On the Price of Locality in Static Fast Rerouting

***Abstract*—Modern communication networks feature fully decentralized flow rerouting mechanisms which allow them to quickly react to link failures. This paper revisits the fundamental algorithmic problem underlying such local fast rerouting mechanisms. Is it possible to achieve *perfect resilience*, i.e., to define local routing tables which preserve connectivity as long as the underlying network is still connected? Feigenbaum et al. (ACM PODC'12) and Foerster et al. (SIAM APOCS'21) showed that, unfortunately, it is impossible in general.**

This paper charts a more complete landscape of the feasibility of perfect resilience. We first show a perhaps surprisingly large price of locality in static fast rerouting mechanisms: even when source and destination remain connected by a linear number of link-disjoint paths after link failures, local rerouting algorithms cannot find any of them which leads to a disconnection on the routing level. This motivates us to study resilience in graphs which exclude certain dense minors, such as cliques or a complete bipartite graphs, and in particular, provide characterizations of the possibility of perfect resilience in different routing models. We provide further insights into the price of locality by showing impossibility results for few failures and investigate perfect resilience on Topology Zoo networks.

Regular submission.

I. Introduction

Traditional communication networks can be modelled as distributed systems in which routers cooperate to compute efficient routes. In particular, using protocols based on link state or distance vector algorithms, routers can—in a distributed manner—compute routing tables which induce shortest paths [1]. These protocols can also naturally cope with failures: whenever one or multiple links fail, the distributed routing protocol is simply invoked again, triggered by the nodes incident to a failed link. The protocols are hence in some sense "*perfectly resilient*" [2]: After reconvergence, the protocol re-establishes a path between any pair of nodes still physically connected, by dynamically updating their routing tables. Unfortunately, however, the recomputation and dynamic update of routing tables comes at the cost of slow reaction time [3].

Modern dependable communication networks hence additionally feature fully decentralized flow rerouting mechanisms which rely on *static* routing tables and allow to react to link failures orders of magnitudes faster than traditional networks [3]. Rather than invoking the distributed routing protocol when detecting a failure, these static fast rerouting mechanisms allow to predefine conditional failover rules at each router: these rules can depend only on *local* information at a node v, and can hence be conditioned on the status of links incident to v or the header of packets arriving at v, but not on failures in other parts of the network. While this enables a very fast reaction, it raises the question of how such local rules can be defined to maintain a high resilience under multiple link failures. Feigenbaum et al. [2] showed that achieving a perfect resilience using static fast rerouting mechanisms is unfortunately impossible in general: the authors presented an example network in which it is not possible to predefine local failover rules which ensure that as long as the underlying graph is connected, the routing tables induce a valid routing path to the destination. In other words, there is a *price of locality*: local fast rerouting comes at a cost of reduced resilience under multiple link failures.

This paper provides a systematic analysis aiming to characterize the feasibility of perfect resilience using static fast rerouting, motivated by Feigenbaum et al.'s counterexample. Indeed, their work raises a number of interesting research questions, such as:

- How significant is the price of locality? Is it at least possible to compute local failover rules which ensure connectivity on the routing level if the underlying network remains highly connected after the link failures?
- How does the resilience depend on the model? What happens if we include the promise of high connectivity or few failures, respectively, if we aim for smaller routing tables and do not match on the packet source or not even on the destination—where are the boundaries between working algorithms and impossibility? This question is particularly interesting in the light of emerging software-defined networks which allow routers to match different parts of the header and hence implement different routing models.

A. Contributions

This paper aims to chart a more complete picture of the feasibility of perfect resilience with local fast rerouting, focusing on the most fundamental aspect: reachability. We first show a perhaps surprisingly general negative result: even when a large number of link-disjoint paths survive after link failures, local failover routes cannot leverage them to reach the destination. Specifically, we prove the following price of locality: even if we are promised that there remain $\Omega(n)$ disjoint paths between source and destination after failures (we refer to this scenario as *r-tolerant* where $r = \Omega(n)$), it is impossible to pre-define static routing tables ahead of time which ensure connectivity without knowing these failures; here n refers to the # of nodes (§III). Prior work only showed impossibility for 1-tolerance and left higher connectivity guarantees to future work.

Motivated by this result, we study the feasibility of perfect resilience in graphs which exclude certain dense minors, such as cliques or a complete bipartite graphs. We present an almost optimal characterization of resilience in the different models. First, for a model in which routers can match both the source and the destination of a packet, we show that perfect resilience

Model	**Subgraph**	**Minor**	**Possible**	**Impossible**
r-Tolerance: $r > 1$ [§II]	Yes [§III-B]	No [§III-C]	K_{2r+1} / $K_{2r-1,2r-1}$ [§III-C]	K_{5r+3} [§III-B]
Bounded # link failures f [§VI]	No [§VI]	No [§VI]	$K_n : f < n-1$ [4, §B.2]	$K_n, n > 8 : f \geq 6n - 33$ [§VI]
			$K_{a,b} : f < \min\{a,b\} - 1$ [4, §B.3]	$K_{a,b}, a, b \geq 4 : f \geq a + 4b - 21$ [§VI]

TABLE I
LANDSCAPE OF FEASIBILITY OF LOCAL FAST REROUTING IN DIFFERENT FAILURE MODELS.

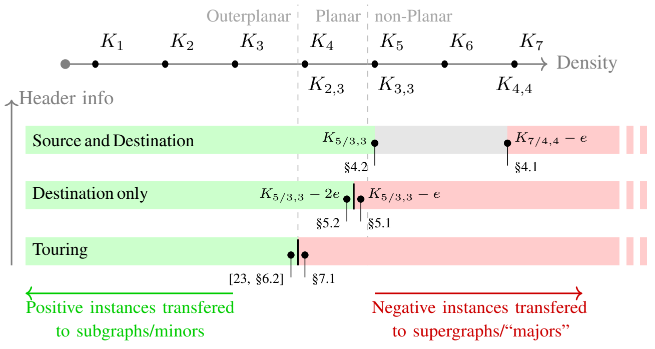

Fig. 1. Feasibility landscape of local fast rerouting in different routing models.

is impossible on *any* graph which has a minor K_7 or a minor $K_{4,4}$ which misses one link, but possible on K_5 and $K_{3,3}$ networks and their minors (§IV).[1]

In a model where routing rules can only match the packet destination, it is impossible to achieve perfect resilience on networks with minors K_5 and $K_{3,3}$ which miss one link; this characterization is complete in the sense that we can show that perfect resilience is always possible on K_5 and $K_{3,3}$ networks which only miss two links, and their minors (§V).

We also study the price of locality in scenarios in which the number of link failures is bounded (§VI) as well as in scenarios in which the local routing rules do not even depend on the destination but where a packet needs to tour the entire network, i.e., visit all nodes under failures (rather than routing to a specific destination); we provide an exact characterization of perfect resilience in this model as well, touring is possible if and only if G is outerplanar (§VII).

Table I and Figure 1 summarize our classification results.

Lastly, we also perform a small case study in §VIII on more than 250 Topology Zoo networks: around a third of all networks allow for perfect resilience in all models, while the classification of the remaining topologies depends on the routing model considered. For destination-based routing, our contributions allow us to to classify more than 30% additional topologies than with previous results.

B. Background and Related Work

The question of how to provide resilient routing in networks is a fundamental one and has been explored intensively in the literature already [3]. However, many existing approaches require dynamic routing tables [5]–[7] which implies slow reaction times [3], or the ability to rewrite or extend packet headers which introduces overheads and is not always possible [8], [9]. Our requirement of static failover tables and immutable headers also rules out the application of graph exploration techniques such as [10]–[13] or the use of rotor routers [14]–[16]. Also classic routing algorithms for sensor networks, such as geographic routing [17]–[21], require memory and are hence not applicable in our context. Furthermore, while there exist graph exploration algorithms which do not require any memory, e.g., for mazes consisting of a single wall (see e.g., the well-known right-hand rule [22]), these algorithms are transferrable only (if at all) to very simple graphs such as outerplanar graphs [23]. Our model hence assumes an interesting new position in the problem space: while it is not possible to use dynamic memory during routing (neither in the packet header nor in the routing table), it is possible to pre-process conditional routing rules ahead of time, without knowing the actual failure scenarios.

The model considered in this paper was introduced by Feigenbaum et al. [2], [24] and, in a slightly more restricted version, by Borokhovich et al. [25] in parallel work. While there has been interesting applied work on this problem, e.g., [26]–[29], in the following, we will focus on related works providing theoretical insights.

Several interesting results are due to Chiesa et al. who presented a technique which relies on a decomposition of the network into arc-disjoint arborescence covers [30]–[32]: any k-connected graph can be decomposed into a set of k directed spanning trees [33] (rooted at the same node, the destination) such that no pair of spanning trees shares a link in the same direction. This allows to route packets along some arborescence until hitting a failure, after which the packet can be rerouted along a different arborescence. This technique is particularly well-suited to provide a weaker notion of resilience, known as *ideal resilience* [32], which is defined for k-connected graphs (while the notion of perfect resilience applies to arbitrary graphs): given a k-connected network, static failover tables are called ideally resilient if they can tolerate any set of $k-1$ link failures. While Chiesa et al.'s paper already led to several follow up works [34]–[37], it remains an open question whether ideal resilience can be achieved in general k-connected graphs.

As mentioned above, already Feigenbaum et al. [2], [24] proved that perfect resilience is impossible to achieve in general, by presenting a counterexample with 12 nodes. Foerster et al. [23] recently generalized this negative result by showing that it is impossible to achieve perfect resilience on any non-planar graph; furthermore, planarity is also not sufficient for perfect resilience. On the positive side, [23] showed that perfect

[1] A K_n is a complete graph with n nodes, whereas a $K_{a,b}$ is a complete bipartite graph with a respectively b nodes in its two partitions.

resilience can always be achieved in outerplanar graphs, and also initiated the study of routing rules which can depend on the source. In this paper, we significantly extend these results along several dimensions.

C. Overview

The remainder of this paper is organized as follows. We introduce our formal model in §II. In §III, we show that maintaining connectivity with local failover rules is challenging already in highly connected graphs, and even if routing rules can depend on the source. This motivates us to study perfect resilience on graphs with dense minors, in a model where routing tables can (§IV) or cannot (§V) depend on the source. We then investigate the problem of perfect resilience under a bounded number of link failures (§VI) and study a novel failover model, where routing cannot depend on source and destination but where a packet needs to visit the entire graph (§VII). In §VIII we then perform a case study on Topology Zoo networks to classify them w.r.t. perfect resilience. We conclude our contribution and discuss future directions in §IX.

II. Model

We are given a communication network which we model as an undirected graph $G = (V, E)$, where the nodes represent *routers* that are connected via *links* E. 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; if clear from the context, we will omit the subscript G. We will also write $V(G)$ and $E(G)$ for the nodes V, respectively links E, of a graph $G = (V, E)$. When talking about connectivity, we always refer to link connectivity, *i.e.*, two nodes $v, w \ \in V(G)$ are k-connected if there are k paths between v and w that do not share any links, such paths are also called link-disjoint paths. The notations K_n and $K_{a,b}$ refer to the complete graph with n nodes, respectively the complete bipartite graph with a and b nodes in its partitions. For the latter notations, when adding the superscript $-c$, i.e., K_n^{-c} and $K_{a,b}^{-c}$, we remove c links from the respective graphs.

The network is subject to link failures, which however are not known ahead of time, when the routers are configured. We will refer to the set of links which will fail by $F \subset E$; failures are undirected. The graph G without links F is denoted by $G \setminus F := G(V, E \setminus F)$. Similarly, $G \setminus E'$ and $G \setminus V'$ denote the graph G without the set of links in $E' \subset E$, respectively, the graph G without the set of nodes $V' \subset V$ and their incident links.

Each node $v \in V$ is configured with a local forwarding function $\pi(v)$, essentially a *forwarding table*. This forwarding table (or synonymously, *routing table*[2]) is essentially a set of forwarding rules which include conditional failover rules that depend on the incident link failures. Specifically, the rules $\pi(v)$ of node v can depend on (a subset of) the following information:

- the set of incident failed links $F \cap E(v)$
- the source s of the to-be-forwarded packet at v
- the destination t of the to-be-forwarded packet at v
- the incoming port (*in-port*) from which the packet arrives at v

[2] While forwarding table is the technically correct term, we will use the term interchangeably with the term routing table.

In this paper we aim to chart a landscape of resiliency results for different models, and we hence consider multiple combinations of the above information. However, all these models have in common that the routing table is pre-configured and static, and forwarding rules do not change the packet header.

A local routing algorithm is hence simply a *forwarding function* π_v for each node v. For example, in the most general model where all information can be accounted for, given a graph G and a destination $t \in V(G)$, the function is

$$\pi_v : \quad 2^{E(v)} \times V \times V \times E(v) \cup \{\perp\} \mapsto E(v)$$

at each node $v \in V(G)$, where $\perp$ represents the empty in-port, i.e. the starting node of the packet. In other words, given the set of failed links $F \cap E(v)$ incident to a node v, the source and the destination, as well as the in-port, the forwarding function π_v maps each incoming port (link) $e = (u, v)$ to the corresponding outgoing port (link). We will call the union of the forwarding functions $\pi = (\pi_v)_{v \in V}$ the *forwarding pattern*, or simply the *routing*. In the following, we will use the notation

$$\pi_v^{s,t}(e, F) \qquad \text{resp.} \qquad \pi_v^{s,t}(u, F)$$

to denote the link to which a packet arriving at v via the link $e = (u, v)$ will be forwarded, given a failure set F and in a model where the rule matches both source s and destination t. We will refer to these types of rules as *source-destination-based routing*. Similarly, we will use the notation $\pi_v^t(e, F)$ resp. $\pi_v^t(u, F)$ to denote the link to which a packet arriving at v from a link $e = (u, v)$ will be forwarded, given a failure set F and in a model where the rule matches only the destination t. We refer to these types of rules as *destination-based routing*. Note that we do not require these forwarding patterns to follow some sort of cyclic permutation of the out-ports for neither of the routing flavours.

We will call a forwarding pattern π *r-resilient* if for all G and all F, where $|F| \leq r$, the forwarding pattern routes the packet from all $v \in V$ to any destination t when v and t are connected in $G \setminus F$. Note that the restriction that source and destination must remain connected when removing the links in the failure set F implies that the connectivity of the graph does not play a big role. E.g., consider a graph G which consists of a cliques of size l an one extra node connected to the clique with one link. While the connectivity of G is one, it is easy to construct forwarding patterns that tolerate two failures for packets originating from the extra node if the remaining graph stays connected. A forwarding pattern is *perfectly resilient* if it is ∞-resilient: the forwarding always succeeds in the connected component of the destination, for all destinations. Let $A_p(G, s, t)$ be the set of such perfectly resilient patterns (algorithms), respectively $A_p(G, t)$, $A_p(G)$ for the different models depending only on the destination or not even that; we abbreviate these versions by A_p when the context is clear.

To explore the achievable resilience of local fast rerouting algorithms beyond perfect resilience, we in this paper are also interested in a relaxed notion of resiliency, where we are given the promise of high connectivity after failures:

Definition 1 (r-tolerant)**.** *A forwarding pattern $\pi^{s,t}$ is called r-tolerant on a graph G, if it can guarantee reaching the destination t from source s under the assumption that s and t remain r-connected under failures.*

Observe that $r = 1$-tolerance corresponds to perfect resilience and that, for $r < r'$, if we have r-tolerance, we also obtain r'-tolerance: the failure sets $\mathcal{F}_r$ that retain r-connectivity are a superset of the failure sets $\mathcal{F}_{r'}$ that retain r'-connectivity. For example, a perfectly resilient algorithm (1-tolerant) is also 2-tolerant. Note that for a k-connected we can hope to achieve at most r-tolerance.

III. On the Price of Locality

Before studying perfect resilience in detail, we first consider a weaker notion of resilience: the design of local rerouting functions for scenarios where the connectivity remains larger than one after failures. We derive a surprisingly strong negative result on what can be achieved with static fast rerouting:

It is generally *impossible* to be $\Omega(n)$-tolerant, even when forwarding rules can depend on both source and destination.

Prior work just showed that 1-tolerance is impossible in general, but the details for higher connectivity promises were left unanswered. Indeed, at first it seems that if we are guaranteed that a linear number of paths exist after failures between source and destination, then surely fast failover mechanisms should be able to leverage this high connectivity. However, we show next that this intuition is false.

A. Intuition and Example

Intuitively, the more highly connected the topology is after failures, the easier it should be to ensure connectivity also with local static rerouting. However, as we will illustrate here on complete networks, this additional topological connectivity is only marginally useful. Concretely, while an r-tolerant algorithm in principle has more flexibility, in the sense that it can afford to *not* explore a certain route at all (as there are for sure alternative routes), and hence e.g., avoid potential loops, this additional connectivity is hard to exploit *locally*: such a choice can only be made $r-1$ times for an r-tolerant algorithm, among *all nodes* in the graph. In other words, the flexibility is restricted globally, while decision making is inherently local. We refer to Fig. 2 for an illustration.

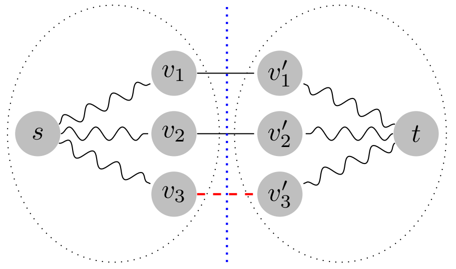

Fig. 2. After failures (in dashed red), s and t remain 2-connected, as there are two crossings across the blue dotted cut. Assume v_1 does not forward to v_1' and v_2 does not forward to v_2'. Then, as the link between v_3 and v_3' has failed, it is impossible to reach t from s. Note that from a local point of view, v_1, v_2, v_3 are unaware of the failures at each other, and hence to guarantee 2-tolerance, at least two of v_1, v_2, v_3 must forward across the blue dotted cut, to their respective v_1', v_2', v_3', if still possible after failures.

B. Impossibility of r-Tolerance in General

We show the following general impossibility result.

Theorem 1. *Let $r \in \mathbb{N}$. The complete graph with $3+5r$ nodes does not allow for an r-tolerant forwarding pattern $\pi^{s,t}$.*

Proof: From K_{3+5r} we choose 5 nodes $V_5 = \{v_1, v_2, v_3, v_4, v_5\}$ not including s and t. Consider all triples $a, b, c \in V_5$, such that, if b has a degree 2 after failures (connected to only a, c), then b will not forward a packet from a to c. If such a triple exists, then leave the path $s-a-b-c-t$ intact after failures and remove all other links of a,b, and c. We have constructed a partial failure set and a path from source to target that is not used by the forwarding pattern under this partial failure set.

If such a triple does not exist, then all nodes in V_5, with degree 2 after failures, will route in a permutation, assuming their neighbors are from V_5. Without loss of generality (W.l.o.g.), leave the path $s - v_1 - v_2$ intact after failures, but fail all other links of v_1. Then, for v_2, leave only the links to v_3, v_4, v_5 alive after failures. If the routing of v_2, coming from v_1, does not enter a permutation containing all 4 neighbors, with $v_i \neq v_1$ missing, then we fail all links incident to v_3, v_4, and v_5, except the links to v_2 and the link between v_i and t. Now, the packet coming from v_1 to v_2 will not reach v_i, and hence we lose one path to destination. On the other hand, if $v_i = v_1$ is missing, then we fail all links incident to v_3, v_4, and v_5 and the packet is trapped in the 5-node construction without returning to s via v_1.

Else the routing is a cyclic permutation. Assume w.l.o.g. the cyclic ordering for v_2 is v_1, v_3, v_4, v_5. We then fail all links incident to v_3, v_4, and v_5, except the link to v_2 and the links (v_4, t) and (v_2, v_5). Now, the packet is routed $s - v_1 - v_2 - v_3 - v_5 - v_2$. The packet will then go to v_1 and start a loop – we lose one path to the destination, namely via v_4.

We repeat above construction r times in total, always picking a new set of 5 nodes. Each time we either lose one path to the destination or find a routing loop. If we lose r paths, then the construction is complete, but we also need to consider the case where we are trapped in a routing loop in a 5-node gadget, as then the st-connectivity is $r-1$. To this end, we use the one remaining node v from K_{3+5r}, and leave it connected to s, but fail all its other incident links except (v, t). W.l.o.g., we can assume that v is last in the visiting order of s. If we lose r paths, then we disconnect v from t and hence the packet loops permanently, as none of the other 5 node constructions allow passing to the destination. If we do not lose r paths, then the path $s - v - t$ restores st-connectivity to r, as promised, but the packet loops in one of the 5 node constructions. ∎

Note that r-tolerance is preserved under iteratively taking subgraphs, *i.e.*, if G allows for r-tolerance, then every $G' \subset G$ allows for r-tolerance as well. The reason is that we can obtain G' as a component of G by failing the missing links.

Corollary 1. *Let $r \in \mathbb{N}$. If a graph G has K_{3+5r} as a subgraph, then G has no r-tolerant forwarding pattern $\pi^{s,t}$.*

C. r-Tolerance and Minors

Even though r-tolerance is preserved under taking subgraphs, we next show that r-tolerance is not preserved for graph minors, for all $r \geq 2$. This is in contrast to the result of Foerster et al. [23], who showed that it is preserved for $r = 1$.

Theorem 2. *For each $r \in \mathbb{N}$: $r \geq 2$ holds: There exists an r-tolerant graph G with a minor G' that is not r-tolerant.*

Proof: Given parameter $r > 1$, we take the construction from Theorem 1 as G' and now build a graph G, s.t. G is r-tolerant and G' is a minor of G. Given G', add a new source node s', connect it with $r - 1$ paths to s, and add the link (s', t). An algorithm that is r-tolerant on this new graph simply routes from s' to t via the direct link; if that link fails, the r-tolerance promise does not hold. Now, observe that the graph construction from Theorem 1 is a minor of the above construction (obtained by merging s', s, as well as the paths between them, and removing the link between s' and t), i.e., the existence of an r-tolerant forwarding pattern does not imply the existence of an r-tolerant forwarding pattern for minors, for any $r \geq 2$. ■

We note that if s and t are less than r-connected before the failures occur, r-tolerance trivially holds: r-tolerance is a promise problem, only considered under high connectivity. On complete graphs, r-tolerance is also trivial for K_{r+1}, as a removal of the source-destination link removes the promise of r-connectivity. We can slightly extend this result:

Theorem 3. *For each $r \in \mathbb{N}$ K_{2r+1} admits r-tolerance.*

Proof: Foerster et al. [23, Theorem 6.1] showed that perfect resilience can be maintained if source and destination have distance at most two after failures, and we now leverage their algorithm. Assume that the link between source and destination fails on K_{2r+1} – else the statement holds by routing in a single direct hop. When source s and destination t on K_{2r+1} remain r-connected, then s is connected to at least r neighbors V_1 different from t and t is connected to at least r neighbors V_2 different from s. Besides source and destination, K_{2r+1} has only $2r - 1$ nodes, and hence $|V_1 \cap V_2| \geq 1$, *i.e.*, a path of length 2 exists between source s and destination t. ■

We next briefly investigate complete bipartite graphs. If source and destination are in the same part, then the distance-2 algorithm [23, Theorem 6.1] applies if the other part has at most $2r - 1$ nodes, as in the proof above. If source and destination are in different parts, then the distance-2 algorithm is no longer directly applicable, as every route besides the direct source-destination link has a length of at least 3. However, we can extend the distance-2 forwarding pattern to distance 3 in complete bipartite graphs.

Theorem 4. *For all bipartite graphs G there is a forwarding pattern $\pi^{s,t}$ that can guarantee reaching the destination t from source s, if s and t are at distance at most 3 in $G \setminus F$.*

Proof: First, whenever the destination is a neighbor, we route to it, as highest priority. Else, the source and each neighbor of the source routes in a cyclic permutation. If a node is not the source or a neighbor of the source, then the packet bounces back (distance to source = 2). We only visit a node v of distance 3 if $v = t$. Moreover, if the st-distance is at most 3, we will also reach t from s, as (without the hop to the destination or stopping when finding the destination), our algorithm traverses all links E_1 incident to the destination and all links E_2 adjacent to those in E_1, *i.e.*, t is found with a distance of at most 2, and if the distance is exactly 3, the last link is adjacent to a link in E_2. ■

Applying Theorem 4 to complete bipartite graph yields:

Theorem 5. *For each $r \in \mathbb{N}$ $K_{2r-1,2r-1}$ admits r-tolerance.*

Proof: Let A and B the two parts of the bipartite graph $K_{2r-1,2r-1}$. Assume w.l.o.g $s \in A$. We now perform a case distinction whether $t \in A$. We start with the case where this is true. Due to r-connectivity after failures, the source s retains at least r neighbors V_s in B, and the destination t retains at least r neighbors V_t in B as well. Hence, $|V_s \cap V_t| \geq 1$ due to B having at most $2r - 1$ nodes, *i.e.*, a route of length 2 exists between s and t.

We next consider the remaining case where w.l.o.g. the destination is in B the second part. If the link (s, t) exists we are done immediately and hence assume it has failed. Else, s has at least r neighbors V_s in B and t has at least r neighbors V_t in A. Pick $u \in V_s$: u has at least $r - 1$ neighbors V_u in $A - \{s\}$. Since $|A - \{s\}| = 2r - 2$, $V_t \cap (A - \{s\})$ (of size r) and $V_u \cap (A - \{s\})$ (of size $r - 1$) necessarily intersect. Let w a node in this intersection: w is neighbor of both u and t, hence source and destination have a distance of at most 3 via the path $s - u - w - t$. ■

We recall that r-tolerance is preserved for all subgraphs and obtain the following corollary:

Corollary 2. *For each $r \in \mathbb{N}$ it holds that K_{2r+1} and $K_{2r-1,2r-1}$ and all their respective subgraphs admit r-tolerance.*

IV. Perfect Resilience with Source

Given our insights on the feasibility of local fast rerouting in more highly connected graphs, we now turn to studying perfect resilience: resilience in scenarios where arbitrary links can fail, as long as the graph remains connected. Recall that we aim to chart a landscape of perfect resilience in this paper, and in this section, we start analyzing a model where routing rules can depend both on the source and the destination of a packet. In the next section, we will then consider the scenario where rules can only depend on the destination. Given the result of the previous section, our characterization will revolve around graphs which feature dense minors before failures occur.

A. Impossibility Results

We first show that it is impossible to achieve perfect resilience on complete graphs with seven nodes.

Theorem 6. *The complete graph with seven nodes, minus one link, does not allow perfect resilience, i.e.,* $A_p(K_7^{-1}, s, t) = \emptyset$.

Note that when considering perfect resilience, it is at most as hard to route in a subgraph as one can treat the missing edges as failed edges to simulate a forwarding pattern of a supergraph. We will see in our case study in Section VIII that removing one link makes a difference for the applicability of our results.

Proof Sketch: The proof idea is depicted in Fig. 3: as any of the neighbors of v_2 could be the only way to reach t, v_2 must route in a cyclic permutation if no incident links fail, analogously for the neighbors of v_2 if they have a degree of two and do not neighbor s, t after failures. Hence, for every permutation chosen for v_2, the failures of the surrounding nodes can be adjusted such that a routing loop occurs. ■

The above proof never removes more than 15 links, hence:

Corollary 3. *Even under the promise that at most 15 links fail and there is an st-path, the complete graph* K_7 *with seven nodes does not allow for a forwarding pattern* $\pi^{s,t}$ *that can guarantee reaching the destination t from source s if* $|F| \leq 15$.

Impossibility can be shown on the $K_{4,4}$ analogously, however, as it is much sparser than the K_7, we need to remove less links. We omit the case distinction details.

Theorem 7. *The complete bipartite graph with eight nodes, four in each part, minus one link, does not allow for perfect resiliency, i.e., it holds that* $A_p(K_{4,4}^{-1}, s, t) = \emptyset$.

Corollary 4. *Even under the promise that at most 11 link fail and that there is an st-path, the complete bipartite graph* $K_{4,4}$ *does not allow for a forwarding pattern* $\pi^{s,t}$ *that can guarantee reaching the destination t from source s if* $|F| \leq 11$.

1) Generalization of Impossibility: Minor Relationships: It was previously shown that if a graph G allows for perfect resiliency with the source, so do all minors of G [23, §4]. Hence and in particular, all graphs containing a $K_{4,4}$ or a K_7 minus one link as a minor do not allow for perfect resilience.

B. Possibility Results

We now provide positive results on when perfect resilience is achievable. Interestingly, as we will see, we can almost perfectly complement above impossibility results, by providing algorithms for graphs characterized by less dense minors.

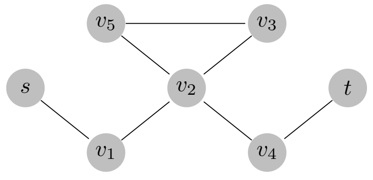

Fig. 3. K_7 (without s-t link) impossibility, when v_2 routes with the permutation v_1, v_3, v_4, v_5, v_1. As v_3, v_5 also route in a cyclic permutation, due to local indistinguishability, packets loops permanently in $v_2 - v_3 - v_5 - v_2$.

Algorithm 1 Perfectly resilient algorithm for K_5 and its minors

```
Input: packet from source $s$ to be delivered to destination $t$,
    local failure set $F_i$, identifiers $u < v < w$
Output: forwarding port decision at node $i$
 1: if $(i,t) \notin F_i$ then
 2:     send to $t$
 3: else if $i = s$ then
 4:     if exactly one neighbor $v$ is reachable then
 5:         send to $v$
 6:     else if exactly two neighbors $u, v$ are reachable, $u < v$ then
 7:         if $inport = \perp$ then send to $u$
 8:         else send to $v$                      ▷ ignore inport
 9:     else if exactly three neighbors $u, v, w$ are reachable, $u < v < w$ then
10:         if $inport = \perp$ then send to $u$
11:         else if $inport = w$ then send to $v$
12:         else then send to $w$          ▷ coming from $u$ or $v$
13: else
14:     if $inport = s$ then send to the neighbor with lowest ID (not $s$) or send back to $s$ if no other choice
15:     else if there is a reachable neighbor $x$ and $inport \neq x$ then send to $x$
16:     else if $s$ is reachable then send to $s$
17:     else send back to $inport$
```

Theorem 8. *For all graphs with at most five nodes Algorithm 1 describes a forwarding pattern matching on the source guaranteeing perfect resilience.*

Proof: We proceed by showing that packets routed with Algorithm 1 reach the destination for all possible distances between source and destination after failures.

By showing it for K_5, we directly show correctness for all minors of K_5 as well due to [23, Corollary 4.2].

If the distance between source and destination is one, Line 2 of the algorithm ensures the packet arrives at its destination directly.

If the distance is two, there are four non-isomorphic candidate graphs on which a packet could visit all other nodes before visiting t, G_1, G_2, G_3, G_4 with $V = \{s, t, x, y, z\}$ and link sets $E_1 = \{(x,y), (y,s), (s,z), (z,t)\}$, $E_2 = E_1 \cup \{(x,s)\}$, $E_3 = \{(x,s), (s,y), (y,z), (z,t), (s,z)\}$ and $E_4 = \{(s,x), (s,y), (s,z), (z,t))\}$, after removing the failed links respectively. Depending on how we order the IDs for x, y, z for E_1, the algorithm may first explore x before returning to s but it will definitely visit z via y and thus find t. For E_2, the algorithm will head straight towards t if z has the lowest identifier. If y is the lowest identifier, the algorithm will visit the nodes in the order s, y, x, s, z, t regardless of the order of the identifiers of x, y. For E_3, the sequence of nodes visited starts with s, x, s if x has the lowest identifier, followed by y, z, t if $y = v$ and $z = w$ or z, t otherwise. If y has the lowest identifier the sequence is s, y, z, t, if $z = y$ it is s, z, t. For E_4 the algorithm guarantees that all neighbors of the source are visited if the previous ones did not connect to the destination as the nodes will send the message back if they can't forward it to t. Note that for subgraphs of G_1, G_2, G_3, G_4 where (s, x) is missing and/or (s, y) is missing from G_4 the destination is reached in at most the same number of steps as well by the same line of arguments, as some detours will not be taken.

If the distance is three, six non-isomorphic candidate graphs exist where a packet could visit all other nodes before visiting t, $G'_1, G'_2, G'_3, G'_4, G'_5, G'_6$ with $V = \{s, t, x, y, z\}$ and link sets $E'_1 = \{(x, s), (s, y), (y, z), (z, t)\}$, $E'_2 = E_1 \cup \{(x, y)\}$, $E'_3 = \{(s, x), (x, y), (y, t), (z, y)\}$, $E'_4 = \{(s, x), (x, y), (y, t), (z, x))\}$, $E'_5 = E_4 \cup \{(z, y)\}$, and $E'_6 = E_5 \cup \{(z, t)\}$, after removing failed link respectively. For E'_1 the algorithm will forward packets on its direct path to the destination if $y = u$. Otherwise there might be a detour to x first. For E'_2, if $x = u$ then the sequence of nodes visited is s, x, y, z, t, if $y = u, x = v$ then it is s, y, x, s, x, y, z, t, if $y = u, z = v$ or $z = u, y = v$ then no detour is taken and it the remaining case with $z = u, x = v$ the path used Is s, x, y, z, t. For E'_3, the path taken is s, x, y, t and for E_4 a visit to z might be included but no loop introduced. For E'_5, z is visited if $z < y$ leading to a path of s, x, z, y, t and s, x, y, t otherwise. In the last graph E'_6, visiting z would lead to a shortcut to t and in both cases t is reached. Note that for subgraphs of G'_1 without (s, x) and G'_3, G'_4 without link to z the destination is reached in at most the same number of steps as well by the same line of arguments, as some detours will not be taken.

If the distance is four, the nodes form a chain and the algorithm ensures that all nodes forward the packet until it reaches it destination. ■

We obtain further positive results for complete bipartite graphs:

Theorem 9. *There exists a forwarding pattern matching on the source and guaranteeing perfect resilience for the complete bipartite graph with three nodes in each part, and its minors.*

Proof: We proceed similarly to the K_5 case. Let $V_1 = \{a, b, c\}, V_2 = \{v_1, v_2, v_3\}$ and $E = V_1 \times V_2$.

We first describe a forwarding pattern for the case where the source is not in the same part as the target and demonstrate that a packet forwarded accordingly reaches its destination under all failure sets if the remaining graph is connected. In the following table, we state for each node and inport combination the order in which a node tries to forward a packet to an outport if w.l.o.g. the source is $s = a$ and the destination is $t = v_3$ (recall that we write $\perp$ for the case without inport at the source):

@s	$\perp:\ t, v_1, v_2$	$v_1:\ v_2$	$v_2:\ v_2$
@b	$v_1:\ t, v_2, v_1$	$v_2:\ t, v_1, v_2$	
@c	$v_1:\ t, v_2, v_1$	$v_2:\ t, v_1, v_2$	
@v_1	$s:\ b, c, s$	$b:\ c, s, b$	$c:\ b, s, c$
@v_2	$s:\ b, c$	$b:\ c, b$	$c:\ b, c$

To illustrate how to read this table, let us look at forwarding pattern for a packet arriving at node b coming from node v_2. If the link to node t is up, the packet will be sent to its destination directly. If the link to t is not available, but the link to v_1 is, then the packet will be forwarded to v_1. If the link to t and the link to v_1 are down, then the packet is returned to v_2.

If the degree of the source after failures is three, the source has a link to the destination and the packet will be sent there directly. If the degree of the source after failures is two, s will forward the packet to v_1 first. *Case k*: If v_1 is only connected to the source, the packet will be sent back to s which in turn will forward it to v_2. In this case, for source and destination to be connected then (v_2, b) and (b, t) must be up and the pattern ensures it reaches t. *Case l*: If v_1 has a degree of two post failures and (v_1, x) with $x \in \{b, c\}$ is up, then the packet is sent to x. If x is connected to t we're done, otherwise the forwarding pattern will either (i) send the packet to v_2, if $(x, v_2) \notin F$ from where it will reach t via c if $x = b$ or via b or (ii) the packet will be bouncing back to s and traverse v_2 and the remaining node in V_1 connecting to t. *Case m*: If all links at v_1 are up, the packet will first visit b from where it will either (i) bounce back to v_1 because the degree of b is one or (ii) reach t directly or (iii) be forwarded to v_2 and reach t via c. If back at v_1, the packet will be forwarded to c next and get to t from there. If the source is only connected to v_1 after the failures, then v_1 must have a remaining degree 2 or 3 and the forwarding pattern visit the nodes in V_1 as described in *Case l .(i)* and *Case m.(i)-(iii)*. If the source is only connected to v_2 after the failures, then v_2 must have remaining degree 2 or 3, the arguments from the previous statement hold for this case too.

Thus we have shown that for s and t in different parts the forwarding pattern routes a packet successfully or the failures disconnect the source from the destination.

If the source and the destination are in the same part, the following forwarding pattern allows packets to reach their destination if w.l.o.g. the source is $s = a$ and the destination is $t = c$ the graph remains connected under failures (using the table notation introduced in the proof of Theorem 9).

@ s	$\perp:\ v_1, v_2, v_3$	$v_1:\ v_3, v_2$	$v_2:\ v_3$	$v_3:\ v_2$
@ b	$v_1:\ v_2, v_3, v_1$	$v_2:\ v_3, v_1, v_2$	$v_3:\ v_1, v_2, v_3$	
@ v_1	$s:\ t, b, s$	$b:\ t, s, b$		
@ v_2	$s:\ t, b, s$	$b:\ t, b, s$		
@ v_3	$s:\ t, b, s$	$b:\ t, s, b$		

Let us assume there is a failure set under which a packet will not reach the destination with this forwarding pattern. If the degree of the source after failures is three, the packet will first go to v_1. (v_1, t) must be in the failure set, as the destination would be reached in the next hop otherwise. Thus the remaining degree of v_1 is either one or two. In the first case the packet is sent back to s, where it will be sent to v_3.

If (v_3, b) is up, the packet is sent to b and if possible forwarded to v_2 from where it would reach the destination. Hence, (v_2, b) must be down and the packet is sent back to v_3 from b and then forwarded back to s from where it will be sent to v_2 and reach the destination.

On the other hand, if (v_1, b) is available, then the packet will be sent to b. There are four possibilities for b. If b has no other neighbors the packet will go back to the source via v_1, then visit v_3 and finally reach the destination from v_2. If b is connected to v_2 but not v_3 after failures, the packet will bounced back to b, and then reach the destination via v_1, s, v_3. If b is connected to v_3 but not v_2 after failures, the packet will forwarded to the destination along the sequence $b - v_3 - s - v_2 - t$. If (b, v_2) and (b, v_3) are up, then the packet will reach the destination via v_2 if (v_2, t) is available or go back to b and visit v_3 to achieve the same end result.

If the degree of the source after failures is two, (s, x) and (s, y) for $x, y \in V_2$ are up. If the remaining degree of x is one, then the packet is sent to y via s. From there it will either reach the destination directly or via b and $z \in V_2 \setminus \{x, y\}$ unless the graph is disconnected. If the remaining degree of x is two, then the packet is sent to b and we can distinguish between three cases for b. If b has no other neighbors the packet will go back to the source via x, then visit y. From there it will either reach the destination directly or via b and $z \in V_2 \setminus \{x, y\}$ unless the graph is disconnected. If b is connected to y but not z after failures, the packet will reach the destination from y unless the graph is disconnected. If b is connected to y and z after failures, the packet will be forwarded to the destination as both y and z will visited since the forwarding pattern at b is a cyclic permutation without any locally incident failures, either directly or via a detour to s.

If the remaining degree of x is three, then the packet is sent to t directly and hence there is no failure set that causes a loop in this case.

If the degree of the source after failures is one, the node $x \in V_2$ the packet is sent to first must have remaining degree at two, as the destination could be reached directly from there or the graph would be either disconnected otherwise. Thus the packet will be forwarded to b which may be still connected to one or two nodes in V_2. In the former case the messages is sent to $y \in V_1, x! = y$ which must be connected to t and the pacet will reach its destination. In the latter case, it will bounced back from from the next node visited and since the forwarding pattern at b without any locally incident failures forms a cyclic permutation the last remaining node in V_1 is explored next. Thus the destination is reached in this last remaining case as well and we demonstrated that there is no failure set that doesn't disconnect source and destination leading to a routing loop.

Lastly, the statement extends to all minors of $K_{3,3}$ due to [23, Corollary 4.2]. ■

V. Perfect Resilience without Source

Given our characterization of when perfect resilience is possible in a model where both the source and the destination of a packet can be matched, we now continue charting the landscape of perfect resilience by considering a model where forwarding rules can only depend on the destination. We are able to provide an almost perfect characterization with respect to complete and complete bipartite graphs.

A. Impossibility Results

Foerster et al. [23] showed that K_5 and $K_{3,3}$ do not allow for perfect resilience in the destination-based model, *i.e.*, $A_p(K_5, t) = \emptyset$ and $A_p(K_{3,3}, t) = \emptyset$. Their proof construction for K_5 starts at some node $v \neq t$, where the link (v, t) is removed, leaving all other links incident to v intact. In their construction, all nodes, except the one node connected to t, must route in a cyclic permutation, a fact retained even if the link (v, t) never existed. Hence:

Theorem 10. *A complete graph with five nodes, minus one link, K_5^{-1}, does not allow for a perfectly resilient forwarding pattern,* i.e., $A_p(K_5^{-1}, t) = \emptyset$.

For $K_{3,3}$, Foerster et al. [23] start their construction on a node v which is in the same part as the destination, and hence there was no link (v, t) to begin with. However, we can observe that in their construction, the permanent loop also traverses nodes of the other part (without t) and the routing behavior remains unchanged if we remove one link incident to t (cyclic permutations are enforced for all non-neighbors of the destination).

Theorem 11. *A complete bipartite graph with three nodes in each part, minus one link, $K_{3,3}^{-1}$, does not allow for a perfectly resilient forwarding pattern,* i.e., $A_p(K_{3,3}^{-1}, t) = \emptyset$.

Whereas K_5 and $K_{3,3}$ are not planar, both K_5^{-1} and $K_{3,3}^{-1}$ are planar [38]. We note that K_5^{-1} is a minor of the planar 7-node construction to show impossibility in [23, Theorem 5.3] and hence Theorem 10 improves their planar result with a smaller number of link and nodes.

1) Generalization of Impossibility: Minor Relationships: It was previously shown that if a graph G allows for perfect resiliency in destination-based routing, so do all minors of G [23, §4]. Hence all graphs containing a $K_{3,3}$ or a K_5 minus one link as a minor do not allow for perfect resilience.

B. Possibility Results

1) One Link Less Gives Perfect Resilience: We next show that the results from §V-A are tight in the sense that removing one additional link from these graphs allows for perfect resilience. We will need the following result:

Corollary 5 (Corollary 6.2 [23]). *Let $G' = (V \setminus \{t\}, E)$ be outerplanar. Then $G = (V, E)$ allows for perfectly resilient forwarding patterns π^t without the source.*

We start with K_5^{-2} in Theorem 12 and $K_{3,3}^{-2}$ in Theorem 13.

Theorem 12. *A complete graph with five nodes, minus two links, K_5^{-2}, allows for a perfectly resilient forwarding pattern π^t, as well as for all minors of K_5^{-2}.*

Proof: Let the nodes of K_5^{-2} be $v_1, v_2, v_3, v_4, v_5 = t$. We proceed by case distinction. If t has one or zero links removed, then the remaining 4-node graph is a proper subgraph of K_4 and hence is outerplanar, *i.e.*, Corollary 5 yields perfect resilience.

If t has two links removed, then let w.l.o.g. v_1, v_2 be the neighbors of t. Note that the graph without t is a K_4 and hence is not outerplanar. In order to obtain perfect resilience, we need to visit, from the starting node v, all of v_1, v_2 being in the same component as v, which we can obtain by using the

@ v_1	$\perp$: v_2, v_3, v_4	$v_3 : v_2, v_4, v_3$	$v_4 : v_2, v_3, v_4$	(v_2: when we visit both we are done)
@ v_2	$\perp$: v_1, v_3, v_4	$v_3 : v_1, v_4, v_3$	$v_4 : v_1, v_3, v_4$	(v_1: when we visit both we are done)
@ v_3	$\perp$: v_2, v_1, v_4	$v_1 : v_2, v_4, v_1$	$v_2 : v_1, v_4, v_2$	$v_4 : v_1, v_2, v_4$
@ v_4	$\perp$: v_1, v_2, v_4	$v_1 : v_2, v_3, v_1$	$v_2 : v_1, v_3, v_2$	$v_3 : v_2, v_1, v_3$

Fig. 4. Routing table to visit both neighbors of t in Fig. 5 under perfect resilience.

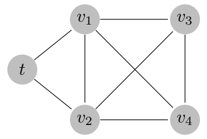

Fig. 5. K_5 with 2 edges incident to t removed.

following forwarding pattern, where we state in the table in Fig. 4 for each inport in which order outports are considered (using the table notation introduced in the proof of Theorem 9):

The correctness of our algorithm follows by careful case distinction, showing that v_1 or v_2 will be visited. Lastly, the proof extends to all minors of K_5^{-2} due to [23, Thm 4.3]. ∎

Theorem 13. *A complete bipartite graph with three nodes in each part, minus two links, $K_{3,3}^{-2}$, allows for a perfectly resilient forwarding pattern π^t, as well as for all minors of $K_{3,3}^{-2}$.*

Proof: Denote nodes of the first part as $v_1, v_2, v_3 = t$ and second part as v_4, v_5, v_6. If $v_3 = t$ has zero or one link removed, then the remaining 5-node graph is a proper subgraph of $K_{2,3}$ and hence is outerplanar, *i.e.*, we obtain perfect resilience with Corollary 5. If t has two link removed, then there is only one node connected to t, w.l.o.g. v_6, and the graph without t, v_6 is a $K_{2,2}$, which is outerplanar. We hence obtain perfect resilience by first routing to v_6 with Corollary 5 and then to t. The proof extends to all minors of $K_{3,3}^{-2}$ [23, Theorem 4.3]. ∎

VI. Resilience with Few Failures

We now study failover routing given a promise that only a small fraction of link are removed. It should be noted that in general we can use any constant-sized graph that does not have a perfectly resilient forwarding pattern, and pad it with extra unhelpful link (and nodes) such that the fraction of link that fail can be made arbitrarily small. Therefore the general case is uninteresting, and we must consider specific graph classes.

We consider routing with source and destination information on complete graphs and complete bipartite graphs. In this setting, there are no perfectly resilient forwarding patterns for K_7 and $K_{4,4}$ (§IV-A). We use a simulation argument to extend these results to graphs of any size: complete graphs and complete bipartite graphs do not have forwarding patterns even if only $O(\sqrt{|E|})$ links fail. In the context of routing without source information slightly better constants can be achieved using the constructions of Foerster et al. [23].

Theorem 14. *For every forwarding pattern on the complete graph K_n on $n \geq 8$ nodes, there is a set of link failures of size at most $6n - 33$ such that the forwarding pattern fails.*

Proof: Assume that the claim does not hold for some n. Then there must exist a forwarding pattern $\pi^{s,t}$ that succeeds even if any $r(n)$ links fail. We simulate $\pi^{s,t}$ on the complete graph K_7 and reach a contradiction with the impossibility of perfectly resilient routing on K_7 (§IV-A). Given K_7, construct a virtual K_n by adding $n - 7$ virtual nodes and virtual links between all pairs of nodes. We construct a failure pattern for K_n that contains the real failure pattern on K_7 as a subset, and simulate $\pi^{s,t}$. The failure set F on K_n is defined as follows.

1) Fail all links between the non-destination nodes of K_7 and the virtual nodes ($6(n - 8)$ links in total).
2) Fail all links that fail in K_7 (≤ 15 links by Corollary 3).

We do not need to fail any additional links incident to the destination. Each node v in K_7 can use the forwarding function $\pi_v^{s,t}$ as if it were on K_n. Since only the destination is connected to the virtual nodes, the packet will never leave the real network K_7. Since we assumed $\pi^{s,t}$ forwards correctly on K_n, it must also forward correctly on this particular failure set. Therefore it forwards correctly on K_7, a contradiction. In total, the number of links in F is $6(n - 8) + 15 = 6n - 33$. ∎

The result is asymptotically the best possible. For example Foerster et al. showed that forwarding with source and destination is always possible if the distance between s and t in $G \setminus F$ is at most 2 [23, Theorem 6.1]. This holds on the complete graph when $|F| \leq n - 2$.

We can give a similar construction for complete bipartite graphs. The proof is a simulation argument based on the impossibility of the $K_{4,4}$ (§IV-A):

Theorem 15. *For every forwarding pattern on the complete bipartite graph $K_{a,b}$, where $a \geq b \geq 4$, there is a set of link failures of size $3a + 4b - 21$ s.t. the forwarding pattern fails.*

Proof: Consider an instance of $K_{4,4}$ such that the node that has the packet initially is on the same side as the destination. Create the virtual graph $K_{a,b}$ by adding $a + b - 8$ nodes and the corresponding links. If $a > b$, we let the larger part be on the opposite side of the source and the destination.

The failure set F is the union of the following sets.

1) The real failure set of $K_{4,4}$ (in total 11 links by Cor. 4).
2) All links from the non-destination nodes of $K_{4,4}$ to the virtual nodes (in total $3(a - 4) + 4(b - 4)$ links).

Again we can simulate the forwarding pattern $\pi^{s,t}$ for $K_{a,b}$ in $K_{4,4}$ since the packet will never enter the virtual part of the graph. Assuming the packet is forwarded correctly on $K_{a,b}$, it is forwarded to the destination on the subgraph that corresponds to $K_{4,4}$, a contradiction (§IV-A). The total size of the failure set F is at most $3a + 4b - 21$. ∎

Chiesa et al. [4, §B.2, B.3] showed how to survive $k-1$ link failures in k-connected complete and complete bipartite graphs. This implies that our result is asymptotically best possible on balanced complete bipartite graphs.

We further briefly investigate the transfer of resilience under few failures to subgraphs and minors. More precisely, if a graph G allows for a k-resilient forwarding pattern, do the subgraphs (respectively, minors) of G then also allow for a

k-resilient forwarding pattern? This property does not hold in general. We know that, *e.g.*, the K_{100} is 99-connected and thus 98-resilient [4, §B.2]. On the other hand, we know that K_7 is not perfectly resilient (§IV-A), and K_7 is a subgraph of the K_{100}. However, $|E(K_7)| < 98$, and hence 98-resilience is equivalent to perfect resilience on K_7, *i.e.*, the 98-resilience of K_{100} does not carry over to its subgraphs. An analogous statement can be made, *e.g.*, with $K_{100,100}$, applying [4, §B.3] and impossibility of perfect resilience on $K_{4,4}$ (§IV-A), As subgraphs are also minors, we hence answer the question in the negative for both complete and complete bipartite graphs.

VII. From Routing to Touring: Perfect Resilience Without Source and Destination

While we have so far focused on the standard routing problem, namely delivering a packet from the source to the destination, in this section, we extend our investigations to a fundamental touring problem: Can local rerouting rules be defined which ensure that a packet will visit *all* nodes in a graph, even under failures? At first this problem seems quite different to normal routing, but touring and routing are deeply connected on complete graphs: in order to reach the destination t, we need to tour all of its neighbors, as an adversary could disconnect t from all neighbors except one.

Our results match the above intuition: as we will show in this section, the borders of (im)possibility move by exactly one node between touring and destination-based routing on complete graphs. What's more, we will provide a complete classification of touring under perfect resilience in Corollary 6.

Beyond the above theoretical motivation, touring can also help in a practical context, by saving expensive routing table space: we deploy the same routing rules, no matter which source or destination a packet has. First, for destination-based routing, the packet will eventually reach the destination, and can then be removed from the network. Second, if we also have the source, we can use touring to implement a broadcast or flooding protocol. Once the source gets the packet again, it checks if the next outport is the same outport as for $\perp$: if yes, the packet has toured the whole network (assuming resilience), and if not, it is still underway in its tour.

A. Complete Touring Characterization in Perfect Resilience

We will now present a complete characterization of touring. Let us first introduce some terminology. We will denote a forwarding pattern $\pi^{\forall}$ as *k-resilient* if for all G and all F where $|F| \leq k$ the forwarding pattern $\pi^{\forall}$ routes the packet from all $v \in V$ to all nodes v' in the connected component of v in $G \setminus F$ and then back to v. We call a forwarding pattern $\pi^{\forall}$ *perfectly resilient* if it is ∞-resilient: all nodes are visited in the tour through the connected component. Let $A_p(G, \forall)$ be the set of such perfectly resilient touring patterns (algorithms).

We can now state our main technical result of this section, yielding a complete classification in Corollary 6.

Theorem 16. *If G is not outerplanar, then it does not support a perfectly resilient touring pattern $\pi^{\forall}$.*

It follows from the arguments of Foerster et al. [23, §6.2] that every outerplanar graph can be toured, by providing a planar embedding and routing according to the right-hand rule, starting on the outer face. In combination with Theorem 16 we hence obtain a complete classification of the possibility of touring all nodes:

Corollary 6. *A graph G allows for a perfectly resilient touring pattern if and only if G is outerplanar.*

It remains to prove Theorem 16. To this end, we first state the auxiliary Lemma 1 which we use to show that K_4 and $K_{2,3}$ do not allow for a perfectly resilient forwarding pattern, its correctness follows analogously as for [23, Lemma 3.1]:

Lemma 1. *Let $G = (V, E)$ with $|E| > 0$ and any $A \in A_p(G, \forall)$. For all F holds: every node routes under A according to a cyclic permutation of all its neighbors.*

As K_4 and $K_{2,3}$ are the forbidden minors of outerplanar graphs, we can then show that no non-outerplanar graph can be toured under perfect resilience. We next study the forbidden minors of outerplanar graphs, as first described by Chartrand and Harary [39, Thm. 1], which we restate as Lemma 2:

Lemma 2. *A graph G is outerplanar if and only if it contains no K_4 or $K_{2,3}$ as a minor.*

The arguments for the next lemmas follow analogously as for Theorems 10 and 11, leveraging the fact that in order to reach the destination therein, all other nodes need to be visited.

Lemma 3. *The complete graph K_4 with four nodes does not support a perfectly resilient touring pattern $\pi^{\forall}$.*

Lemma 4. *The complete bipartite graph $K_{2,3}$ with five nodes, two in one part and three in the other, does not support a perfectly resilient touring pattern $\pi^{\forall}$.*

Foerster et al. [23, §4] showed that perfect resilience for the destination-based model on a graph G transfers to minors G' of G. Their technique relies on taking a perfectly resilient forwarding pattern and showing that for the two fundamental operations in the minor relationship, namely 1) contracting two neighboring nodes and 2)subsetting (taking a subgraph), that the pattern can be naturally adapted to stay perfectly resilient on the obtained minor. Note that a forwarding pattern can also be understood as a port mapping, where packets are forwarded from an inport to an outport, independent of source or destination, and hence the following holds:

Corollary 7. *Given two graphs G and G' such that G' is a minor of G, it holds that $A_p(G, \forall) \neq \emptyset$ implies that $A_p(G', \forall) \neq \emptyset$: if G permits a perfectly resilient touring pattern, so do its minors.*

Combining Lemma 2, Lemma 3, Lemma 4, and Corollary 7, we obtain the desired proof of Theorem 16:

Proof of Theorem 16.: Lemma 2 states that outerplanar graphs are characterized by not having a K_4 or $K_{2,3}$ as a minor. Moreover, K_4 and $K_{2,3}$ do not allow perfectly resilient touring schemes according to Lemmas 3 and 4. As perfect touring resiliency transfers to graph minors, due to Corollary 7, the theorem statement holds. ■

a) k-Resilient Touring: As touring is limited to outerplanar graphs, only very small complete ($K_{\leq 3}$) and complete bipartite graphs ($K_{2,2}$ and $K_{1,n}$) can be toured perfectly. We thus also investigate touring under k-resilience:

Theorem 17. *Let $k \in \mathbb{N}$ and let $G = (V, E)$ be a $2k$-connected complete or complete bipartite graph. There is a forwarding pattern $\pi^\forall$ s.t. G can be toured under every F with $|F| \leq k-1$.*

Proof: A $2k$-connected complete or complete bipartite graph contains k link-disjoint Hamiltonian cycles, following the results of Walecki [40] and Laskar and Auerbach [41]. We generate routing rules as follows, inspired by Chiesa et al. [4, §B.6]: we enumerate the k Hamiltonian cycles as $H_1, \ldots, H_k$. Starting with H_1, the forwarding pattern routes along H_i until a failure is encountered in the next link of H_i at some node v, upon which we switch to the next H_j, where $j > i$ is chosen to be minimum at v (the current Hamiltonian cycle can be identified based in the incoming port). Hence, after $k-1$ failures, at least one Hamiltonian cycle will be without failures (there are k such cycles), and upon entering this Hamiltonian cycle in our routing, we continuously tour all nodes. ■

VIII. Topology Zoo Case Study

In order to better understand the possibility of perfect resilience on real-world networks, we performed a case study[3] on 260 networks from the Internet Topology Zoo [42]. This data set collects information provided by network operators. The networks in this data set have been 3 and 754 nodes and between 7 and 1790 links. We used `SageMath` 9.3[4] to compute if a network is outerplanar or (non-)planar, respectively `minorminer` 0.2.6[5] if it contains a forbidden minor for the respective routing model: K_5^{-1} or $K_{3,3}^{-1}$ for destination-based routing and K_7^{-1} or $K_{4,4}^{-1}$ for source-destination based routing. In case a forbidden minor was found, or if the graph was not outerplanar for touring, we marked the graph as *impossible* w.r.t. perfect resilience—on the other hand, if the network was outerplanar, we marked it as *possible*. From the remaining graphs, if there exists a forwarding pattern for a subset of destinations[6], we marked them as *sometimes*, with the remaining networks marked as *unknown*.

The results are shown in Fig. 6. Even though `minorminer` relies on a heuristic to solve the computationally hard minor search problem, most instances can be classified quickly.

In general, we see that roughly one third of all topologies allow for perfect resilience. Regarding impossibility, the remaining networks cannot be toured under perfect resilience, whereas for the other two routing models 42.5% and 2.7% are impossible, with 1.1% and 31.8% being unknown, and 23.4% and 32.6% allow forwarding patterns for some destinations, for routing algorithms matching on destination and

[3] We will make our code publicly available together with this paper.
[4] https://www.sagemath.org/ [5] https://github.com/dwavesystems/minorminer
[6] I.e, if the graph is outerplanar after removing the destination.

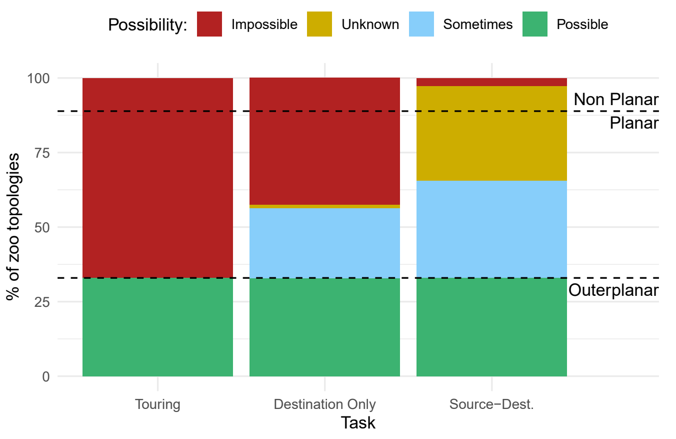

Fig. 6. Perfect resilience classification of Internet Topology Zoo [42] instances.

source-destination, respectively. For the topologies marked as sometimes, on average 21.3% of the destinations are reaching perfect resilience.

Moreover, 55.8% of all topologies are planar but not outerplanar. In this context the seemingly small jump in impossibility classification for destination-based routing, from K_5 or $K_{3,3}$ [23] to K_5^{-1} or $K_{3,3}^{-1}$ in this work, hence allows us to classify 31.3% of the Topology Zoo instances as planar and impossible—previous work cannot show the impossibility of perfect resilience for them.

This implies that our classification measures really lie at the frontiers of (im-)possibility for the Topology Zoo data set and our new results let us classify a lot more real-world topologies.

IX. Conclusion

We presented a detailed characterization of the resiliency which can be achieved using local fast rerouting algorithms that rely on a static forwarding table, in different models. In particular, we have shown that while local routing rules which can also account for the packet source are naturally more resilient, the benefit over rules which only depend on the destination is relatively limited. Similarly, we have shown that exploiting additional topological connectivity, beyond the notion of perfect resilience, is challenging due to the inherently local information the nodes have.

While our work presents a fairly complete landscape of the achievable perfect resilience, there remain several interesting directions for future research. In particular, it would be interesting to chart a similar landscape for the practically relevant scenarios in which links failures are random, or where the routing rules themselves can be subject to randomization.

References

[1] M. Chiesa, G. Kindler, and M. Schapira, "Traffic engineering with equal-cost-multipath: An algorithmic perspective," *IEEE/ACM Transactions on Networking*, vol. 25, no. 2, pp. 779–792, 2016.
[2] J. Feigenbaum, B. Godfrey, A. Panda, M. Schapira, S. Shenker, and A. Singla, "Brief announcement: on the resilience of routing tables," in *PODC*. ACM, 2012, pp. 237–238.

[3] M. Chiesa, A. Kamisinski, J. Rak, G. Retvari, and S. Schmid, "A survey of fast-recovery mechanisms in packet-switched networks," *IEEE Communications Surveys and Tutorials (COMST)*, 2021.

[4] M. Chiesa, A. Gurtov, A. Madry, S. Mitrovic, I. Nikolaevkiy, A. Panda, M. Schapira, and S. Shenker, "Exploring the limits of static failover routing (v4)," *arXiv:1409.0034 [cs.NI]*, 2016.

[5] E. Gafni and D. Bertsekas, "Distributed algorithms for generating loop-free routes in networks with frequently changing topology," *Trans. Commun.*, vol. 29, no. 1, pp. 11–18, 1981.

[6] M. S. Corson and A. Ephremides, "A distributed routing algorithm for mobile wireless networks," *Wireless netw.*, vol. 1, no. 1, pp. 61–81, 1995.

[7] C. Busch, S. Surapaneni, and S. Tirthapura, "Analysis of link reversal routing algorithms for mobile ad hoc networks," in *SPAA*. ACM, 2003, pp. 210–219.

[8] T. Elhourani, A. Gopalan, and S. Ramasubramanian, "IP fast rerouting for multi-link failures," in *INFOCOM*. IEEE, 2014, pp. 2148–2156.

[9] M. Canini, P. Kuznetsov, D. Levin, and S. Schmid, "A distributed and robust SDN control plane for transactional network updates," in *INFOCOM*. IEEE, 2015, pp. 190–198.

[10] M. Borokhovich, L. Schiff, and S. Schmid, "Provable data plane connectivity with local fast failover: introducing openflow graph algorithms," in *HotSDN*. ACM, 2014, pp. 121–126.

[11] O. Reingold, "Undirected connectivity in log-space," *J. ACM*, vol. 55, no. 4, pp. 17:1–17:24, 2008.

[12] K.-T. Foerster and R. Wattenhofer, "Lower and upper competitive bounds for online directed graph exploration," *Theor. Comput. Sci.*, vol. 655, pp. 15–29, 2016.

[13] N. Megow, K. Mehlhorn, and P. Schweitzer, "Online graph exploration: New results on old and new algorithms," *Theor. Comput. Sci.*, vol. 463, pp. 62–72, 2012.

[14] E. Bampas, L. Gasieniec, N. Hanusse, D. Ilcinkas, R. Klasing, and A. Kosowski, "Euler tour lock-in problem in the rotor-router model," in *DISC*, ser. Lecture Notes in Computer Science, vol. 5805. Springer, 2009, pp. 423–435.

[15] D. Dereniowski, A. Kosowski, D. Pajak, and P. Uznanski, "Bounds on the cover time of parallel rotor walks," *J. Comput. Syst. Sci.*, vol. 82, no. 5, pp. 802–816, 2016.

[16] P. Berenbrink, R. Klasing, A. Kosowski, F. Mallmann-Trenn, and P. Uznanski, "Improved analysis of deterministic load-balancing schemes," *ACM Trans. Algorithms*, vol. 15, no. 1, pp. 10:1–10:22, 2019.

[17] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *MobiCom*. ACM, 2000, pp. 243–254.

[18] F. Kuhn, R. Wattenhofer, and A. Zollinger, "An algorithmic approach to geographic routing in ad hoc and sensor networks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 1, pp. 51–62, 2008.

[19] ——, "Worst-case optimal and average-case efficient geometric ad-hoc routing," in *MobiHoc*. ACM, 2003, pp. 267–278.

[20] R. Wattenhofer and A. Zollinger, "XTC: A practical topology control algorithm for ad-hoc networks," in *IPDPS*. IEEE Computer Society, 2004, pp. 216:1–216:8.

[21] A. Zollinger, "Geographic routing," in *Algorithms for Sensor and Ad Hoc Networks*, ser. Lecture Notes in Computer Science, vol. 4621. Springer, 2007, pp. 161–185.

[22] M. Behrend, "How to solve a maze," *Caerdroia Journal*, vol. 36, pp. 10–17, 2006.

[23] K.-T. Foerster, J. Hirvonen, Y.-A. Pignolet, S. Schmid, and G. Tredan, "On the feasibility of perfect resilience with local fast failover," in *Symposium on Algorithmic Principles of Computer Systems (APOCS)*. SIAM, 2021, pp. 55–69.

[24] J. Feigenbaum, B. Godfrey, A. Panda, M. Schapira, S. Shenker, and A. Singla, "On the resilience of routing tables (v2)," *arXiv:1207.3732 [cs.DC]*, 2012.

[25] M. Borokhovich and S. Schmid, "How (not) to shoot in your foot with SDN local fast failover - A load-connectivity tradeoff," in *OPODIS*, ser. Lecture Notes in Computer Science, vol. 8304. Springer, 2013, pp. 68–82.

[26] B. Yang, J. Liu, S. Shenker, J. Li, and K. Zheng, "Keep forwarding: Towards k-link failure resilient routing," in *INFOCOM*. IEEE, 2014, pp. 1617–1625.

[27] B. E. Stephens, A. L. Cox, and S. Rixner, "Scalable multi-failure fast failover via forwarding table compression," in *SOSR*. ACM, 2016, p. 9.

[28] J. Bogle, N. Bhatia, M. Ghobadi, I. Menache, N. Bjørner, A. Valadarsky, and M. Schapira, "TEAVAR: striking the right utilization-availability balance in WAN traffic engineering," in *SIGCOMM*. ACM, 2019, pp. 29–43.

[29] T. Meng, N. R. Schiff, P. B. Godfrey, and M. Schapira, "PCC proteus: Scavenger transport and beyond," in *SIGCOMM*. ACM, 2020, pp. 615–631.

[30] M. Chiesa, A. V. Gurtov, A. Madry, S. Mitrovic, I. Nikolaevskiy, M. Schapira, and S. Shenker, "On the resiliency of randomized routing against multiple edge failures," in *ICALP*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016, pp. 134:1–134:15.

[31] M. Chiesa, I. Nikolaevskiy, S. Mitrovic, A. Panda, A. V. Gurtov, A. Madry, M. Schapira, and S. Shenker, "The quest for resilient (static) forwarding tables," in *INFOCOM*. IEEE, 2016, pp. 1–9.

[32] M. Chiesa, I. Nikolaevskiy, S. Mitrovic, A. V. Gurtov, A. Madry, M. Schapira, and S. Shenker, "On the resiliency of static forwarding tables," *IEEE/ACM Trans. Netw.*, vol. 25, no. 2, pp. 1133–1146, 2017.

[33] J. Edmonds, "Edge-disjoint branchings," *Combinatorial algorithms*, vol. 9, no. 91-96, p. 2, 1973.

[34] K.-T. Foerster, Y. A. Pignolet, S. Schmid, and G. Trédan, "CASA: congestion and stretch aware static fast rerouting," in *INFOCOM*. IEEE, 2019, pp. 469–477.

[35] K.-T. Foerster, A. Kamisinski, Y. A. Pignolet, S. Schmid, and G. Trédan, "Bonsai: Efficient fast failover routing using small arborescences," in *DSN*. IEEE, 2019, pp. 276–288.

[36] ——, "Improved fast rerouting using postprocessing," in *SRDS*. IEEE, 2019, pp. 173–182.

[37] K.-T. Foerster, Y. A. Pignolet, S. Schmid, and G. Trédan, "Local fast failover routing with low stretch," *Computer Communication Review*, vol. 48, no. 1, pp. 35–41, 2018.

[38] K. Wagner, "Ueber eine eigenschaft der ebenen komplexe," *Mathematische Annalen*, vol. 114, pp. 570–590, 1937.

[39] G. Chartrand and F. Harary, "Planar permutation graphs," *Annales de l'I.H.P. Probabilités et statistiques*, vol. 3, no. 4, pp. 433–438, 1967.

[40] B. Alspach, "The wonderful walecki construction," *Bull. Inst. Combin. Appl*, vol. 52, pp. 7–20, 2008.

[41] R. Laskar and B. Auerbach, "On decomposition of r-partite graphs into edge-disjoint hamilton circuits," *Discrete Mathematics*, vol. 14, no. 3, pp. 265–268, 1976.

[42] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, 2011.