $i \in V$ s.t. there are relevant neighbors v_1, v_2 of i where there is no $k \in \mathbb{N}$ s.t. $\left(\pi_v^t(\cdot, F)\right)^k (v_1) = v_2$. In other words, v_1, v_2 are relevant for i, but not in the same orbit of π_v^t for F.

Assume the packet starts on v_1, and beyond F, we fail all further links, except for those still incident to i and links forming a path from v_2 to t without visiting any other neighbors of i except v_2. Such a path must exist as v_2 is relevant for i. The only routing choice at v_1 now is to route to i. Observe that routing under π_i will never reach v_2, as each packet leaving i will immediately bounce back to i (except if it would go to v_2). Hence, one will not reach t, even though there is a path from i to t under F, and thus A is not perfectly resilient.   ◀

We now extend Lemma 4 to allow matching on the source node as well. However, we can now no longer enforce that the packet starts on an arbitrary neighbor of the node i, and hence obtain slightly modified results. Essentially, we need to enforce that the packet can reach node i from s, and that this taken path is node-disjoint from other paths from i to t.

▶ **Lemma 5.** *Let $G = (V, E), A \in A_p(G, s, t)$, where $s \neq i$ is connected to two different relevant neighbors $v_1, v_2 \in V$ of $i \in V$. For all F where $F \cap \{(v_1, i), (v_2, i)\} = \emptyset$ it holds that all relevant neighbors of i under F must be part of the same orbit in A's forwarding function $\pi_i^{s,t}(\cdot, F)$.*

Proof. We use an analogous argumentation as for the proof of Lemma 4, where we will fail links not incident to i, s.t. i cannot further adapt its forwarding function. We start with the case where we fail all incident links of v_1 except (v_1, i) and (v_1, s). Next, pick any further relevant neighbor $j \neq v_1$ of i, and fail all links in the graph except 1) a path from j to t, not using further nodes from $V(i)$ (due to j being relevant), 2) $(v_1, i), (v_1, s)$, and 3) all links still alive incident to i. Then there must be some $k \in \mathbb{N}$ s.t. $\left(\pi_i^{s,t}(v_1, F)\right)^k = j$. The same argumentation can be made for v_2, i.e., every further relevant neighbor of i will be reached by iterative usage of the forwarding function, as that neighbor might be the only connection to the target t. As every relevant neighbor of i (including v_2) is in this way reachable from v_1, and as every relevant neighbor of i (including v_1) is in this way reachable from v_2, all relevant neighbors of i are in the same orbit w.r.t. $\pi_i^{s,t}(\cdot, F)$.   ◀

Note that the above proof can be directly extended to s being connected to multiple relevant neighbors of i, as long as two such neighbors remain under failures:

▶ **Corollary 6.** *Let $G = (V, E), A \in A_p(G, s, t)$, where $s \neq i$ is connected to $k \geq 2$ relevant neighbors $v_1, \ldots, v_k \in V$ of $i \in V$. For all F where $|F \cap \{(v_1, i), \ldots, (v_k, i)\}| \leq k - 2$ it holds that all relevant neighbors of i under F must be part of the same orbit in A's forwarding function $\pi_i^{s,t}(\cdot, F)$.*

Observe that the previous proof arguments rely on the path from the source s to i via some relevant neighbor v_1 being node-disjoint from the path from some other relevant neighbor v_2 of i to the target t. If the above statement also holds vice-versa, v_1, v_2 need to be in the same orbit. We cast this insight into the following corollary:

▶ **Corollary 7.** *Let $G = (V, E), A \in A_p(G, s, t)$ with $i \neq s$. Fix any F where i has k relevant neighbors, s.t. for $k' \geq 2$ of them, denoted $v_1, \ldots, v_{k'}$. In this case there is a path from s to i via v_x which is node-disjoint to a path from i to t via some v_y, $y \neq x, 1 \leq y \leq k'$ for*

each individual v_x, $1 \leq x \leq k'$. Then, all $v_1, \ldots, v_{k'}$ must be part of the same orbit in A's forwarding function $\pi_i^{s,t}(\cdot, F)$ of i.

We now analyze our results in the context of prior work, which considered forwarding without matching on the source.

Relation to the work of Feigenbaum et al. [20]. Note that Lemma 4 implies corollaries reminiscent to two results presented by Feigenbaum et al. [20, Lemma 4.1, 4.2] namely that

- if all neighbors are relevant, the forwarding function at each node must be a cyclic permutation, cf. [20, Lemma 4.2],
- if a node has only two neighbors v_1, v_2 and both are relevant, then a packet from v_1 must be forwarded to v_2 and vice versa, cf. [20, Lemma 4.1].

Feigenbaum et al. use these lemmas to prove the impossibility of perfect resilience for a specific graph. Their lemmas implicitly define a version of relevance based on link-connectivity, whereas we require a node version, namely that a neighbor v of i is only relevant (from the viewpoint i) if there is a routing from v to t that does not use any other neighbor of i. As such, their assumptions are weaker and hence their results are stronger. However, both Lemmas 4.1 and 4.2 from Feigenbaum et al. [20] are not correct in their generalized form.

For a counterexample, consider the graph in Figure 2. If u sends the packet to v, then both Lemma 4.1, 4.2 from Feigenbaum et al. state that the packet *must not* be forwarded back to u, i.e., for perfect resilience, it *must* next be sent to w, and from there to u. Yet, v could also forward the packet back to u, and then to x, a contradiction to their claim: the only two possible neighbors of t are x and v, and hence visiting both nodes suffices.
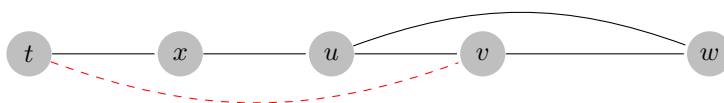
Figure 2 Graph with target t where the dashed red link has failed. Assume the packet starts at u and is forwarded to v. The node v can bounce the packet back to u, as only u is relevant.

4 A Minor Perspective on Perfect Resilience

We now study the relationship between graph minors and perfect resilience algorithms. As we will see, this relationship will be relevant for the design of local resilient algorithms as well. We will first observe that a perfectly resilient algorithm is also perfectly resilient on subgraphs and contractions of its original graph (Theorems 8 and 12). Then, since subsetting and contracting are the two fundamental operations in the minor relationship, we deduce that the existence of a perfectly resilient algorithm on a graph G implies its existence on any minor of G (Theorem 15).

As an application of the above results, we prove in §4.2 that if a graph is not planar, it does not allow for perfect resilience (Theorem 19), as both K_5 and $K_{3,3}$ do not allow for perfect resilience (Lemmas 17 and 18).

4.1 Contraction and Minors

We start by retaining perfect resilience on subgraphs:

▶ **Theorem 8** (Subset Stability). *If a perfectly resilient forwarding pattern exists for a graph G and target t then there also exists a perfectly resilient pattern for all subgraphs of G when they contain t, $A_p(G,t) \neq \emptyset \to G' \subset G, t \in V(G'), A_p(G', t) \neq \emptyset$*

The proof idea is as follows: we take the original algorithm A, and fail all links needed to obtain the desired subgraph G'. Then, we can simulate a perfectly resilient algorithm on G' via A, where A also needs to function correctly when further links fail in the subgraph G'.

Proof. Let $A \in A_p(G,t)$. Let $\bar{G}' = G \setminus G'$. Let A' be a fast rerouting algorithm on G' s.t. $\forall F' \subset E(G'), A'(F') = A(F' \cup E(\bar{G}'))$: algorithm A' simply executes A in a context where all the parts of G that are not in G', i.e., in $\bar{G}'$ have also failed.

Since $A \in A(G,t)$, we have that if v, t are connected in $G \setminus (F' \cup E(\bar{G}'))$ then A succeeds in routing. Since $\forall v, t \in V(G')$, if v, t are connected in $G' \setminus F'$ then (v, t) is connected in $G \setminus (F' \cup E(\bar{G}'))$, and we deduce that A achieves a perfect resilience on G'. ◀

Note that the above proof arguments hold analogously when we can match on the source s, where perfect resilience only needs to hold when s, t remain in the same connected component.

▶ **Corollary 9** (s Subset Stability). *$A_p(G, s, t) \neq \emptyset \to G' \subset G, s, t \in V(G'), A_p(G', s, t) \neq \emptyset$*

After covering subgraphs, we next cover node contractions. Abstractly, a node contraction merges two neighboring nodes, while retaining their joint connectivity to the graph.

▶ **Definition 10** (Node Contraction). *Let $G = (V, E)$ and $i, j \in V$ be two neighboring nodes, i.e., $(i, j) \in E(G)$. Let G' be the (i, j)-contracted graph of G such that: $V(G') = V(G) \setminus \{j\}$ and $E(G') = E(G) \cup \{(a, i), \forall a \in V_G(j)\} \setminus \{(a, j), \forall a \in V_G(j)\}$. We denote by $cont(G)$ the set of all possible contracted graphs of G.*

We next define the natural emulation of a (i, j)-contraction with respect to port mappings:

▶ **Definition 11** (Mapping Contraction). *Let i, j be two neighboring nodes without common neighbors, $V(i) \cap V(j) = \emptyset$, and let π_i and π_j be their forwarding patterns. We define $\pi_{ij} : V(i) \cup V(j) \cup \{\bot\} \mapsto V(i) \cup V(j)$ as the contracted mapping where on i's side $\forall v \in V(i)$, $\pi_{ij}(v) = \pi_i(v)$ unless $\pi_i(v) = j$; in the latter case $\pi_{ij}(v) = \pi_j(\pi_i(v)) = \pi_j(i)$, unless $\pi_j(i) = i$ in which case $\pi_{ij}(v) = \pi_i(j)$, unless $\pi_i(j) = j$; in this last case, $\pi_{ij}(v) = \bot$ For j's side, proceed symmetrically accordingly. By convention, we set $\pi_{ij}(\bot) = \pi_i(\bot)$*

Note that for the above two definitions, matching on the source is immaterial, and hence the definitions are identical for forwarding patterns with and without the source. For the following Theorem 12, we prove that they preserve resilience when not matching on the source. In the proof of the result we show that the routing of a packet, starting at some node, towards the target t, can be directly transferred to the contracted case, by simulating the original algorithm. For this simulation, it is again immaterial if we match on the source or not, described in Observation 14.

▶ **Theorem 12** (Contraction Stability). *Let $G = (V, E), i, j \in V$ be two neighboring nodes, and let G' be the (i, j)-contraction of G. Given $A \in A_p(G, t)$, a perfectly resilient algorithm on G, then the (i, j)-contracted algorithm of A is perfectly resilient on G'.*

We will utilize the following lemma for the proof of Theorem 12:

▶ **Lemma 13** (Contraction Stability: Algorithm Transfer). *Given G and $(i, j) \in E(G)$ let G' be the corresponding (i, j)-contraction. Let $R = \{(j, r), r \in V(i) \cap V(j)\}$. Let $A : F \mapsto \{\pi_v^t(\cdot, F), v \in V\}$. Define $A' : F \mapsto \{\pi_v^t(\cdot, F), v \in V'\}$ to be the (i, j)-contracted algorithm of A as follows:*

- *Case I: Identical behavior on unaffected nodes, $\forall v \in V', v \neq \{i\}$, $\pi_v'^t(\cdot, F) = \pi_v^t(\cdot, F \cup R)$.*
- *Case II: Replace i's permutation by the contracted algorithm. Let $\pi_i'^t(\cdot, F) = \pi_{\{i,j\}}^t(\cdot, F \cup R)$.*
- *Case III: Replace j's port by i's port on j's neighbors permutations: $\forall k \in V(j), \exists v$ s.t. $\pi_k^t(v, F \cup R) = j \Rightarrow \pi_k'^t(v, F) = i$, $\forall k \in V(j), \exists v$ s.t. $\pi_k^t(j, F \cup R) = v \Rightarrow \pi_k'^t(j, F) = v$.*

Let P (resp P') be the sequence of links traversed using A by a packet from s to t under $F \cup R$ (resp traversed using A' under F).

Let Q be a rewriting of P in which we replace every occurence of j by i. And let Q' be the rewriting of Q in which we remove every occurrence of (i, i). We have $Q = P'$.

Proof of Lemma 13. We proceed by induction on the k first hops $Q[1..k]$ and $P'[1..k]$ of the sequences. When the context is clear, we write π_v for the forwarding function of node v using A in context $F \cup R$, and π_v' the forwarding function of v using A' in context F.

Base case: let $Q[1] = (s, q), P'[1] = (s, a)$. We need to prove $q = a$. 1): if $s \neq i$, $\pi_s'(\bot) = \pi_s(\bot)$ as defined in case I. 2): if $s = i$, the packet starts in the i, j contraction. Since $a = \pi_s'(\bot) = \pi_{i,j}(\bot) = \pi_i(\bot)$ (Case II), we deduce that P describes possibly some hops between i and j, and then q. Since (i, j) transitions are rewritten (i, i) in Q and removed in R, the first node that is not i nor j to appear in P must be a. Thus $q = a$.

Induction step: Assume the following statement holds for any $k' \leq k$: $Q[k'] = P'[k']$. We prove that necessarily $Q[k] = P'[k]$. Let $Q[k - 1] = (v_-, v), P'[k - 1] = (v_-, v), Q[k] = (v, q), P'[k] = (v, a)$. We need to show that $a = q$. We again start by the simplest case 1): $v \notin V'(i) \cup \{i\}$: since $\pi_v = \pi_v'$ by Case I, $q = \pi_v(v_-) = \pi_v'(v_-) = a$. 2): if $v \in V'(i)$. We need to look at v_-. If 2.1): $v_- \neq i$ we again directly use Case I: $\pi_v(v_-) = \pi_v'(v_-)$. If 2.2): $v_- = i$ the packet just left the (i, j) contraction. In P, v_- can correspond to either i or j in P. If it corresponds to i, use Case I. If it was a j, since in Case III we replaced j's connections (in A) by i's connections (in A'), and since no node v is both a neighbor of i and j in $F \cup R$, this mapping uniquely applies. 3): if $v = i$: the packet is in the (i, j) contraction. By definition of Q, necessarily $v_- \neq i$ or j. Since $\pi_i'(v_-) = \pi_{ij}(v_-)$ we deduce $q = a$. ◀

Proof of Theorem 12. Let $R = \{(j, r), r \in V(i) \cap V(j)\}$. Let A' be the (i, j) contracted algorithm of A. Let F be a set of link failures of G'. Observe that if s, t are connected in $G' \setminus F$ then s, t are connected in $G \setminus F \cup R$. Let s, t be connected in $G' \setminus F$. Let $p_{A'}$ (resp. p_A) be the path produced by A' on $G' \setminus F$ (resp. A on $G \setminus (F \cup R)$) from s to t.

Since $A \in A_p$ and s, t are connected in $G \setminus (F \cup R)$, then necessarily p_A is finite and ends up in t. Therefore, Lemma 13 implies $p_{A'}$ is finite and ends in t too: A' succeeds. ◀

▶ **Observation 14** (*s* Contraction Stability). The results of Theorem 12 also apply when matching on the source, i.e., in particular a perfectly resilient algorithm $A \in A_p(G, s, t)$ implies a perfectly resilient algorithm on a contracted graph G', i.e., $A_p(G', s, t) \neq \emptyset$.

We next combine Theorem 8 (subsetting) and Theorem 12 (contraction) to obtain the corresponding result for minors:

▶ **Theorem 15** (Minor Stability). *Given two graphs G, G', G' a minor of G it holds that $A_p(G, t) \neq \emptyset \Rightarrow A_p(G', t) \neq \emptyset$: if G permits a perfectly resilient scheme, so do its minors.*

Proof. Observe that there exists a sequence of graphs $G_1, G_2 \ldots$ s.t. $G \mapsto G_1 \mapsto G_2 \mapsto \ldots \mapsto G'$, where $\mapsto$ is either a subsetting or a contraction operation. Since both operations preserve the existence of a perfect scheme thanks to Theorems 8 and 12, in combination, they imply the general minor relationship. ◀

The above proof can be directly transferred to the model with matching on the source, by utilizing Corollary 9 and Observation 14, instead of Theorems 8 and 12:

▶ **Corollary 16** (Minor Stability). *Given two graphs G, G', G' a minor of G, it holds that $A_p(G, s, t) \neq \emptyset$ implies that $A_p(G', s, t) \neq \emptyset$: if G permits a perfectly resilient scheme, so do its minors.*

The subset and contraction lemmas provide constructive proofs, showing how to derive a perfectly resilient scheme from a larger graph to one of its minors. We can also exploit this result in its contrapositive form: by showing the absence of perfect resilience schemes on the minors defining a minor-closed graph family, we can prove the impossibility of perfectly resilient schemes for whole graph families, e.g., planar graphs.

4.2   Case Study: Non-Planar Graphs

In order to prove the impossibility of perfect resilience on non-planar graphs, we first cover the impossibility of perfect resilience on the K_5 and then on the $K_{3,3}$. Note that our impossibility results in this section are for forwarding patterns that do not match on the source.

▶ **Lemma 17.** *The complete graph with five nodes does not allow for perfect resilience, i.e., it holds that $A_p(K_5, t) = \emptyset$.*

Proof. Let $V(K_5) = \{v_1, v_2, v_3, v_4, v_5\}$, where we assume w.l.o.g. v_1 to be the source s and v_5 to be the target t. To prove the lemma, we construct sets of link failures in which we leave target v_5 connected to only one of the non-target nodes, and "fine-tune" the set of link failures so that v_1's packet will only visit 3 out of the 4 neighbors of t (say (v_1, v_2, v_3)).

By contradiction, let $A \in A_p(K_5, t)$. Let $\pi_{v_1}^t(\cdot, F)$ be the port mapping produced by A at node v_1 given $F = \{(v_1, t = v_5)\}$ and let $\pi_{v_1}^t(\bot, F) = v_2$ w.l.o.g. Since A is perfectly resilient, we know by Lemma 4 that $\pi_{v_1}^t$ is a permutation over its relevant neighbors (all neighbors are relevant and must be in the same orbit). Necessarily, v_2 has a predecessor in its orbit in $\pi_{v_1}^t(\cdot, F)$. W.l.o.g. assume it is v_3: $\pi_{v_1}^t(v_3, F) = v_2$. Construct further sets of link failures as follows:

- $F_t = \{(t, v_1), (t, v_2), (t, v_3)\}$: leave only v_4 connected to the target.
- $F_{v_2} = \{(v_2, t), (v_2, v_4)\}$: make sure v_2 can only pass v_1's packets to v_3.
- $F_{v_3} = \{(v_3, v_4), (v_3, t)\}$: make sure v_3 can only pass v_2's packets to v_1.

Let $F_\emptyset = F \cup F_t \cup F_{v_2} \cup F_{v_3}$. Note that from the perspective of v_1, only the failure of (v_1, t) is visible, and hence F and $F_\emptyset$ are locally indistinguishable. Let us now construct the sequence of links traversed in A: at v_2, due to Lemma 4, the packet is necessarily forwarded to v_3. At v_3 it is necessarily forwarded to v_2 for the same reason. Since $F_\emptyset$ and F have the same impact on v_1's ports, we deduce $\pi_{v_1}^t(v_3, F) = \pi_{v_1}^t(v_3, F_\emptyset) = v_2$. The link (v_1, v_2) is thus used repeatedly, i.e., A causes a permanent loop. As (v_1, v_4) and $(v_4, t) \notin F_\emptyset$, v_1 and t are connected in $G \setminus F_\emptyset$, and yet A loops, leading to the desired contradiction.   ◀

▶ **Lemma 18.** *The complete bipartite graph with six nodes, three in each partition, does not allow for perfect resiliency, i.e., it holds that $A_p(K_{3,3}, t) = \emptyset$.*

Proof. We proceed similarly to the K_5 case. Let $V_1 = \{a, b, c = t\}, V_2 = \{v_1, v_2, v_3\}$ and $E = V_1 \times V_2$, where we assume w.l.o.g. that we start on a.

By contradiction, let $A \in A_p(K_{3,3})$. Let v_1 be the first target chosen by a. Let $\pi_a^t(\cdot, \emptyset)$ be the port mapping produced by A at node a given $F = \emptyset$. Since A is perfectly resilient, we

know by Lemma 4 that $\pi_a^t(\cdot, \emptyset)$ is a permutation over its relevant neighbors (all neighbors). Necessarily, v_1 has a predecessor in π_a. W.l.o.g. assume it is $v_2 : \pi_a^t(v_2, \emptyset) = v_1$.

We now construct a further set of link failures as follows, where we do not touch links incident to a:

- $F_t = \{(t, v_1), (t, v_2)\}$: leave only v_3 connected to the target.
- $F_b = \{(b, v_3)\}$: make sure b can only pass v_1's packets to v_2 (v_1, v_2 are relevant).

Set $F' = F_t \cup F_b$. Let us construct the sequence of links traversed in A: at v_1, due to Lemma 4, the packet is necessarily forwarded to b. At b it is necessarily forwarded to v_2 for the same reason. Since $F = \emptyset$ and F' have the same impact on a's forwarding function, we deduce $\pi_a^t(v_2, \emptyset) = \pi_a^t(v_2, F') = v_1$ As link (a, v_1) is used repeatedly, A thus causes a permanent loop. Since (a, v_3) and $(v_3, t) \notin F'$, a and t are connected in $G \setminus F'$, and yet A fails, which leads to the desired contradiction. ◀

We can now show that only planar graphs can permit perfect resilience:

▶ **Theorem 19** (Not planar implies no perfect resilience). *If G is not planar, then it does not support a perfectly resilient forwarding pattern π^t, i.e., $A_p(G, t) = \emptyset$.*

Proof. First, observe that both K_5 and $K_{3,3}$ do not support perfectly resilient schemes due to Lemmas 17 and 18. Next, Wagner's theorem [38] states that G planar $\Leftrightarrow (K_5 \notin min(G) \wedge K_{3,3} \notin min(G))$. The contrapositive form of Theorem 15 is $A(G', t) = \emptyset \Rightarrow A(G, t) = \emptyset$. Since perfect resilience is impossible on both K_5 and $K_{3,3}$, we deduce that no graph with K_5 or $K_{3,3}$ as a minor permits a perfectly resilient scheme. ◀

5   Negative Results

Our observations above also allow us to derive a number of additional impossibility results.

5.1   Boosting the Impossibility Result of Feigenbaum et al.

Feigenbaum et al. [19] gave a construction (see Figure 3) for which there is no perfect resilience forwarding pattern. We strengthen their result slightly, showing that no perfectly resilient forwarding pattern exists, even when the source is known. This allows us to boost their result to almost any resilience.
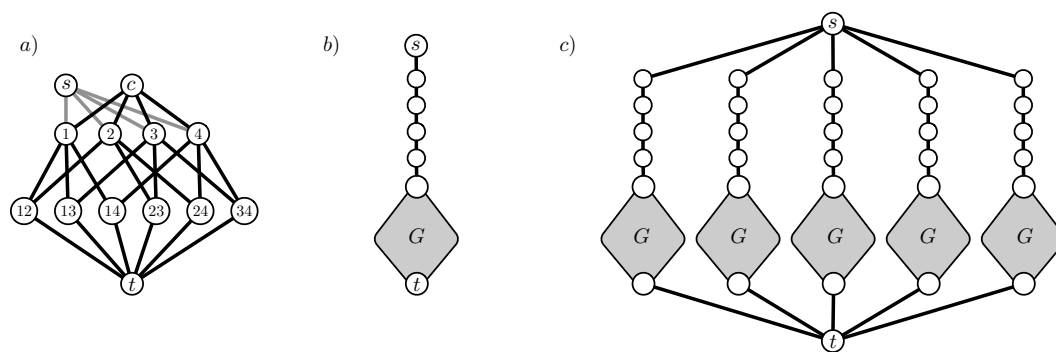
Figure 3 a) The original construction of Feigenbaum et al. [19] with an extra source node s. b) The padded construction for Theorem 21. G represents the gadget from Theorem 20. c) Padded and replicated construction from Theorem 22. Each gadget G is a copy of the gadget from Theorem 20.

▶ **Theorem 20.** *There exists a graph G on 13 nodes such that no forwarding pattern will succeed on G when source, target, and in-port are known.*

Proof of Theorem 20. Construct the gadget G as follows. Take four nodes 1, 2, 3, and 4 and connect them by a link to a center node c. Then, for each unordered pair (i, j) of nodes from $\{1, 2, 3, 4\}$ create a new node ij, and connect ij to i and j. Connect all ij to a new target node t. Finally, create a source node s and connect it to $\{1, 2, 3, 4\}$. G is illustrated in Figure 3a.

We will call nodes 1, 2, 3, and 4 *level one* nodes, and nodes 12, 13, 14, 23, 24, and 34 *level two* nodes.

We first claim that if nodes of levels one and two have degree 2 after failures, then they must *always* forward the packet coming from s, level one, or level two to the other port, with the exception of forwarding back towards s.

- If node vu on level two is connected to t, we can cut all other links to t and force vu to forward to t, as otherwise the forwarding fails.

- If uv is not connected to t, we can again create a unique path to t that goes through uv. Choose some vw on level two and cut all links to t except $\{vw, t\}$. In addition cut all links to vw except $\{v, vw\}$, and cut all links to v except $\{uv, v\}$. A packet coming from $\{u, uv\}$ must be forwarded to $\{uv, v\}$. The other direction is symmetric.

- If a level one node v is connected to s, then by setting $F = \{\{s, u\} : u \neq v\}$ any non-forwarding rule at v fails.

- If a level one node v is not connected to s, then it is connected to some level two nodes uv and wv or c and some level two node uv. In the first case, by cutting all links from level two to t except for uv or wv, and, respectively, the link $\{u, uv\}$ or the link $\{w, wv\}$, we force the unique path from s to t to take the links $\{vu, v\}$ and $\{wv, v\}$, in both directions. In the second case, we can create the following unique path from s to t by failing all other links: $(s, u, uv, v, c, w, wu, t)$, where w is some node not v or u. This forces forwarding at u from uw to c. For the other direction fail everything except the path (s, u, c, v, vw, t) for some w.

Any perfectly resilient forwarding pattern must therefore forward packets over degree-2 nodes.

We assume that there are no failures around c so the forwarding function $\pi_c^{s,t}$ remains fixed. In the following we identify the incident links of c with the neighbors they connect to. If $\pi_c^{s,t}(i) = i$ for any $i \in [4]$ the forwarding fails when s is connected only to i and i only to c. Assume without loss of generality that $\pi_c^{s,t}(1) = 2$. By having $\pi_c^{s,t}(2) = 1$ we have a loop when all other links incident to 2 fail and 1 has links $\{s, 1\}$ and $\{1, c\}$. Without loss of generality assume that $\pi_c^{s,t}(2) = 3$. By a similar argument we see that we must have $\pi_c^{s,t}(3) = 4$, as otherwise we would create a loop. Now consider different settings for $\pi_c^{s,t}(4)$.

- $\pi_c^{s,t}(4) = 1$: we fail all links incident to level one except $\{s, 4\}, \{1, 13\}, \{13, 3\}$, all links incident to c or 2. The forwarding pattern loops $(s, 4, c, 1, 13, 3, c, 4)$ and then either visits s or goes back to c immediately, finishing the loop.

- $\pi_c^{s,t}(4) = 2$ or $\pi_c^{s,t}(4) = 3$: These cases are similar, as we can force the packet to never visit 1, which could be the unique remaining path to t. Fail all links of s except $\{s, 4\}$, and all other links of 4 except $\{4, c\}$. In addition fail all links from 2 and 3 to level two. The packet will start a loop on $(s, 4, c, 2, c, 3, c, 4)$ or on $(s, 4, c, 3, c, 4)$, respectively. Then it will either visit s or go directly back to c, finishing the loop.

As all choices for $\pi_c^{s,t}(4)$ create a forwarding pattern with a loop, a perfectly resilient forwarding pattern cannot exist. ◀

By padding and replicating the construction it is possible to gain different parametrisations of the impossibility result. In particular, we observe that no $\omega(1)$-resilient forwarding pattern exists (as a function of m), and show that no $\Theta(f(n))$-resilient forwarding pattern exists even when the source and the target are $\Theta(f(n))$-connected. Theorems 21 and 22 are asymptotic in m, the number of links in the input graph before failures.

▶ **Theorem 21.** *There is no $\omega_m(1)$-resilient forwarding pattern with source, target, and in-port matching.*

Proof. Modify graph G as follows: replace the source node s with a path $P_k = (s_0, s_1, \ldots, s_k)$ on k nodes, and connect node s_k to the nodes 1, 2, 3, and 4. The node s_0 is the new designated source node. Denote this construction by G_k. See Figure 3b for an illustration.

Any forwarding pattern for G could simulate the existence of the path P_k. Since there is no forwarding pattern for G, there is no forwarding pattern for G_k. By setting k large enough we can make $|E(G)| = 18 < f(k)$ for any function $f(m) = \omega_m(1)$.   ◀

By a combination of padding and replicating the construction of Feigenbaum et al. it is possible to create a construction where the source and the target are connected by many link-disjoint paths after the failure, yet any forwarding pattern will fail.

▶ **Theorem 22.** *There is no $\Theta(f(m))$-resilient forwarding pattern with source, target, and in-port matching, for any $f(m)$ between $\Omega(1)$ and $O(m)$, even when there is a promise that there are $\Theta(f(m))$ link-disjoint paths between the source and the target after the failures.*

Proof. First observe that the gadget G contains $m_0 = 22$ links. The gadget G_k from Theorem 21 contains a total of $m_k = k + 22$ links.

The case of $f(m) = O(1)$ is covered by Theorem 20. Therefore fix $f(r)$ to be a function between $\omega_r(1)$ and r. We construct an infinite family of graphs as follows. For a fixed r, take $f = f(r)$ copies $G_g^{(1)}, G_g^{(2)}, \ldots, G_g^{(f)}$ of G_g for $g = g(r) = \lceil r/f(r) \rceil$. Then identify the nodes $s_0^{(1)}, \ldots, s_f^{(1)}$ as the same node, which is designated to be the source node. Finally, add a new designated target node t and connect it to the old target nodes $t^{(1)}, \ldots, t^{(f)}$ in each copy of the gadget. See Figure 3.c) for an illustration.

Since the links of the different gadgets can fail independently, and, by Theorem 20, for each $G^{(i)}$ and each partial forwarding pattern of $G^{(i)}$ there exists a partial failure set F that prevents the packet from being forwarded to the (old) target node $t^{(i)}$. By assigning the suitable failure sets to each gadget we can see that there exists a path between s and t through each gadget $G^{(i)}$ yet no packet will ever reach an old target node $t^{(i)}$ and therefore cannot reach the target node t.

By construction there are $\Theta(r)$ links and $\Theta(f(r))$ paths between s and t after failures. In each gadget at most a constant number of links fail (inside each original gadget $G^{(i)}$), and therefore the total number of failed links is $\Theta(f(r))$, as required.   ◀

5.2 Impossibility on Planar Graphs

In the previous section, we showed that non-planar graphs do not have resilient forwarding patterns. We now show that there are also planar graphs that do not permit perfect resiliency. However, as we will see later, for every outerplanar graph there is a perfectly resilient forwarding pattern, and there are also non-outerplanar planar graphs that allow for perfect resilience.

▶ **Theorem 23.** *There exists a planar graph G on 7 nodes such that no forwarding pattern π^t will succeed on G.*
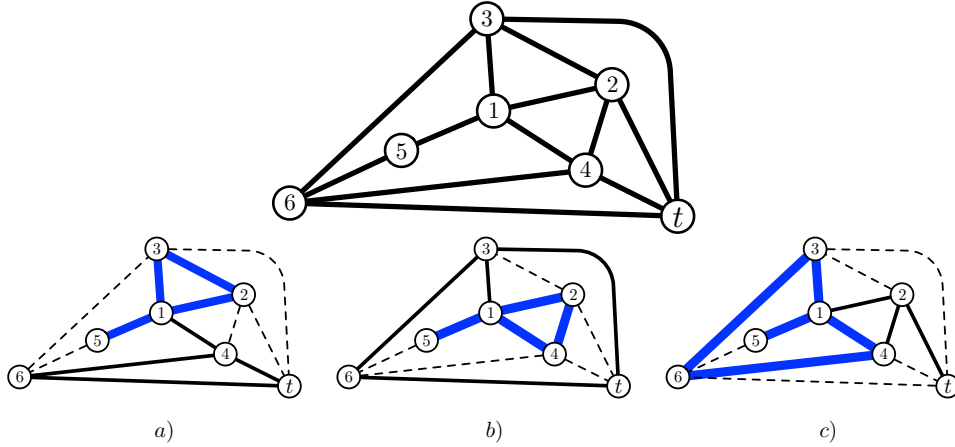
Figure 4 Planar graph with no perfectly resilient forwarding pattern. If the dashed links fail, in any forwarding pattern, packets will be stuck in one of the loops shown in bold in the bottom figure, even though there is at least one remaining path to the target.

Proof. Consider a packet emitted by node 5 for target t in the graph G depicted in Figure 4. In G all neighbors of all nodes are relevant if no failures occur. Thus the forwarding function at any node v $\pi_v^t(\cdot, \emptyset)$ must be a cyclic permutation of its neighbors due to Lemma 4. Since failures can only add relevant neighbors to nodes (Observation 2), any node of degree 2 of G after failures must forward packets with incoming port p to port $p' \neq p$ due to the same lemma. With this, we can show that there exists a failure set F that leads to a loop for all possible forwarding functions at node 1. We write $\pi_1^t(\cdot, \emptyset) = (v_0, v_1, v_2, v_3)$ to denote the permutation that assigns $v_{i+1 \mod 4}$ to v_i for $v_i \in \{2, 3, 4, 5\}$. We name the ports of node 1 by the identifiers of the neighbors they are connected to and we analyse the different forwarding permutations at node 1, based on the out-port they assign to packets arriving on in-port 5

Case (i) $\pi_1^t(5, \emptyset) = 2$. To ensure $\pi_1^t(\cdot, \emptyset)$ is a cyclic permutation under this constraint, a packet arriving on port 2 can only be forwarded to either 3 or 4. Thus the possible permutations are $(5, 2, 3, 4)$ and $(5, 2, 4, 3)$, which lead to loops under the failure sets illustrated in Figure 4.b and a.

Case (ii) $\pi_1^t(5, \emptyset) = 3$. To ensure $\pi_1^t(\cdot, \emptyset)$ is a cyclic permutation under this constraint, a packet arriving on port 4 can only be forwarded to either 2 or 5. Thus the possible permutations are $(5, 3, 4, 2)$ and $(5, 3, 2, 4)$ which lead to loops under the failure sets illustrated in Figure 4.a and c.

Case (iii) $\pi_1^t(5, \emptyset) = 4$. To ensure $\pi_1^t(\cdot, \emptyset)$ is a cyclic permutation under this constraint, a packet arriving on port 4 can only be forwarded to either 2 or 5. Thus the possible permutations are $(5, 4, 3, 2)$ and $(5, 4, 3, 2)$ which lead to loops under the failure sets illustrated in Figure 4.b and c.

Case (iv) $\pi_1^t(5, \emptyset) = 5$ There is no cyclic permutation with this assignment, thus there would be a loop according to Lemma 4.

Hence there is exists no forwarding pattern without a failure set that causes a loop. ◀

We can extend the proof with Lemma 5 and Corollary 7 to also include the source:

▶ **Theorem 24.** *There exists a planar graph G on 8 nodes such that no forwarding pattern $\pi^{s,t}$ will succeed on G.*

Proof. We note that by choosing the node 5 in Figure 4 as the source, the proof for Theorem 23 would directly carry over, if we could apply Lemma 4 to all nodes in Figure 4. However, Lemma 4 only considers target-based routing that does not consider the source, and we hence need to utilize Lemma 5 for node 1 (all relevant neighbors are in the same orbit) and Corollary 7 for the other nodes, where the degree two case suffices.

In order to apply Lemma 5 to node 1, we need to connect the new source s to two relevant neighbors of s, where we pick nodes 3 and 5, which keeps the 7+1 node construction planar. Note that for the actual packet path in Figure 4(a,b,c), we can fail the link $(s, 3)$, keeping $(s, 5)$ alive, and if 5 were to route back to s, s can only bounce the packet back.

It remains to force all degree 2 nodes in Figure 4(a,b,c) to relay incoming packets through the other port, via Corollary 7, which we prove by case distinction. To this end, for each of the seven cases (nodes 2,3 for a, 2,4 for b, and 3,4,6 for c), we need to show that there are pairs of node-disjoint paths for each such node i, with two different surviving links $(v_1, i), (v_2, i)$, that are in $E(G)$ minus all links incident to i, except $(v_1, i), (v_2, i)$:[2]

- start from s and end with (v_1, i), and start with (v_2, i) and end at t,
- start from s and end with (v_2, i), and start with (v_1, i) and end at t.

- Figure 4(a): Node 2 $[(3, 2), (2, 1)]$
  - $s - 5 - 6 - 3 - 2 : 2 - 4 - t$ and $s - 5 - 1 - 2 : 2 - 3 - t$
- Figure 4(a): Node 3 $[(1, 3), (3, 2)]$
  - $s - 5 - 1 - 3 : 3 - 2 - t$ and $s - 5 - 1 - 2 - 3 : 3 - 6 - t$
- Figure 4(b): Node 2 $[(1, 2), (2, 4)]$
  - $s - 5 - 1 - 2 : 2 - 4 - t$ and $s - 5 - 1 - 4 - 2 : 2 - 1 - t$
- Figure 4(b): Node 4 $[(2, 4), (4, 1)]$
  - $s - 5 - 1 - 4 : 4 - 2 - 3 - t$ and $s - 3 - 2 - 4^3 : 4 - 1 - 5 - 6 - t$
- Figure 4(c): Node 3 $[(1, 3), (3, 6)]$
  - $s - 5 - 1 - 3 : 3 - 6 - t$ and $s - 5 - 6 - 3 : 3 - 1 - 4 - t$
- Figure 4(c): Node 4 $[(1, 4), (4, 6)]$
  - $s - 5 - 1 - 4 : 4 - 6 - t$ and $s - 5 - 6 - 4 : 4 - 1 - 3 - t$
- Figure 4(c): Node 6 $[(3, 6), (6, 4)]$
  - $s - 5 - 1 - 3 - 6 : 6 - 4 - t$ and $s - 5 - 1 - 4 - 6 : 6 - 3 - t$

◀

6 Positive Results

Despite the numerous networks in which perfect resilience cannot be achieved, there are several interesting scenarios for which perfectly resilient algorithms exist. In this section, we present a particularly simple algorithm: the algorithm orders its neighbors arbitrarily, and reacts to link failures by simply "skipping over" the failed links according to this order. The exact forwarding function can hence be computed locally based on F, and the forwarding rules can be stored in linear space: a significant advantage in practice.

This motivates us to introduce the following definition:

---

[2] Recall that i is only aware of its incident link failures.   [3] Note that the packet is not actually routed via 3 in Figure 4(b), but the node 4 must also provision for this case in order to guarantee perfect resilience.

▶ **Definition 25** (Skipping Forwarding Functions). *Given a set S and a function $\pi\colon S \mapsto S'$ where $S' \subset S$, define the* tail *of s to be the sequence $(\pi(s), \pi(\pi(s)), \pi(\pi(\pi(s))), \dots)$ for each $s \in S$. We say that a forwarding function $\pi_v^{s,t}$ is* skipping *if there exists a bijection $f\colon E(v) \cup \{\bot\} \mapsto E(v)$ such that for each failure set F and each $e \in \{\bot\} \cup E(v) \setminus F$ we have that $\pi_v^t(e, F)$ equals the first element in the tail of e (with respect to f) that is not in F. A forwarding pattern is skipping if each of its forwarding functions is skipping.*

6.1   The Target is Close

Assume that a source s is close to the target t, even after the failures. In the following, we show that it is always possible to predefine conditional forwarding rules, with skipping, which ensure a route from s to t if their distance is at most two hops.

▶ **Theorem 26.** *For all graphs G there is a forwarding pattern, matching on the source, that succeeds if source s and target t are at distance at most 2 in $G \setminus F$.*

Proof. Resilience can be ensured with the following forwarding pattern. Let $V(s)$ be the set of neighbors of the source node s which are on a path of length two to the target t before the failures. We define the forwarding function $\pi_s^{s,t}$ of the source s as follows:

- We order the neighbors $V(s)$ of s arbitrarily, i.e., $V(s) = (v_1, v_2, \dots, v_k)$, and source s first tries to forward to the first neighbor v_j which is connected to it after the failures $(\pi_s^{s,t}(\bot) = v_j)$, i.e., in a skipping fashion.
- When a packet arrives on the in-port from v_i, for any i, s forwards the packet to v_j, the next neighbor in $V(s)$ such that $\{s, v_j\} \notin F$, i.e., skipping as well.

For all other nodes v, the forwarding function is defined as follows. When a packet arrives from the source s, send it to the target if the link has not failed. Otherwise return the packet to the source. Note that this is a skipping pattern, as we can predefine a skipping of (v, t) to route to s. This ensures that the source s tries all of its neighbors which could be on a path of length two to the target. Since we assumed such a path existed, the forwarding pattern must succeed in routing the packet to the target.   ◀

The algorithm relies on the source probing all neighbors if they are connected to the destination, which then succeed or return the packet. We can extend this idea to not require matching on the source, if the target is at most 2 hops away after failures. The idea is to assign unique identifiers to the nodes, converge to a 2-hop minimum identifier, and then forward similarly to the previous proof.

▶ **Theorem 27.** *There is a forwarding pattern, not matching on the source, that ensures a packet reaches its target t if all nodes are at most 2 hops apart from t after failures.*

Proof. Let us assign a unique ID to every node and construct the forwarding pattern based on these identifiers. The forwarding function of each node $v \in V \setminus \{s, t\}$ is defined as follows:
- If t is a neighbor of v, forward to t.
- If t is not a neighbor, then
  - if v does not have the lowest ID in its neighborhood, forward to the neighbor with lowest ID, i.e., skipping potential lost neighbors with lower ID,
  - if v has the lowest identifier in its neighborhood, then route according to a skipping pre-defined cyclic permutation of all neighbors.

Observe that, unless the target is reached on the way, the packet will reach a node w that has the lowest identifier in its two-hop neighborhood. From there on, the argument is analogous to the proof of Theorem 26: The node w will forward the packet to all its one-hop neighbors, which in turn bounce it back to w, unless connected to t. Since, by assumption, one of them is connected to t, the forwarding succeeds. ◄

6.2 Planar and Outerplanar Graphs

For outerplanar graphs there exists a planar embedding such that all nodes are part of the outer face and there is a walk along the links of the face that visits all nodes for connected graphs. This property holds also after arbitrary failures as long as the graph remains connected. Thus we can route along the links of the outer face of a planar graph using the well known right-hand rule [1, 5] despite failures.[4]

▶ **Theorem 28.** *Let $G = (V, E)$ be an outerplanar graph with $t \in V$. Then, there is a perfectly resilient skipping forwarding pattern π^t which does not require source matching.*

Proof. Fix an arbitrary outerplanar embedding of the graph without failures (a consistent embedding must be used on all nodes when constructing the forwarding pattern). Without loss of generality we define the canonical direction to be clockwise and we number the ports of the nodes accordingly starting from an arbitrary port.

Note that if a link l belongs to the outer face of a planar graph G, it also belongs to the outer face for all subgraphs of G, in particular also in $G \setminus F$ for $l \notin F$. Since all nodes belong to the outer face for an outerplanar graph, it is hence enough to demonstrate that the forwarding pattern ensures packets use only the links of the outer face and do not change the direction despite failures.

Given an in-port belonging to a link on the outer face of a graph in clockwise direction we can decide locally which out-port belongs to the outer face to continue the walk using the right-hand rule encoded in the the forwarding function with F.

The routing algorithm assigns the following forwarding functions to node v with degree d: $\pi_v^t(x) = x + 1 \bmod d$. Hence the forwarding function assigns out-port $x + 1 \bmod d$ to in-port x. When node s sends a packet to t it picks an arbitrary out-port that belongs to the outer face in the clockwise direction. When failures occur, the skipping method is used to derive the next usable output.

As we will see next, this is enough to encode the information of which links belong to the outer face of $G \setminus F$. To this end, we show that this forwarding pattern routes a packet along the links of the outer face for an arbitrary link failures F in clockwise direction.

We proceed to show this claim for the source and then for all other nodes on the path to the target by induction.

Each node has at least one pair of an incoming port and an outgoing port on the outerplanar face of G (for a node with degree one these two ports belong to the same link). Let (i_j, o_j) denote the jth such pair out of k for node v in G, starting to number them from port 0 at node v in clockwise order. Note that it holds for all these k pairs that $o_j = i_j + 1 \bmod d$.

---

[4] Such face routing was first considered over 20 years ago for ad-hoc networks by Kranakis et al. [29] and Bose et al. [8]. More involved face-routing algorithms have been devised and analyzed e.g. [26, 39] and go beyond the simple right-hand rule algorithm used for Theorem 28 and Corollary 29. They have mostly been studied without considering failures and use either state in packet or at nodes if source and target are not on the same face. We refer to some reference articles [30, 41] and book chapters [25, 28] for in-depth discussions and to the article by Behrend [3] for a historical overview.

Without failures, the source selects an outgoing port on the outer face in the correct direction by definition. Without loss of generality, let this out-port be o_0 from the pair (i_0, o_0) defined above. If the corresponding link has failed, the routing scheme will try port $o_0 + j \mod d$ for $j = 1, \ldots, d-1$ until successful according to the skipping forwarding pattern. Due to the failures, the corresponding link is now on the outer face and the statement thus holds for the base case.

For a subsequent node v on the walk to the target the packet will enter the node on port i_j for some j if no failures affect this node. In this case, the routing scheme selects $i_j + 1 \mod d$ as the out-port, which happens to be o_j as discussed above. Thus the packet remains using only links of the outer face travelling in the right direction on $G \setminus F$. If there has been a failure affecting node v, then the outgoing port might no longer be available. As shown for the source, iterating over the next ports clockwise will guarantee that the packet is still using a link of the outer face of $G \setminus F$. If the link incident to i_j is down, then v's neighbors will select another link and the packet might enter the node on a port that is not among the pairs for the graph without failures. Nevertheless, due to the arguments above this implies that the incoming port now is part of the walk along the links of the outer face in clockwise direction and also the outgoing port chosen subsequently is part of the links on the outer face of $G \setminus F$. Hence the routing scheme succeeds to lead packets to the target successfully without maintaining state in the packet or at the nodes.   ◀

Note that the face-routing pattern is even target-oblivious: starting on any node, it will visit every node. Moreover, while we have seen earlier that planar graphs do not offer perfect resilience, the approach guaranteeing perfect resilience for outerplanar graphs can also be applied for planar graphs if source and target are on the same face.

▶ **Corollary 29.** *Let $G = (V, E)$ be a planar graph where the packet starts on the same face as t. Then, there is a perfectly resilient skipping forwarding pattern π^t.*

Proof. The algorithm from the proof for Theorem 28 only needs to be adapted at the start. Since the source and the target might not be part of the outer face in G, the source chooses an arbitrary (clockwise direction) out-port that belongs to the face it shares with the target (some nodes may share several faces with the target, any of them can be chosen in this case). Since failures only reduce the number of faces, the shared face may contain more links after failures but not fewer and both source and target will always belong to the same face on $G \setminus F$, as long as the graph is still connected. Thus following the links of the face in clockwise direction will lead to the target for arbitrary non-disconnecting failures.   ◀

Another way of thinking about the proof is that we can also choose the embedding of the planar graph s.t. any desired face is the outer face, see e.g. Schnyder [36]. As before, note that our forwarding pattern can visit every node on the face despite failures.

6.3   A Non-Outerplanar Planar Graph with Perfect Resilience: K_4

So far, we have established that perfect resiliency is possible on outerplanar graphs as well as on the same face of planar graphs, and that it is impossible on some planar graphs and on all non-planar graphs. This raises the question if perfect resilience is possible on some non-outerplanar planar graphs, which we answer in the affirmative for the K_4: we employ forwarding along a cyclic permutation, unless the target is a neighbor.

▶ **Theorem 30.** *K_4 allows for perfectly resilient forwarding patterns π^t with skipping, i.e., not requiring to match on the source.*

Proof. Let the four nodes of the K_4 w.l.o.g. be $\{v_1, v_2, v_3, v_4 = t\}$. The node $v_4 = t$ must have at least one connected neighbor, w.l.o.g. v_3, where we are done if we start on v_3 or v_4.

It remains to fix the routing for any node not neighboring the target (where we directly reach t), for which we choose a cyclic permutation of all neighbors, picking skipping. Hence, if the packet starts on w.l.o.g. v_1, where v_1 is part of the connected component of t, and reaches v_3 or v_4, perfect resilience is achieved. We now perform a case distinction, where we note that while the node t is distinguishable as a neighbor, for v_1 it is unknown which of its neighbors is connected to t.

Assume v_1 has only one neighbor, then forwarding succeeds after two or three hops (a line). If v_1 has two neighbors and picks the one neighboring the target, we are done after two hops. Else, assume it picks the wrong one, say v_2. Then, if v_2 only has v_1 as a neighbor, the packet bounces back, then reaches v_3 and then t. Lastly, if v_2 is connected to t, we are done, and if not, it is connected to v_3, which then forwards the packet to t.   ◀

6.4   Graph-Families Closed under Link Subdivision

In this section we prove that under certain conditions, skipping forwarding functions retain any resilience guarantees. Our result applies to any graph family $\mathcal{G}$ that is closed under link subdivision. A family $\mathcal{G}$ is closed under link subdivision if for all $G \in \mathcal{G}$ the graph G' constructed by replacing a link (u, v) of G by a new node w, and two new links (u, w) and (w, v) is also in $\mathcal{G}$. For example planar and bounded genus graphs are closed under subdivision, but graphs of bounded diameter are not.

▶ **Theorem 31.** *Let $\mathcal{G}$ be a family of graphs that is closed under the subdivision of links. Assume that for each $G \in \mathcal{G}$ there exists a $f(m)$-resilient forwarding pattern (with or without source matching). Then there exists a $f(m)$-resilient skipping forwarding pattern with the same matching for each $G \in \mathcal{G}$, where f is any monotone increasing function.*

It follows that in many graph classes, it is sufficient to consider skipping forwarding patterns.

▶ **Corollary 32.** *In the following graph classes, there is a resilience-optimal skipping forwarding pattern: planar graphs, graphs of bounded genus, cycles, graphs of bounded maximum degree, and graphs of bounded arboricity.*

Proof of Theorem 31. Consider any $G \in \mathcal{G}$ and let $|E(G)| = m$. Consider a graph H that is constructed by subdividing each link $e = (u, v) \in E(G)$ into three links $e_u = (u, uv), e_v = (v, vu)$, and $e_{uv} = (vu, uv)$, where uv and vu are the two new nodes introduced in the subdivision. We call these nodes the *new nodes* and the other nodes *old nodes* of H. Let $M = |E(H)|$. We prove the theorem for forwarding patterns with source matching – the case with source matching is similar.

Now given a set of failed links $F \subseteq E$ on G, we can consider the corresponding failure set $F' = \{(uv, vu) \in E(H) : (u, v) \in F\}$ in H. Since $m \leq M$ and therefore $f(m) \leq f(M)$, there must exist a $f(M)$-resilient forwarding pattern ϕ^t for H and any $t \in V(H)$. Since only the middle links of the subdivided links fail, the forwarding functions of the old nodes must remain constant over all F' constructed from the failure sets F of G.

Before simulating ϕ^t on G, we take care of a technicality. We say that a link $(u, uv) \in E(H)$ is *cut* by ϕ^t if, for all F not containing $(u, uv), (uv, vu)$, or (vu, v), either $\phi^t_{uv}(u, F) = u$ or $\phi^t_{vu}(uv, F) = uv$. That is, uv or vu sends the packet from the direction of u back. In this case the packet cannot pass from u to v. We modify ϕ^t_u to ignore cut links: let S denote the set of cut links in $E(u)$. For each $e \in E(u)$ such that $\phi^t_u(e) \in S$ we set $\phi^t_u(e)$ to be the first element in the tail of e that is not in S.

Now let ϕ^t be such a modified forwarding pattern for any old node $t \in V(H)$. We construct a forwarding pattern π^t for G as follows. For each F, each $v \in V(G)$, and each e incident to v, set $\pi_v^t(e, F)$ to be the first element in the tail of e that is not in F. In the degenerate case where $e = E(v) \setminus F$ assign $\pi_v^t(e, F) = e$. Finally, if all links corresponding to non-failed links are cut, we set $\pi_v^t(e, F) = e$ for each non-failed cut link e. In this case the forwarding pattern will never reach v in H.

By construction the forwarding pattern π^t is skipping. Links corresponding to the cut links of H are never forwarded to and failed links are skipped. It remains to show that it is correct. Let $F \subsetneq E(G)$ be a failure set such that $|F| \le f(n)$. At each node $v \in V(H)$ with in-port e, the packet will travel the tail of e until it finds an out-port that has not failed (i.e. the middle link of the subdivided link does not correspond to a failed link of the original failure set F). The packet, due to the modification, will then travel over the subdivided links (v, vu), (vu, uv), and (uv, u) to some old node u. In the original graph, due to the construction of π^t, the packet with in-port e will directly be forwarded to (v, u). By an inductive argument for each initial node and failure set F, the sequence of nodes followed by the packet in $G \setminus F$ equals the sequence of *old nodes* followed by the packet in $H \setminus F'$.   ◄

7   Conclusion

We studied the fundamental question of when it is possible to provide perfect resilience in networks based on local decisions only. We provided both characterizations of infeasible instances and algorithms for robust networks for perfect and parametrized resilience.

While our results cover a significant part of the problem space, it remains to complete charting the landscape of the feasibility of perfect and parametrized resilience, both in the model where the source can and cannot be matched. Furthermore, we have so far focused on resilience only, and it would be interesting to account for additional metrics of the failover paths, such as their length and congestion.

References

1   H. Abelson and A. DiSessa. *Turtle Geometry: The Computer as a Medium for Exploring Mathematics.* MIT Press, Cambridge, MA, 1981.

2   Gregor Bankhamer, Robert Elsässer, and Stefan Schmid. Local fast rerouting with low congestion: A randomized approach. In *Proc. ICNP*, 2019.

3   Michael Behrend. How to solve a maze. *Caerdroia Journal*, (36):10–17, 2006. `http://www.labyrinthos.net/Caerdroia%2036%20HR.pdf`.

4   Petra Berenbrink, Ralf Klasing, Adrian Kosowski, Frederik Mallmann-Trenn, and Przemysław Uznański. Improved analysis of deterministic load-balancing schemes. *ACM Transactions on Algorithms*, 15(1):1–22, 2018.

5   J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications.* Elsevier, New York, 1976.

6   Michael Borokhovich, Liron Schiff, and Stefan Schmid. Provable data plane connectivity with local fast failover: Introducing openflow graph algorithms. In *Proc. ACM HotSDN*, 2014.

7   Michael Borokhovich and Stefan Schmid. How (not) to shoot in your foot with sdn local fast failover: A load-connectivity tradeoff. In *Proc. OPODIS*, 2013.

8   Prosenjit Bose, Pat Morin, Ivan Stojmenovic, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. In *DIAL-M*. ACM, 1999.

9   S Bryant, M Shand, and S Previdi. Ip fast reroute using not-via addresses. `https://datatracker.ietf.org/doc/draft-bryant-shand-ipfrr-notvia-addresses/`, 2006.

10   Costas Busch, Srikanth Surapaneni, and Srikanta Tirthapura. Analysis of link reversal routing algorithms for mobile ad hoc networks. In *Proc. SPAA*, 2003.

11 Marco Canini, Petr Kuznetsov, Dan Levin, and Stefan Schmid. A Distributed and Robust SDN Control Plane for Transactional Network Updates. In *Proc. INFOCOM*, 2015.

12 Marco Chiesa, Andrei V. Gurtov, Aleksander Madry, Slobodan Mitrovic, Ilya Nikolaevskiy, Michael Schapira, and Scott Shenker. On the resiliency of randomized routing against multiple edge failures. In *Proc. ICALP*, 2016.

13 Marco Chiesa, Andrzej Kamisinski, Jacek Rak, Gabor Retvari, and Stefan Schmid. Fast recovery mechanisms in the data plane. In *IEEE TechRxiv*, 2020.

14 Marco Chiesa, Ilya Nikolaevskiy, Slobodan Mitrovic, Andrei V. Gurtov, Aleksander Madry, Michael Schapira, and Scott Shenker. On the resiliency of static forwarding tables. *IEEE/ACM Trans. Netw.*, 25(2):1133–1146, 2017.

15 Marco Chiesa, Ilya Nikolaevskiy, Slobodan Mitrovic, Aurojit Panda, Andrei Gurtov, Aleksander Madry, Michael Schapira, and Scott Shenker. The quest for resilient (static) forwarding tables. In *Proc. IEEE INFOCOM*, 2016.

16 Marco Chiesa, Roshan Sedar, Gianni Antichi, Michael Borokhovich, Andrzej Kamisinski, Georgios Nikolaidis, and Stefan Schmid. Purr: A primitive for reconfigurable fast reroute. In *Proc. ACM CoNEXT*, 2019.

17 M Scott Corson and Anthony Ephremides. A distributed routing algorithm for mobile wireless networks. *Wireless netw.*, 1(1):61–81, 1995.

18 Theodore Elhourani, Abishek Gopalan, and Srinivasan Ramasubramanian. Ip fast rerouting for multi-link failures. In *Proc. IEEE INFOCOM*, 2014.

19 Joan Feigenbaum, Brighten Godfrey, Aurojit Panda, Michael Schapira, Scott Shenker, and Ankit Singla. Brief announcement: On the resilience of routing tables. In *Proc. PODC*, 2012.

20 Joan Feigenbaum, Brighten Godfrey, Aurojit Panda, Michael Schapira, Scott Shenker, and Ankit Singla. On the resilience of routing tables. *CoRR*, abs/1207.3732, 2012.

21 Klaus-Tycho Foerster, Andrzej Kamisiński, Yvonne-Anne Pignolet, Stefan Schmid, and Gilles Tredan. Bonsai: Efficient fast failover routing using small arborescences. In *Proc. DSN*, 2019.

22 Klaus-Tycho Foerster, Andrzej Kamisinski, Yvonne-Anne Pignolet, Stefan Schmid, and Gilles Tredan. Improved fast rerouting using postprocessing. In *Proc. SRDS*, 2019.

23 Klaus-Tycho Foerster, Yvonne Anne Pignolet, Stefan Schmid, and Gilles Trédan. Local fast failover routing with low stretch. *Computer Communication Review*, 48(1):35–41, 2018.

24 Klaus-Tycho Foerster, Yvonne-Anne Pignolet, Stefan Schmid, and Gilles Tredan. Casa: Congestion and stretch aware static fast rerouting. In *Proc. IEE INFOCOM*, 2019.

25 Hannes Frey and Ivan Stojmenović. *Geographic and Energy-Aware Routing in Sensor Networks*, chapter 12, pages 381–415. John Wiley & Sons, Ltd, 2005.

26 Hannes Frey and Ivan Stojmenovic. On delivery guarantees and worst-case forwarding bounds of elementary face routing components in ad hoc and sensor networks. *IEEE Trans. Computers*, 59(9):1224–1238, 2010.

27 E.M. Gafni and D.P. Bertsekas. Distributed algorithms for generating loop-free routes in networks with frequently changing topology. *Trans. Commun.*, 29(1):11–18, 1981.

28 Holger Karl and Andreas Willig. *Routing Protocols*, chapter 11, pages 289–329. John Wiley & Sons, Ltd, 2006.

29 Evangelos Kranakis, Harvinder Singh, and Jorge Urrutia. Compass routing on geometric networks. In *Proc. CCCG*, 1999.

30 Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. An algorithmic approach to geographic routing in ad hoc and sensor networks. *IEEE/ACM Trans. Netw.*, 16(1):51–62, 2008.

31 Karthik Lakshminarayanan, Matthew Caesar, Murali Rangan, Tom Anderson, Scott Shenker, and Ion Stoica. Achieving convergence-free routing using failure-carrying packets. In *Proc. ACM SIGCOMM*, 2007.

32 Junda Liu, Aurojit Panda, Ankit Singla, Brighten Godfrey, Michael Schapira, and Scott Shenker. Ensuring connectivity via data plane mechanisms. In *Proc. USENIX NSDI*, 2013.

33 Grzegorz Malewicz, Alexander Russell, and Alexander A Shvartsman. Distributed scheduling for disconnected cooperation. *Distributed Computing*, 18(6):409–420, 2006.

34 Yvonne-Anne Pignolet, Stefan Schmid, and Gilles Tredan. Load-optimal local fast rerouting for dependable networks. In *Proc. DSN*, 2017.

35 Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4):17:1–17:24, 2008.

36 Walter Schnyder. Embedding planar graphs on the grid. In *Proc. SODA*, 1990.

37 Brent Stephens, Alan L Cox, and Scott Rixner. Scalable multi-failure fast failover via forwarding table compression. In *Proc. ACM SOSR*, 2016.

38 Klaus Wagner. Ueber eine eigenschaft der ebenen komplexe. *Mathematische Annalen*, 114:570–590, 1937.

39 Mirjam Wattenhofer, Roger Wattenhofer, and Peter Widmayer. Geometric routing without geometry. In *Proc. SIROCCO*, 2005.

40 Baohua Yang, Junda Liu, Scott Shenker, Jun Li, and Kai Zheng. Keep forwarding: Towards k-link failure resilient routing. In *Proc. IEEE INFOCOM*, 2014.

41 Aaron Zollinger. Geographic routing. In *Encyclopedia of Algorithms*, pages 825–828. 2016.