Lancet: Better network resilience by designing for pruned failure sets

YIYANG CHANG*, Bytedance, USA
CHUAN JIANG, Purdue University, USA
ASHISH CHANDRA, Purdue University, USA
SANJAY RAO, Purdue University, USA
MOHIT TAWARMALANI, Purdue University, USA

Recently, researchers have started exploring the design of route protection schemes that ensure networks can sustain traffic demand without congestion under failures. Existing approaches focus on ensuring worst-case performance over simultaneous f-failure scenarios is acceptable. Unfortunately, even a single bad scenario may render the schemes unable to protect against *any* f-failure scenario. In this paper, we present Lancet, a system designed to handle most failures when not all can be tackled. Lancet comprises three components: (i) an algorithm to analyze which failure scenarios the network can intrinsically handle which provides a benchmark for any protection routing scheme, and guides the design of new schemes; (ii) an approach to efficiently design a protection schemes for more general failure sets than all f-failure scenarios; and (iii) techniques to determine which of combinatorially many scenarios to design for. Our evaluations with real topologies and validations on an emulation testbed show that Lancet outperforms a worst-case approach by protecting against many more scenarios, and can even match the scenarios that can be handled by optimal network response.

CCS Concepts: • **Networks** → **Traffic engineering algorithms**; **Network performance evaluation**.

Additional Key Words and Phrases: network availability; network optimization; protection routing

ACM Reference Format:
Yiyang Chang, Chuan Jiang, Ashish Chandra, Sanjay Rao, and Mohit Tawarmalani. 2019. Lancet: Better network resilience by designing for pruned failure sets. *Proc. ACM Meas. Anal. Comput. Syst.* 3, 3, Article 49 (December 2019), 26 pages. https://doi.org/10.1145/3366697

1 INTRODUCTION

With the increasing adoption of online and cloud-based services, there is an ever growing requirement on the underlying network infrastructure to ensure that business-critical applications continually operate with acceptable performance [4, 9, 23]. Networks must meet their performance objectives while coping with significant *uncertainty* in their operations. The global scale and rapid evolution of networks imply that failure is the norm in both cloud provider [19, 20, 23, 33] and

*This work was done when Yiyang Chang was at Purdue University.

Authors' addresses: Yiyang Chang, Bytedance, Bellevue, USA; Chuan Jiang, Purdue University, West Lafayette, USA; Ashish Chandra, Purdue University, West Lafayette, USA; Sanjay Rao, Purdue University, West Lafayette, USA; Mohit Tawarmalani, Purdue University, West Lafayette, USA.

2476-1249/2019/12-ART49 $15.00
https://doi.org/10.1145/3366697

ISP [30, 37] settings, and the complexity of failures is on the rise [20]. Failures may involve concurrent, or overlapping events (e.g., a hardware or software failure may occur during a maintenance operation, or during repair of another failure) [20, 29, 35, 39].

Motivated by these challenges, recent works have explored the use of robust approaches where the primary goal is to optimize the *worst-case performance* of a network across a range of possible failure scenarios [14, 29, 36, 39]. Unfortunately, while an important first step, worst-case design may be unduly conservative since a small number of bad failure scenarios may be expensive [3, 5, 11, 22, 25] or even infeasible to design for. In practice, network practitioners seek to compute the frequency of failure scenarios for which performance is unacceptable, and design networks so the percentage of time a network may encounter bad failure scenarios is within acceptable limits. Since existing robust approaches do not directly apply to such design goals, practitioners resort to ad-hoc simulation-based methods today [9].

In this paper, we seek to bridge the gap between existing theory that focuses on worst-case performance, and practical requirements that require that performance is acceptable for scenarios that occur with a desired frequency. For concreteness, we focus on the design of widely used link-based protection routing schemes, where local mechanisms are used to quickly reroute traffic on link failures [39].

A key challenge in designing protection routing schemes is ensuring they are *congestion-free*, i.e., guaranteeing that traffic demand can be sustained when a failure occurs in a manner that does not overload a link, over a wide range of scenarios that may occur owing to concurrent, or overlapping failures. The state-of-the-art approach to achieving these objectives is to design a protection routing that is congestion-free under all possible f or fewer simultaneous link failures [39], where f specifies the desired resilience level. Unfortunately, even a few bad cases may make it infeasible to achieve this goal, forcing a design with much lower resilience. For example, if an architect wishes to protect against 3-failure scenarios, a small number of bad 2- and 3-failure cases imply that the architect can only design for single failures with the existing approach.

In this paper, we present Lancet, a system that can (i) analyze which failure scenarios a network is intrinsically capable of tackling; (ii) enable the design of a protection routing that can protect against *most* scenarios with f or fewer failures (all single and most 2- and 3-failure scenarios in the above example) when infeasible to protect against *all* such scenarios; and (iii) compare which scenarios a heuristically designed protection routing can tackle relative to intrinsic network capability. We make the following contributions:

- We present a linear programming (LP) approach that allows the design of protection routing schemes for general failure sets that involve protecting against most f or fewer failure scenarios with some excluded scenarios. Our approach involves representing general failure sets as a union of m sets. We show that our approach provides the same theoretical guarantees as a more obvious approach that explicitly enumerates scenarios, while being more compact in that the number of constraints depend on m, rather than the number of scenarios.
- We present a novel algorithm that can efficiently classify failure scenarios based on whether the network can intrinsically handle them. To capture intrinsic network capability, Lancet considers a response scheme where the network responds *optimally* to failures, which though not practically realizable, provides a benchmark for comparison. The algorithm handles the multitude of possible scenarios through a divide-and-conquer approach that partitions scenarios into sets, and analytically classifies entire sets as a whole. This allows the algorithm to compactly represent scenarios with acceptable performance as the union of a small number of sets.
- We present a structured approach to determining *which* subset of combinatorially many scenarios to design a protection routing for, and *how* to compactly represent these scenarios as a union of

m sets, with small m to ensure tractable design. To this end, Lancet leverages the classification algorithm above, and selects scenarios based on their performance with optimal network response.

We discuss several extensions with Lancet including tackling richer failure models such as shared-risk link groups [30, 36], design when scenarios that occur with a desired probability must be handled, extensions to tackle multiple traffic demands, and multiple traffic classes.

We evaluate Lancet using multiple real network topologies. Our results show that (i) if design is restricted to the worst-case, there is a huge gap between the set of scenarios that can be handled by protection routing, and optimal network response; (ii) Lancet enables design of protection schemes that significantly outperform existing protection schemes, and surprisingly can exactly match the performance of the optimal response in many cases; (iii) design with Lancet is tractable and takes acceptable time even for large topologies. Further, Lancet's approach of representing general failure sets as a union of m sets is critical to the tractability of the design. While the design phase is offline, we also show how the protection routing parameters can be efficiently adjusted online without running an actual LP. Validations over an SDN testbed demonstrate the benefits of Lancet's approach and the viability of the online adjustments. Overall the results show the promise of Lancet.

Our work may be seen as an early step towards designing networks in a manner that ensures they perform acceptably over scenarios with a desired frequency, a direction important in practice [9], and only starting to receive attention in the research community [12]. While our work focuses on link-based protection schemes [29], the approach is more general, and may be applied to path-based recovery schemes as well including more recent schemes [29]. By analyzing what the network is intrinsically capable of tackling, our approach provides a comparison benchmark for all existing practical routing schemes. A side result of our work is that we show the problem of determining whether the worst-case performance of a network is acceptable when the network responds optimally is NP-complete (Proposition 2), resolving an open question posed by recent work [14]. Finally, from a theoretical perspective, our work shows how robust optimization approaches may be leveraged to tackle design problems which require that a desired percentage of scenarios is tackled, a fact that is likely to have applications beyond networking.

2 GENERALIZED PROTECTION ROUTING MODEL

Two mechanisms are used for quick recovery from network failures today: (i) link-based protection; and (ii) path-based protection [32]. In link-based protection [39], traffic on a link $l = \langle i, j \rangle$ is re-routed upon its failure, along pre-computed detour paths from i to j. To achieve this, an offline procedure is used, which for each link $l = \langle i, j \rangle$, makes a bypass reservation not used until l fails, along paths that are disjoint from l and can carry flow from i to j. When l fails, the network uses this reservation to re-route the traffic on l and executes an efficient online procedure to compute the changes required should another failure occur. In contrast, in a path-based protection scheme [29], failures are handled by diverting traffic to pre-computed backup paths between each source and destination.

In this paper, we focus on link-based protection. The primary advantage of link-based protection is that repair is faster since it happens locally (i.e., at node i if link $\langle i, j \rangle$ fails) - in contrast, with path-based schemes, the failure information must be propagated to the source. However, we note that the ideas behind Lancet are more general, and can be applied to path-based protection schemes as well. In the rest of this section, we discuss a model for link-based protection, which generalizes the state-of-the-art scheme [39].

Protection routing definition. Consider a network with nodes V and edges E for a traffic matrix d with each link e having a capacity c_e. Determining a protection routing , requires computing three sets of variables, r, p, and a, where r denotes routing under normal condition, p denotes

	Symbol	Definition
Parameters	c_e	Capacity of link e
	d_{st}	Traffic from s to t
Variables	a_l	Bypass reservation for link l
	x_e	Failure status of link e
	$r_{st}(e)$	Fraction of s to t traffic on link e (no failure)
	$p_l(e)$	Reservation on link e to handle link l failure

Table 1. Symbol table.

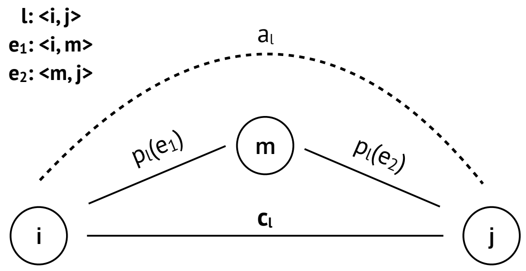

Fig. 1. Illustrating protection routing.

protection routing, and a denotes bypass reservations. (Table 1). r is represented by a set of values $r_{st}(e), s, t \in V, e \in E$, which denotes the fraction of traffic from source s to destination t that traverses link $e = \langle i', j' \rangle$ under normal conditions. We use the notation $r_{st}(e)$ and $r_{st}(i', j')$ interchangeably. For each s, t pair, r represents a flow, and must satisfy the constraints presented below which capture that one unit of traffic exits s and enters t, and traffic is conserved at intermediate nodes. In addition, r should also satisfy capacity constraints, and the full model will be presented in LP (H) below.

$$\sum_{j';\langle i',j'\rangle \in E} r_{st}(i',j') - \sum_{j';\langle j',i'\rangle \in E} r_{st}(j',i') = \begin{cases} 1 & i' = s \\ 0 & i' \neq s, i' \neq t \quad \forall i' \in V \\ -1 & i' = t \end{cases} \tag{1}$$
$$r_{st}(i',j') \geq 0 \quad \forall i', j'; \langle i', j' \rangle \in E$$

The reservation a is represented by a set of values $a_l, l \in E$, which denotes the amount of reservation along bypass paths to protect against the failure of link l. The reservation a_l to protect against the failure of a link l may be less, equal, or larger than c_l but cannot be arbitrarily large since sufficient capacity must be available on the bypass paths. The variable $p_l(e)\ \forall l, e \in E$ denotes the reservation on link e to handle the failure of $l = \langle i, j \rangle$. p_l represents a flow of a_l from i to j, and must satisfy constraints similar to (1) except replacing (i) $r_{st}(i', j')$ with $p_l(i', j')$; (ii) the right-hand-side (RHS) of the flow balance equation with a_l, 0, and $-a_l$; and (iii) (s, t) with $l = \langle i, j \rangle$. Further, the routes protecting against $e's$ failure should not traverse e, i.e., $p_e(e) = 0$.

Fig. 1 illustrates the notation related to bypass reservations. c_l denotes the link capacity of link l. In order to protect against the failure of link l, we reserve the amount a_l to bypass the traffic between i and j. a_l can only be used when l fails. In this example, we use e_1 and e_2 to construct the bypass. To support a flow of a_l, e_1 and e_2 need to carry $p_l(e_1)$ and $p_l(e_2)$ respectively. In this example, each of these variables are equal to a_l because of the flow balance constraints on the routing p_l.

Offline protection routing design. We next present an offline linear program (LP) that computes an optimal protection routing (r, p, a) so as to protect the network against all failure scenarios in a set X. For instance, if the architect wishes to protect against the set of all simultaneous f link failure scenarios, X consists of the set of $\binom{|E|}{f}$ possible f link failure scenarios. The LP minimizes the utilization of the most congested link (or Maximum Link Utilization, henceforth referred to as MLU) across all scenarios $x \in X$. We focus on this metric since it is widely used in the traffic engineering community [14, 39], but discuss a variant in the context of multiple traffic classes later (§5.3). If the optimal MLU U is under 1, then it indicates the protection routing is *congestion free*, i.e., the network can handle any failure scenario $x \in X$. An MLU higher than 1 indicates that the network cannot guarantee a congestion-free routing for all scenarios $x \in X$.

$$
\begin{aligned}
\text{(H)} \quad & \min_{r,p,a,U} \quad U \\
\text{s.t.} \quad & r_{st} \text{ is a unit flow from } s \text{ to } t. \quad \forall s, t \in V \\
& p_l \text{ is a flow of } a_l \text{ from } i \text{ to } j. \quad \forall l \in E, l = \langle i, j \rangle \\
& \forall x \in X, e \in E, \\
& \sum_{s,t} d_{st} r_{st}(e) + \sum_{l \in E} x_l p_l(e) \le U c_e (1 - x_e) + a_e x_e \qquad (2) \\
& a_e \ge 0 \quad \forall e \in E; \quad U \ge 0
\end{aligned}
$$

The LP is show above. The first two constraints capture the definition of a protection routing as elaborated above and summarized in (1). (2) captures the capacity and reservation constraints of all links must be met under all possible failures $x \in X$. The two terms on the left-hand-side (LHS) indicate the total traffic that link e must carry, which includes (i) the traffic under normal conditions ($\sum_{s,t} d_{st} r_{st}(e)$); and (ii) the bypass reservations made on link e to protect against other link failures ($\sum_{l \in E} x_l p_l(e)$). The RHS captures that (i) link e carries at most $U c_e$ traffic when e is operational; and (ii) when e fails ($x_e = 1$), a reservation of at most a_e is available along bypass paths.

A key difficulty in translating (H) into an LP is that the obvious strategy would create $|E|$ constraints per failure scenario, which is not scalable since X may need to be designed for potentially up to $\binom{|E|}{f}$ f-failure scenarios. We discuss how to address this in §3.1.

Our treatment is a generalization of R3 [39], the state-of-the-art protection routing scheme. To protect against the failure of e, R3 reserves a fixed amount c_e that matches the capacity of link e. Instead, our formulation introduces the a_e variables, and thus determines the bypass reservation to be made. There are two advantages to our approach. First, our formulation is valid even when the utilization is higher than 1 while R3 is not. This allows the formulation to be used in settings where we wish to determine how best to augment link capacity to handle failures [14]. Second, when utilization is under 1, our formulation achieves lower MLU than R3. For instance, for the Abilene network [1], and an example traffic matrix [2], the MLU with R3 when protecting against all 2-failure scenarios (each failure impacting 50% of link capacity) is 0.56. In contrast, the MLU with (H) is 0.12 (the optimal achievable if the network could respond ideally). We refer to this generalized approach as Gen-R3.

3 LANCET DESIGN

Existing work [14, 29, 39] only considers design for the worst-case across all f or fewer simultaneous failures. When designing a protection routing for all such scenarios with (H), an MLU that exceeds 1 indicates the infeasibility of achieving the goal. In such cases, an architect must restrict design to scenarios with all $f - 1$ or fewer simultaneous failures. Alternately, an architect may devise heuristic approaches that protect against a subset of f failure scenarios. Lancet is a system that architects may use to answer several questions:

- Which failure scenarios is the network intrinsically capable of handling? Does there exist opportunity for better designs that can protect against *most* if not *all* scenarios involving f failures? How to evaluate a heuristically designed protection routing that seeks to tap into this opportunity?
- When provided with an arbitrary set of failure scenarios to design for, how to efficiently design a protection routing for these scenarios? Using (H) directly for design is not viable since the constraints depend on the number of failure scenarios which is typically large. For a topology with 150 links, there are 11,175 2-failure scenarios, and if most of them (say over 10K) can be handled by the network (as we would expect in practice), LP (H) may have over 1.5 million constraints.
- How to determine which of combinatorially many scenarios to design for in the first place?

To tackle the first question, Lancet captures intrinsic network capability by considering a response scheme where the network responds *optimally* to failures. Since our focus is on minimizing the MLU, the optimal network response involves the network re-routing traffic to minimize the MLU by solving a multi-commodity flow (MCF) problem in response to each failure scenario. We henceforth refer to such a scheme as **Centralized**. While this scheme may not be practically realizable since it takes time to compute and reroute traffic optimally, and to update router configurations, it provides a benchmark for comparison for any protection routing scheme.

To tackle the second question, we show that if the subset of f or fewer failure scenarios to design for is expressed as the union of m sets, (H) can be reformulated into an LP whose number of constraints depend on m, rather than the number of scenarios, a more tractable formulation.

To tackle the third question, we make two observations. First, to ensure design is tractable with (H) reformulated as above, the scenarios to design for must be expressed as a union of m sets with m as small as possible. Merely identifying candidate scenarios to design for is insufficient since directly using them with (H) would result in a large number of constraints. Second, classifying which scenarios can be handled by Centralized can guide protection routing design. This is because the set of scenarios that can be handled by Centralized is a super-set of any set of scenarios for which a congestion-free routing is viable. Leveraging these insights, Lancet includes a novel divide and conquer algorithm to analyze Centralized. The algorithm classifies scenarios that can be handled with Centralized as a compact union of sets, and additionally, is more efficient when analyzing Centralized than enumerating all scenarios.

Roadmap. We show how to efficiently reformulate (H) when designing for arbitrary subsets of f-failure scenarios in §3.1. We next present a divide-and-conquer algorithm to efficiently classify scenarios that can be handled by Centralized in §3.2. Finally, we discuss how Lancet designs protection routing guided by the classification algorithm in §3.3.

3.1 Protection routing design with excluded scenarios.

Consider that a set of scenarios X has been identified, and we seek to design a protection routing for these scenarios using (H). As discussed earlier, directly using (H) is not scalable since it would create $|E|$ constraints per failure scenario, and the number of scenarios can be large. In this section, we discuss how to reformulate (H) into a more tractable LP.

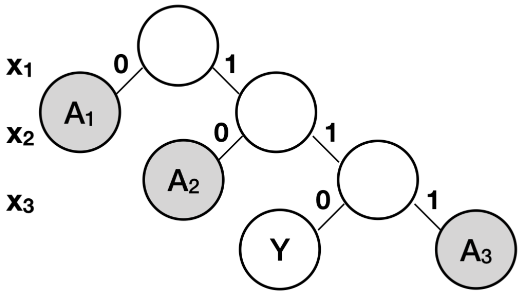

Fig. 2. Illustrating union representation.

Design for all f-failure scenarios. Observe that the capacity constraints of (H) may be equivalently rewritten as:

$$\max_{x \in X} \sum_{l \in E} x_l q_l(e) \leq Uc_e - \sum_{s,t} d_{st} r_{st}(e) \quad \forall e \in E \tag{3}$$

where $q_e(e) = Uc_e - a_e$, and $q_l(e) = p_l(e), l \neq e$.

Consider the special case where X consists of *all* f or fewer link failure scenarios (denoted by X_f). Then, $x \in X_f$ can be expressed using the constraints $\sum_l x_l \leq f, x_l \in \{0, 1\} \, \forall l \in E$. Each of the constraints (3) may be reformulated by (i) relaxing the integrality requirements on x variables, resulting in an LP; and (ii) expressing the LHS as a minimization problem leveraging LP duality. A similar approach was used by [39] when designing for all f failures.

In general, such a reformulated LP is conservative (i.e, it is possible that (H) achieves a lower optimal than the reformulated LP) since the integrality constraints on the x variables have been relaxed. However, we prove below (Proposition 1) that for a specific class of failure sets (of which X_f is a special case), the reformulated LP is exact (i.e., achieves the same objective as the LP (H)). The proof centers around Lemma 1 which shows that for certain failure sets, the corner points are integral allowing us to relax the integrality requirements[1].

Design for general subsets of f-failure scenarios. We next consider design when some f-failure scenarios are to be excluded because they are difficult for a routing mechanism to protect against. Scenarios may be excluded by adding constraints. For instance, to exclude scenarios with $x_1 = 1, x_2 = 1$, and $x_3 = 0$ (links 1 and 2 fail, link 3 does not), the constraint $x_1 + x_2 + (1 - x_3) \leq 2$ is added to the constraints of X_f. Unfortunately, when (3) is reformulated as above, the resulting LP is not exact, since not all corner points correspond to actual failure scenarios. For example, when $|E| = 3$ and $f = 2, x_1 = 1, x_2 = \frac{1}{2}, x_3 = \frac{1}{2}$ is a fractional corner point. To see the effect, note that, although $x_1 + x_2$ does not exceed 1 if x variables are integral, its value at the specified corner point is $\frac{3}{2}$. Hence, the maximum is overestimated, and the resulting reformulation of (H) is overly conservative.

Thus, it is important to find a better representation for X when designing for more general failure sets. We will instead express X as the *union* of multiple sets A_i. Each A_i represents a set containing multiple scenarios. We will later show that even though A_i may contain exponentially many scenarios, it can be captured with constraints that are linear in the number of edges, which leads to a more tractable LP. We then modify LP (H) by replicating (2) m times, once for each A_i. We illustrate using Fig. 2 for the example above. The set X_f (root node) consists of 4 disjoint subsets

[1]In linear programming, a corner point of a set X is a feasible point which cannot be placed in the middle of a line segment consisting of feasible points. The optimal value of a linear program with a bounded feasible region is always at one of the corner points. When corner points are integral, there is an integral optimal solution.

(leaf nodes in the tree). Y corresponds to the scenarios with $x_1 = 1, x_2 = 1$, and $x_3 = 0$ that are to be excluded. Then, we define $X = A_1 \cup A_2 \cup A_3$. We represent A_2 by adding the constraints $x_1 = 1$ and $x_2 = 0$ to the original constraints of X_f. A_1 and A_3 are similarly represented. We replicate (2) three times, once for each A_i. We show in Lemma 1, that for each A_i, the constraints $x_l \in \{0, 1\}$ can now be relaxed to $0 \le x_l \le 1$ because all the corner points of the resulting set are integral. Proposition 1 shows this allows an exact reformulation of (H).

We show that if X is expressed using the union representation, (H) may be reformulated exactly to a more tractable form. Our results apply to not only full link failures but also partial link failures as we discuss later.

Lemma 1. *The corner points of the set:*

$$\begin{aligned} f_1 \le \sum_{l \in E} x_l \le f_2 \\ a_l \le x_l \le b_l \quad \forall l \in E \end{aligned} \tag{4}$$

are integral if f_1, f_2, and $\forall l \in E$, a_l and b_l are integers.

Proof. We assume (4) defines a nonempty set, otherwise there is nothing to show. A matrix is totally unimodular if every square submatrix has determinant 0, 1, or −1 (see Section 4.2 in [15]). It follows easily that the matrix $[1, \ldots, 1]$ is totally unimodular since each square submatrix (the 1×1 submatrix [1]) has determinant 1. Then, it follows from Theorem 4.5 in [15] that (4) defines an integral polyhedron. Since (4) defines a bounded set, its minimal faces are its corner points (see Proposition 3.15 and Theorem 3.33 in [15] and observe that corner points are defined as faces of dimension 0). The result then follows from Theorem 4.1(ii) in [15] which shows that each minimal face contains an integral point. □

Proposition 1. *Let $X = \bigcup_{i \in M} A_i$. For each A_i, let $E_i \subseteq E$ be the set of links whose failure state is fixed, so that, for each failure in A_i, $x_l = x_{il}$ $\forall l \in E_i$. Further, let f_{i1}, f_{i2}, for each $l \notin E_i$, a_{il}, and b_{il} be integers, and let A_i be solutions to:*

$$f_{i1} \le \sum_{l \in E} x_l \le f_{i2} \tag{5}$$

$$a_{il} \le x_l \le b_{il} \ \ \forall\, l \notin E_i \tag{6}$$

$$x_l = x_{il} \ \ \forall\, l \in E_i \tag{7}$$

$$x_l \text{ is integer } \ \forall\, l \notin E_i. \tag{8}$$

Then, (3) can be reformulated exactly as:

$$\begin{aligned} f_{i2}u_{ei2} - f_{i1}u_{ei1} + \sum_{l \notin E_i} b_{il} w_{ei2l} - \sum_{l \notin E_i} a_{il} w_{ei1l} \\ + \sum_{l \in E_i} x_{il}\big(q_l(e) + u_{ei2} - u_{ei1}\big) \le \\ Uc_e - \sum_{s,t} d_{st} r_{st}(e) \quad \forall\, e \in E, i \in M \end{aligned} \tag{9}$$

$$u_{ei2} - u_{ei1} + w_{ei2l} - w_{ei1l} \ge q_l(e) \quad \forall\, e \in E, i \in M, l \notin E_i \tag{10}$$

$$u_{ei2}, u_{ei1}, w_{ei2l}, w_{ei1l} \ge 0 \quad \forall\, e \in E, i \in M, l \notin E_i. \tag{11}$$

Proof. We decompose each constraint in (3) into $|M|$ constraints, one for each A_i. We may do so because for a given t, $\max_{x \in X} \sum_{l \in E} x_l q_l(e) \le t$ if and only if $\max_{x \in A_i} \sum_{l \in E} x_l q_l(e) \le t$ for each $i \in M$.

Constraints (9)-(11) are obtained by taking the dual of the LP that computes max $\sum_{l \in E} x_l q_l(e)$ over the constraints (5)-(7). For each $l \in E_i$, we substitute x_{il} for x_l and we use u_{ei1}, u_{ei2} (resp. w_{ei11} and w_{ei2l}) as the dual variables for the LHS and RHS of constraints (5) (resp. (6)). Since dual feasibility is easy to check from constraints (10) and (11), it follows that taking the dual does not introduce a gap, i.e., the optimal value of the dual LP matches that of primal LP. In particular, (9) is satisfied if the set defined by (5)-(7) is empty. Now, consider the case where (5)-(7) defines a nonempty set. Since this set is bounded, there is a corner point that maximizes $\sum_{l \in E} x_l q_l(e)$ over this set. By Lemma 1, this optimal value matches $\max_{x \in A_i} \sum_l x_l q_l(e)$. This is because the LP relaxes (8), but this constraint is satisfied by at least one of its optimal solutions. Finally, since duality does not introduce a gap, it follows that (3) can be reformulated as (9)-(11). □

Tractability of reformulated LP. The number of constraints in the reformulation (9)-(11) are $|E| \sum_{i \in M} \big(1 + 2\chi_i + 3|E - E_i|\big)$, where we define $\chi_i = 1$ if $E_i \subsetneq E$ and 0 otherwise. Regardless, the number of constraints is $O(|M||E|^2)$. In contrast, the number of constraints in the original formulation (H) is $O(N|E|)$. Here, N is the total number of failure scenarios, which could be as high as $\binom{|E|}{f}$, thus rapidly growing with f. Further, computational evidence shows that the LP running time is sensitive to the number of dense constraints (i.e., constraints with a large number of variables), specifically (2) and (9). The reformulation only has $|M||E|$ such constraints while (H) has $|N||E|$ constraints, and this can help to significantly reduce running time. We will discuss how we express X in the union representation keeping $|M|$ relatively small in §3.2. Finally, in the extreme case where each A_i consists of exactly one failure scenario, $|E - E_i| = \chi_i = 0$, and the reformulation has O($|M||E|$) constraints, matching the size of the explicit formulation (H).

Example. Consider again the example in Fig. 2. Assume a network with 100 links ($|E| = 100$) and at most 3 links can fail simultaneously. Then A_1, A_2 and A_3 contain 156849, 4753 and 1 scenarios respectively. If the original formulation were used, the number of capacity constraints will be $|N||E| = 16160300$. If the new formulation is used, the number of capacity constraints is only $|E| \sum_{i \in M} \big(1 + 2\chi_i + 3|E - E_i|\big) = 59800$, a 270X reduction.

Partial link failures. While our discussion so far has centered around complete failure of a link, our results above also apply to partial link failures. Partial link failures impact part of the capacity of an IP link, and occur because each IP link is usually provisioned as multiple sub-links with different failure modes (e.g., attached to different router line cards). We model partial link failures by considering each link e to consist of n_e sub-links, each of capacity c_e. The first term of the RHS of (2) in LP (H) is changed to $Uc_e(n_e - x_e)$. Similarly, in the reformulation of Proposition 1, the first term of the RHS of (9) is changed to $Uc_e n_e$. The set that represents all f or fewer partial link failures is represented by $\sum_{l \in E} x_l \leq f$, $0 \leq x_l \leq n_l$ $l \in E$, $x_l \in \mathbb{Z}$ $l \in E$. Here, x_l is integral to capture the number of sub-links of l that fail. When designing for X that corresponds to a subset of scenarios involving f partial link failures, Proposition 1 holds provided X can be expressed as a union of sets in a manner required by the proposition, which allows an exact reformulation.

3.2 Analyzing intrinsic network capability

We next discuss how Lancet analyzes intrinsic network capability by classifying the scenarios that the network can handle if it responded optimally using Centralized for any failure scenario. We consider that a failure can be handled if the MLU with Centralized is under a desired threshold U^t. Analyzing Centralized provides a benchmark when evaluating a protection routing scheme, and an upper bound on the set of scenarios that a protection routing can be designed for.

Lancet employs a divide-and-conquer algorithm (Algorithm 1), which classifies an entire set of scenarios as acceptable or violating to the extent possible, and when necessary, partitions the set further. A key highlight of the algorithm is that it generates a compact representation of the large

Algorithm 1: Lancet classification.

```
Function Lancet()
    SSets ⟵ $[F_f, F_{f-1}, \dots, F_1, F_0]$
    pass_set, fail_set ⟵ $\emptyset, \emptyset$
    while SSets do
        $A$ ⟵ SSets.pop()
        Classify($A$, pass_set, fail_set, SSets)
Function Classify(A, pass_set, fail_set, SSets)
    if DoAllCerify(A) then
        pass_set.add($A$)
    else if DoAllViolate(A) then
        fail_set.add($A$)
    else
        $A_1, A_2, \dots, A_n$ ⟵ Partition($A$)
        SSets.extend($[A_1, A_2, \dots, A_n]$)
```

number of scenarios with acceptable performance as the union of m sets, where m is small. Doing so facilitates the design of protection routing for these scenarios because the number of constraints in the reformulated LP (H) depend on m (3.1).

Lancet begins by creating SSets of the form $F_0, F_1, \dots, F_f$, where F_m comprises all failure scenarios with exactly m links failed. At any stage, Lancet picks an SSet whose scenarios have not been classified. For $i < j$, Lancet explores F_i before F_j, because scenarios in F_i are more likely to occur and have acceptable performance. Let A represent a set of scenarios (henceforth referred to as an **SSet**). Lancet uses the oracle procedures *DoAllCertify(A)* and *DoAllViolate(A)* which respectively determine whether *all scenarios* in A meet or violate the performance requirements. If either procedure returns True, the status of A is resolved. If neither returns True (since performance may be acceptable for some scenarios in A, but not others), the algorithm partitions A further into two or more disjoint and complementary subsets using the procedure *Partition(A)*, and the process is repeated on each smaller SSet.

For tractability, when A consists of multiple scenarios, false negatives are permissible with *DoAll-Certify(A)* and *DoAllViolate(A)*. For instance, *DoAllCertify(A)* may return False even if all scenarios certify, but must return True only if the condition is met. Likewise, *DoAllViolate(A)* may return False even when all scenarios violate. However, the algorithm requires that the oracle procedures are exact for SSets containing single scenarios, which guarantees the algorithm converges. We next discuss how Lancet realizes the oracle procedures.

DoAllCertify(A). We show the intractability of checking if all scenarios certify, a previously open problem [14, 39].

Proposition 2. *Consider that the network response for any failure scenario is an optimal multi-commodity flow that minimizes MLU. Then the problem of determining whether the worst-case MLU over all f link failure scenarios is less than U^t (the desired performance threshold) is NP-complete.*

Proof. First, the problem (that we denote as RV) is in NP. Given a particular f link failure scenario, MLU can be checked to not exceed U^t by solving an LP, in polynomial-time, on a reduced graph with corresponding links deleted.

To show that RV is NP-hard, we reduce, to an instance of RV, the Cardinality Maximum Flow Network Interdiction Problem (CMFNIP) [40], where an enemy moves as much of a single commodity as possible from a node s to t in a network where edge $\langle i, j\rangle$ has capacity c_{ij}. An interdictor must decide whether there is a collection of R arcs in the network such that breaking them will ensure that enemy cannot transmit more than M flow. The CMFNIP problem is known to be NP-hard by a reduction from the MAX_CLIQUES problem [40].

Our reduction relies on the simple fact that scaling all the demands in a network increases the utilization of links by the same factor. Given a CMFNIP instance, we construct an instance of RV on the same graph where each edge retains the same capacity, but the demand between s and t is scaled down to 1 while demand between remaining nodes is set to zero. Then, based on the observation above, the CMFNIP instance admits a flow of M if and only if the MLU of the corresponding RV instance, with R failures, is at most $\frac{1}{M}$. Therefore, to solve the CMFNIP instance it suffices to check the latter condition. □

Given the intractability, Lancet obtains a conservative upper bound U^b on the worst-case MLU across all scenarios in A, and verifies if $U^b < U^t$. This guarantees that *DoAllCertify(A)* only returns True when all scenarios have an MLU lower than U^t. Lancet obtains U^b by designing the optimal protection routing for A using the LP (H) and using the resulting MLU. Note that LP (H) models a link bypass mechanism, a more restricted form of rerouting than Centralized. Thus, when (H) indicates that the MLU is at most U^b across all scenarios for the more restricted mechanism, this guarantees that the MLU is at most U^b with Centralized. Finally, if A is a singleton set, Lancet runs an MCF which is exact and ensures the algorithm converges. As an aside, we note that the worst-case MLU may also be bounded using an LP relaxation strategy suggested in [14]. which produces provably tighter bounds than (H). We use (H) since (i) it is computationally faster; and (ii) our experiments show (H) matches the optimal in all practical instances in our context.

DoAllViolate(A). Lancet determines whether all scenarios violate by checking the contrary, i.e., whether there exists $x \in A$ for which performance is acceptable. To check this, Lancet solves an LP that determines whether there exists a feasible multi-commodity flow r from s to t such that the capacity constraints of all non-failed edges are met, and no failed link carries traffic.

$$\begin{aligned} \min_{r,x} \quad & 1 \\ \text{s.t.} \quad & r_{st} \text{ is a unit flow from } s \text{ to } t. \quad \forall s,t \in V \\ & \sum_{s,t} d_{st} r_{st}(e) \le c_e(1-x_e) \quad \forall e \in E \\ & x \in A \quad \forall e \in E \end{aligned} \tag{12}$$

The integrality requirements on x are relaxed, permissible since false negatives are permitted with $DoAllViolate(A)$.

Partition(A). If Lancet is unable to classify all scenarios in A as certifying or violating, it partitions A further. It chooses a link l, and partitions A into two subsets that correspond to (i) scenarios in A where link l has failed; and (ii) remaining scenarios. For the partial link failure model, A may be partitioned into more than two subsets as we illustrate below. The choice of l only impacts the performance of the algorithm and not its correctness. Lancet seeks to pick l such that one of the subsets on partition is found acceptable or violating so that the remaining subset is the only one needing further exploration. To this end, Lancet picks l that is part of the bad scenarios for many highly utilized edges e. Specifically, Lancet computes $\sum_e \pi_l(e)$ for each l, and chooses the largest. Here, $\pi_l(e)$ is the dual multiplier of (10), and is a measure of whether l is present in the worst-case failure scenario for edge e, and how utilized e is in the worst-case.

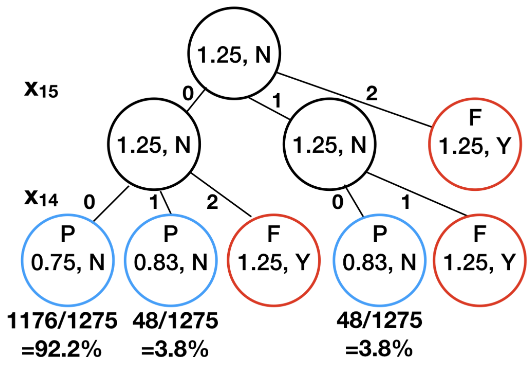

Fig. 3. Lancet's failure classification.

Illustrating classification with Lancet. Fig. 3 illustrates the results obtained by running Lancet's classification algorithm (§3.2) with Centralized for a real topology GEANT, where each link consists of two sub-links (§5). Each tree node represents an SSet, with the root node representing all scenarios involving the simultaneous failure of 2 sub-links.

Each node is annotated with the MLU of the worst-case scenario (a value under 1 indicates all scenarios corresponding to that node are certifiable), and the return value (Yes or No) of *DoAllViolate*(). For example, Lancet is neither able to certify all scenarios corresponding to the root node (MLU = 1.25), nor determine they all violate ("N"). Lancet partitions the corresponding SSet into complementary and disjoint subsets that differ based on the status of link 15. Notice that there are 3 possible states, corresponding to the number of sub-links of link 15 that fail (0, 1, or 2). The algorithm terminates by classifying all scenarios (1,275 in total) into six SSets (leaves of the tree). Blue nodes marked with "P" represent the certified SSets. Red nodes marked with "F" represent the violating SSets. Each certified node is also annotated with the number and percentage of certified scenarios. For instance, the leftmost leaf node corresponds to a single certifiable SSet that captures 92.2% of the scenarios. Compacting certifiable scenarios into a small number of SSets aids design as we discuss next.

3.3 Determining scenarios to design for as a union of sets.

While §3.1 discussed how to efficiently design a protection routing given a set of scenarios expressed in union representation, in this section, we discuss how to determine scenarios to design for and compactly represent them.

To appreciate the complexity of the problem, for a network with E links, there are $\binom{E}{f}$ scenarios involving f link failures, and $2^{\binom{E}{f}}$ possible sets of failure scenarios to consider (e.g., for a network with 300 links, there are 44,850 2-failure scenarios, and 2^{44850} possible sets of 2-failure scenarios). The design of an implementable protection routing involves finding a set among these possible sets (ideally, with cardinality as large as possible) for which (H) can find a congestion-free routing.

Observe that any set of scenarios for which a protection routing design is feasible must exclude *all* scenarios that have unacceptable performance with the optimal Centralized network response. This is because a protection routing can perform no better than Centralized, and including a single bad scenario can make protection routing design infeasible. Ideally, any design strategy must meet three goals: (i) exclude **all** bad scenarios with Centralized; (ii) include as many scenarios as possible; and (iii) compactly represent the scenarios so the LP for designing a protection routing is tractable.

Simultaneously meeting these goals is not easy since by Proposition 2, discovering even a single bad scenario with Centralized is NP-complete, let alone discovering all such scenarios.

Lancet begins by using its classification algorithm (§3.2) to determine the set (say L) of failure scenarios for which Centralized performs acceptably (i.e., achieves an MLU under U^* where U^* is a desired bound). When design is viable for L (which interestingly has always been the case in our evaluations), it is guaranteed to be the set with maximum cardinality for which protection routing is viable. More generally, Lancet classification may be continued to identify all scenarios with Centralized for which MLU is under an updated lower threshold (say $0.9U^*$). The insight is that considering scenarios which are comfortably handled by Centralized could identify a failure set that is amenable to a protection routing. The threshold can be iteratively reduced if needed.

Obtaining a compact union representation. A naive approach to identifying scenarios that can be handled by Centralized is to exhaustively enumerate and run an MCF for each scenario. This approach (or more generally, any heuristic that merely identifies scenarios to design for without compacting them) would result in an LP with $|E|$ constraints per failure scenario, which is intractably large even for small topologies (see §5.4 for empirical results). In contrast, Lancet's classification algorithm (§3.2) obtains scenarios certifiable with Centralized as a compact union of sets. For instance, for the GEANT network (Fig. 3), Lancet represents 1,272 certifiable scenarios with Centralized as a union of $|M| = 3$ sets. Consequently, the number of constraints in the LP for designing a protection routing depends on $|M|$ (3) rather than the number of scenarios (1,272), substantially reducing the number of constraints. Below, we present a result that bounds the number of sets in the union representation with Lancet.

Theorem 3.1. *Assume that there are s violating scenarios within F_f, where F_f are scenarios that have f failures, and that DoAllCertify(A) is exact (i.e., no false negatives). Then, Lancet solves at most $2sE + 1$ nodes while exploring F_f, where E is the number of branching variables, which is the number of links if branching is on link state.*

Proof. Assume a set of 0-1 points (failure scenarios with f failures), C, form the corner points of a set defined by linear constraints and the set V of 0-1 points (violating scenarios) has cardinality s. Consider the divide-and-conquer approach used by Lancet. It is not possible that all children of a node N only contain scenarios outside of V, since in this case, DoAllCertify() would return true at node N, and N would not have been partitioned. Hence, at any level there are at most s nodes that contain scenarios in V, and s siblings that do not contain such scenarios. Including the root node, there are a total of $2sE + 1$ nodes. The argument can also be extended to address partial failures, where each link $\langle i, j\rangle$ may be viewed as consisting of n_{ij} sub-links, any of which could fail leading to partial capacity loss. The only difference is that there may be more siblings that do not contain scenarios in V. In this case, the tree may contain at most $1 + s\sum_{\langle i,j\rangle}(1 + n_{ij})$ nodes. □

Discussion and Implications. Given a few, say s, bad failure scenarios involving at most f failures, an implication of the above argument is that we can express the remaining failure scenarios as a union of no more than $s|E|$ failure sets, each of the form (4). This allows exact reformulation as per Proposition 1, which in turn bounds the number of constraints in the reformulated LP. In practice, we expect s to be small since the network is designed to tackle most f failure scenarios besides a few bad cases. Our empirical results (§5) show significantly fewer failure sets are needed in practice than this theoretical bound since Lancet's partitioning heuristic is effective in isolating a bad failure scenario by fixing a few links rather than all links as the proof conservatively assumes. Further, we have observed empirically that the first few certifiable SSets identified by Lancet typically account for a large number of scenarios. Terminating the algorithm at this stage and using these SSets in the design can already provide significant benefits.

4 GENERALIZATIONS AND EXTENSIONS

We describe several extensions with Lancet below:

Richer failure models. Lancet is easily extended to tackle the fact that multiple links belonging to a Shared Risk Link Group (SRLG) [30, 36] may fail together (e.g., owing to a common optical related failure). The set of link groups G is considered, and each group g_k is associated with a set of links that may fail together. We introduce binary variables x_k^g which indicates whether a particular link group has failed. Each SSet is captured by the constraints:

$$\begin{aligned}
&\sum_{k \in G} x_k^g = f \\
&x_k^g \leq x_{ij} \quad k \in G, \langle i,j \rangle \in g_k \\
&\sum_{k \in G, \langle i,j \rangle \in g_k} x_k^g \geq x_{ij} \\
&x_k^g, x_{ij} \in \{0,1\} \quad k \in G, \langle i,j \rangle \in E
\end{aligned} \tag{13}$$

The constraints capture that (i) at most f SRLGs fail together; (ii) when a group fails, all its associated links fail; and (iii) a link fails only if one of the groups it is part of fails. Lancet's classification and reformulation approach applies as before, except that the above constraints are used to capture the set of scenarios X. Although the reformulation of LP (H) is no longer exact, Lancet is conservative and guarantees that the MLU in practice is no more than the value reported by Lancet.

Design to meet probability requirements. Consider an architect goal of design a protection routing that is guaranteed to be congestion free for scenarios that occur occur $p\%$ of the time, given the probability of different failure scenarios. Lancet is easily adapted to this task. Specifically, the classification algorithm (§3.2) is modified to maintain a weighted count of scenarios for which performance is acceptable or violates requirements, with the weight indicating the probability that the network is in a given failure state. The algorithm terminates when either the weighted count of certifiable scenarios exceeds $p\%$, or that of violating scenarios exceeds $1 - p\%$ (indicating the network is not intrinsically capable of meeting the goal). In the former case, the approach in §3.1 is then used to design a protection routing for the certifiable scenarios.

More formally, let $\mathcal{F}(x)$ denote the minimum MLU under failure scenario x. Let χ be a random failure scenario drawn from X, the set of all failure scenarios, with a given probability distribution. Then, the random variable $\mathcal{F}(\chi)$ measures MLU for randomly drawn scenario χ. We are interested in obtaining $P(C)$, where C denotes the set of certifiable scenarios and corresponds to the event $\mathcal{F}(\chi) \leq U^*$, where U^* is the desired MLU target (typically, 1 or below). The complement of C, the set of violating scenarios, is denoted as B.

Let the current SSet A be partitioned into $\{A_1 \mid \dots \mid A_n\}$ so that $\sum_{i=1}^n P(A_i) = P(A)$. For any event Y, and in particular for C and B, we have $P(Y \cap A) = P(Y \mid A)P(A) = \sum_{i=1}^n P(Y \mid A_i)P(A_i)$. To bound $P(Y \cap A)$, we approximate the right-hand-side of the above equivalence. Consider two index sets $I, J \subseteq \{1, \dots, n\}$ so that, for all $i \in I$, $P(Y \mid A_i) = 1$, and, for $i \in J$, $P(Y \mid A_i) = 0$. Then, $\sum_{i \notin J} P(A_i) \geq P(Y \cap A) = \sum_{i \in I} P(A_i) + \sum_{i \notin I \cup J} P(Y \mid A_i)P(A_i) \geq \sum_{i \in I} P(A_i)$. Applying the bounding argument with $Y = C$, an element $i \in I$ if $P(C \mid A_i) = 1$, which occurs if and only if the performance of all scenarios in A_i is certifiable. Likewise, $i \in J$ if $P(C \mid A_i) = 0$, which occurs if and only if no scenario is certifiable. Then, for any classification tree generated by Lancet, and at any snapshot (even when not all scenarios are classified), the weighted sum of certifiable scenarios $\sum_{i \in I} P(A_i)$ is a lower bound on the probability of certifiable scenarios. Since the above treatment also applies when $Y = B$, it yields lower as well as upper bounds on the probability of certifiable and violating scenarios.

Network	Nodes	Edges
Abilene	11	14
GEANT	32	50
Deltacom	103	151
ION	114	135

Table 2. Topologies.

Topology (# of failures)	# of SSets	# of scenarios
GEANT (2)	3	1,272
ION (2)	5	9,172
Deltacom (2)	6	11,465
Deltacom (3)	3	466,486

Table 3. Effectiveness of Lancet in compacting certifiable scenarios into a small number of sets.

The above approach is general, and works with any oracle that provides the probability of each network failure scenario which in turn allows $P(A)$ to be computed. While Lancet can handle more general correlations, typically correlated failures are captured using SRLGs, with the failure of distinct SRLGs treated as independent events. [19, 30]. If failure probabilities are homogeneous, the probability of each SSet can be computed using a binomial distribution. Since more generally, failure probabilities may vary [18, 19, 30, 33, 37], Lancet categorizes failure events into classes, where probabilities of events in the same class are the same, while those of events belonging to different classes may vary. Lancet then appropriately adapts the SSet probability computations. Lancet can handle more general correlations if provided necessary information. For instance, Lancet can consider correlated failures of SRLGs g_1 and g_2 if the joint failure probability is provided, along with the probability of each link g_1 or g_2 failing by itself. The failure probabilities may themselves be obtained from historical data of time between failures and time to repair, or reliability estimates of links. Our discussions with network operators of several large online service providers suggests such data is available and is already being used in simulation-based testing [4, 9].

Multiple traffic demands. While we have focused on failure uncertainty, we next discuss how Lancet tackles the case, where demand is also uncertain and can take one of h values, say d_i for $i \in \{1, \ldots, h\}$. Here, each d_i is a traffic matrix consisting of traffic between all node-pairs. The uncertainty set X now consists of scenarios specified jointly by the network failure state and traffic demand. The worst-case problem would involve multiple demands and multiple failures, which can be handled by modifying LP (H) in a similar fashion as [39]. Lancet uses this oracle, and branches on failure state and demand.

For continuous demands (such as the convex hull of a set of discrete demands), we partition possible demands into regions, and, associate each region with a demand that dominates all the demands in that region. For instance, if, within a region, the demand for a node-pair (s, t) varies between d^l_{st} and d^h_{st}, with $d^l_{st} < d^h_{st}$, then, in the dominating demand, we assign the (s, t) demand at its highest possible value d^h_{st}. When the resulting discrete approximation is used, Lancet is conservative, which guarantees the MLU in practice is no more than the value reported by Lancet.

5 EVALUATIONS

Our evaluations are motivated by the following questions:

• What is the set of scenarios for which existing protection routing schemes can ensure a congestion-free routing? How does this compare to the scenarios that the network is intrinsically capable of handling in a congestion-free manner?

• How effective is Lancet in designing better protection routing schemes that can handle a wider range of scenarios than the state-of-the-art?

• Is the design time with Lancet acceptable? How important and effective is Lancet's union representation in reducing computation time?

Schemes compared. To answer the questions above, we compare Lancet with the following schemes:

• *Gen-R3.* This refers to the generalized version of R3 [39], the state-of-the-art approach (§2). Here, LP (H) is used to design a protection routing for all f or fewer failures.

• *Centralized.* This refers to an ideal scheme that uses an optimal multi-commodity flow to re-route traffic on failures (§3). This measures whether the network is intrinsically capable of handling a given failure scenario.

Topologies evaluated. Table 2 summarizes the topologies used in our evaluations. We select one moderate-size network, GEANT, and two large networks, Deltacom and ION, from the Internet Topology Zoo [1]. One-degree nodes in the topologies are recursively removed so that the networks are not disconnected with a single link failure. We use the gravity model [43] to generate traffic matrices with MLU in the range [0.6, 0.67] across the topologies. GEANT has link capacities ranging from 1 to 100 Gbps. Link capacities for Deltacom and ION are unavailable, and we use uniform capacities. We model each link with its capacity split evenly across two sub-links that fail independently.

Lancet implementation. We implement Lancet in Python, and use Gurobi 8.0 [24] to solve LPs. We implement the classification algorithm and oracle procedures described in §3.2. Starting with the base case where the design is tested against fixed numbers of failures, we dynamically derive and add constraints to fix failure states of links.

5.1 Does design for f failures suffice?

We use Lancet to analyze the intrinsic ability of a network to handle failures, and how close existing protection routing designs are to this intrinsic capability. We begin by considering the GEANT network. When LP (H) is used to design a protection routing for GEANT for all f or fewer failures, the MLU is under 1 for $f = 1$, but exceeds 1 for $f = 2$ (we term the resulting routings Gen-R3(1) and Gen-R3(2) respectively). This indicates it is feasible to design a protection routing for all single failure scenarios, but not all two failure scenarios. With current state-of-the-art, the architect can only guarantee protection against single failure scenarios alone.

Fig. 4 shows the percentage of 1-, 2-, and 3-failure scenarios for which MLU < 1 with Centralized (optimal network response), and the two Gen-R3 schemes. We obtained these results using Lancet's classification algorithm (3.2). While the algorithm is described only for Centralized, we have extended it to classify scenarios when the network response involves a fixed protection routing. Note that the Gen-R3 schemes may be able to handle some 2- or 3-failures, even though not explicitly designed for them.

The results show that Centralized can handle practically all 2- and 3-link failure scenarios which indicates the network's intrinsic capability to handle such failures. However, while the Gen-R3 schemes can handle all single failures, they can only handle a much smaller subset of 2- and 3-failure scenarios. Gen-R3(2) performs poorly because it optimizes the worst-case MLU under all 2-link failures, and cannot control the performance of the remaining scenarios. Gen-R3(1) performs better, but still incurs a large gap relative to Centralized because it does not explicitly design for any 2-failure scenario. Overall, the results reveal a big gap between the scenarios that the Gen-R3 schemes protect against, and intrinsic network capability.

5.2 Design with excluded scenarios

We now evaluate the potential for bridging the large gap between Gen-R3 and Centralized using Lancet. Recall that Lancet represents scenarios that can be handled by Centralized as a union of m sets, and uses the reformulated LP(H) to design a protection routing (that we term Lancet) for these scenarios (§3.3). Our evaluations focus on two aspects: (i) how effective is Lancet in ensuring

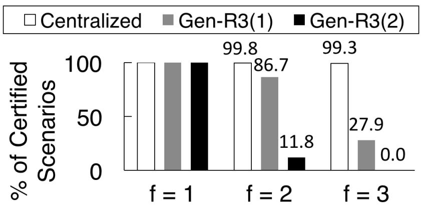

Fig. 4. Worst-case design effectiveness across different number of link failures.

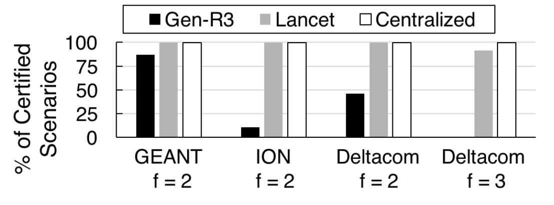

Fig. 5. Effectiveness of Lancet generated designs.

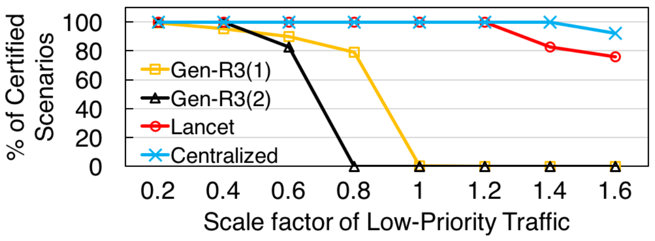

Fig. 6. Efficacy of Lancet aided designs with two traffic classes.

m is small relative to the total number of certifiable scenarios? This is critical to ensuring tractable protection routing design; (ii) how does Lancet perform relative to Gen-R3 and Centralized?

Table 3 shows the number of scenarios that certify with Centralized and the number of sets that Lancet can compact these scenarios into. For GEANT, Lancet compacts 1,272 certifiable 2-failure scenarios into 3 sets. The benefits are more significant for larger case studies. For Deltacom (ION), Lancet compacts 11,465 (9,172) 2-failure scenarios into 6 (5) sets.

Fig. 5 shows the percentage of scenarios that can be handled by Lancet, in comparison to Gen-R3 and Centralized. For Gen-R3, we consider the largest f for which LP (H) is able to find a protection routing, but always use an f of at least 1. For all topologies, and with 2-failure scenarios, Lancet achieves much better performance than Gen-R3. Surprisingly, it matches Centralized, indicating it can protect against *all* the scenarios that the network can intrinsically handle. For example, for ION, the percentage of certified scenarios improves from 11.0% to 99.9%, matching Centralized.

To gain insights with an even larger case study, we explored Deltacom under 3-failure scenarios. We observed that the first three SSets identified by Lancet already succinctly captured 466,486 certifiable scenarios (Table 3). When these SSets were used in the design, Fig. 5 shows the protection routing produced by Lancet certifies 91.5% of scenarios. This is in contrast to Gen-R3 that certified no 3-failure scenarios, and close to the 99.7% of 3-failure scenarios that can be handled by Centralized.

5.3 Design with multiple traffic classes

So far, we have focused on the MLU metric when all traffic must be handled. We next show Lancet extends to a context with multiple traffic classes where the architect wishes to meet all high priority, and as much low priority traffic as possible.

Designing a protection routing for multiple traffic classes involves minor adaptations to LP (H) that we detail in the Appendix. The two-class LP determines a protection routing that handles all high-priority traffic and the low-priority traffic scaled by a factor Z, while maximizing Z. We refer to Z as the scale factor, with $Z \geq 1$ indicating the entire low priority traffic is handled. We refer to Gen-R3(f) as a protection routing derived from this two-class LP when protecting against all simultaneous failures involving f or fewer links. Similar to earlier, Lancet is obtained by (i) using Lancet to classify scenarios that obtain a $Z \geq 1$ with Centralized; and (ii) using the two-class LP to design with the set so obtained.

Fig. 6 shows the fraction of scenarios that achieve different Z thresholds for the schemes above for GEANT's 2-failure scenarios. We split the original traffic matrix into two classes with high and low priority. For each cell we assign a random fraction of it to the high-priority traffic class, and the rest to the low-priority one. Each curve corresponds to one scheme, and shows the fraction of 2-failure scenarios that can attain a particular Z. The top-most curve shows the ideal Centralized scheme, which attains Z of 1 for over 99% of the scenarios. Lancet performs nearly as well as Centralized. While it degrades moderately for the most stringent performance thresholds ($Z = 1.4$ and 1.6), we note that an architect could use Lancet to generate new protection routings optimized for these thresholds if this is desirable. The two Gen-R3 schemes perform poorly, with no scenario achieving a Z of 1 where all low priority traffic could be carried. Note that Gen-R3(2) matches Centralized and performs slightly better than Lancet for the worst-case (achieving a Z of 0.42) but this comes at the expense of performance for the vast majority of scenarios.

5.4 Design time with Lancet

We present the time to design a protection routing with Lancet using reformulated LP (H) given scenarios expressed in the compact union representation. We compare Lancet with an approach that we refer to as *Enumeration*. This approach uses LP (H) for protection routing design but explicitly enumerates scenarios. We report the time on a 3.00GHz Intel Xeon CPU using a single-threaded implementation.

The left-most bar of Fig. 7 shows the design time with LP (H) for *Enumeration* for the moderate-sized GEANT network. The design was prohibitively expensive, and did not terminate even after 18 hours. All but the left-most bar of Fig. 7 present design time with Lancet. For the same GEANT topology, Lancet takes only 10 seconds. The key reason is Lancet's ability to compact the $n = 1,272$ certifiable 2-failure scenarios into just $m = 3$ sets (Table 3). The number of constraints (and dense constraints) grow with n with *Enumeration*, but grow with m for Lancet.

The remaining bars of Fig. 7 present the design time with Lancet for the larger sized Deltacom, and Ion topologies. We do not report *Enumeration* given it is prohibitively expensive even for GEANT. Lancet continues to perform well with larger case studies taking 46 minutes for Deltacom, and a little over an hour for Ion. To further stress scaling, the right most bar presents the performance of Lancet with Deltacom when designing for certifiable scenarios with 3 or fewer failures (over 466K scenarios). Lancet takes less than 2 hours because it can compactly represent these scenarios using 3 sets.

We note that the design times of a few hours for larger topologies is acceptable since the protection routing is designed offline. Online updates on failure are extremely fast as they involve simple recomputations, and no LP need be solved (as we will discuss further in §6). Further, there are several performance optimizations possible with Lancet in the future that can further reduce design time. Overall the results show Lancet's compact union representation of failure sets is critical and effective for protection routing design.

6 VALIDATIONS ON SDN TESTBED

In this section, we validate the effectiveness of Lancet in protecting against a wider range of failure states than Gen-R3 using SDN testbed experiments.

Emulation setup and protection routing implementation. We conduct emulations using Mininet 2.2 and OpenVSwitch 2.10. Both normal (r) and protection routing (p) variables require each

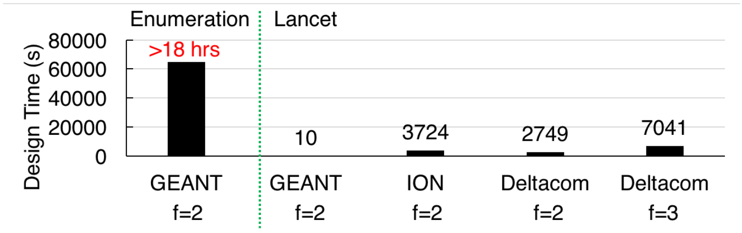

Fig. 7. Design time for the offline phase with Lancet and comparisons to Enumeration. The bar to the left of the green dotted line shows that Enumeration performs poorly even for the moderate-sized GEANT, and it is not considered further for larger topologies. The bars to the right show Lancet scales well with topology size and the number of scenarios.

router to forward traffic to a destination with multiple next hops. We achieve this leveraging the select group table feature in OpenFlow 1.5 using multiple buckets with different weights in a group table entry. OpenVSwitch's internal hashing method guarantees that packets belong to a single flow map to the same forwarding path. Initial flow rules for normal and protection routing are installed by a central controller. When a link $\langle i,j \rangle$ fails, the head (i) and tail (j) switches are responsible for pushing and popping a unique MPLS label (say l_{ij}) encoding the failure. All the switches forward the labeled packets based on flow rules pre-installed by the controller that translate the appropriate protection routing variables p. Packets may carry stacked labels if they encounter multiple failed links along the path.

Efficient online adjustment of protection routing parameters. Consider a protection routing (r, p, a) initially computed offline using LP (H) (Table 1). If link $\epsilon = \langle i,j \rangle$ fails, the parameters for the routing must be recomputed since ϵ is unavailable for reservations to protect against future failures. To achieve this, failure information of $\langle i,j \rangle$ is propagated to the controller which efficiently adjusts the parameters as we discuss next. Since this update is only to protect against future failures, it is acceptable to involve the controller.

One approach to achieving this is for the controller to solve the design LP again to compute (r', p', a') but for the network where link ϵ does not exist. However, this can take time and is not suitable for online operation. Instead, we have been able to show that (r', p', a') can be efficiently computed online by making the following quickly computable adjustments to (r, p, a). Specifically, we have:

$$r'_{st}(e) = \begin{cases} r_{st}(e) + r_{st}(\epsilon)(p_\epsilon(e)/a_\epsilon) & \forall s,t,e \in E \backslash \{\epsilon\} \\ 0 & \forall s,t,e = \epsilon \end{cases}$$

$$\tilde{p}_l(e) = \begin{cases} p_l(e) + p_l(\epsilon)(p_\epsilon(e)/a_\epsilon) & \forall l \in E \backslash \{\epsilon\}, e \in E \backslash \{\epsilon\} \\ 0 & \forall l \in E \backslash \{\epsilon\}, e = \epsilon \end{cases}$$

$$a'_e = a_e - \tilde{p}_e(e) \quad \forall e \in E \backslash \{\epsilon\}$$

$$p'_l(e) = \begin{cases} \tilde{p}_l(e) & \forall l \in E \backslash \{\epsilon\}, e \in E, l \neq e \\ 0 & \forall l \in E \backslash \{\epsilon\}, e \in E, l = e \end{cases}$$

These adjustments capture the increase in (s,t) traffic on link e when ϵ fails, for each (s,t) pair, and the increase in reservation on link e to protect against a subsequent failure of l, because reservations on ϵ have been invalidated. The final few steps ensure link e does not use itself when protecting against the failure of e. The update to r' is locally computed by each router since router i starts labeling packets to be sent along ϵ and forwards them using the protection routing p as discussed earlier. Thus, the controller only pushes p' and a' to switches. We next discuss the complexity of the update process. During the online adjustment, a node i will locally update r for each of its adjacent links, and for all source-destination pairs. As shown above, this takes $d_i|V|^2$ operations where d_i is the degree of node i and V is the set of nodes. However, while we do not elaborate, it is possible to optimize this further by implementing r in a manner that only tracks the destination rather than both the source and destination, which would result in only $d_i|V|$ operations. Further, p is updated in a centralized manner and the process takes $O(|E|^2)$ operations.

Proposition 3. *Let (r, p, a, U) be a feasible solution to LP (H) for a graph $G = (V, E)$ when protecting against all failure scenarios defined in the set X. Then, we can show that (r', p', a', U) is a feasible solution to (H) for the graph $G' = (V, E - \epsilon)$ and for failure scenarios in X with ϵ failing.*

The proof proceeds by showing that (r', p', a', U) satisfies every constraint of LP (H) for the graph $G' = (V, E - \epsilon)$ and for failure scenarios in X with ϵ failing. We defer a detailed proof to the Appendix.

Experimental results. We compare the performance of Lancet and Gen-R3 by emulating Abilene (with each link failing fully) on the testbed for a matrix where the traffic is dominated by one source destination pair. It is infeasible to design a protection routing with Gen-R3 to protect against all 2-failure scenarios, since the graph may get disconnected on a few of these failure scenarios (and LP (H) is infeasible to solve). Hence, we design Gen-R3 for all single failures. For Lancet, we design the protection routing for the union of the SSets corresponding to 0-failure, 1-failure and all 2-failure SSets which can achieve an MLU under 1 with Centralized (20 SSets representing 93% of failure scenarios in all) as described in §3. The MLU with LP (H) for the union of sets is 0.9, indicating all these scenarios can be tackled but the network may operate close to saturation on some of them.

We create 30 UDP flows between the source and the destination, and split the demand uniformly across the flows. Note that any flow is hashed to one path constantly, but multiple flows may be hashed to different paths following the routing parameters. The rate is set to the maximum possible given the system resources of the host machine, and link capacities are proportionally scaled down. Fig. 8 presents the throughput and packet loss rate measured at the destination on the failure of two links e_1 and e_2 (highlighted by the circles), occurring 30 seconds apart. Fig. 8 (top) shows the result for Lancet. Notice that each failure event leads to a transient impact on throughput and loss rate, though performance quickly recovers after protection routing is activated. Fig. 8 (bottom) shows the result for Gen-R3. While it can handle the first failure, there is 100% packet loss after the second one, because Gen-R3 is restricted to design for single failure scenarios. In this example, Gen-R3 resulted in e_1 and e_2 mutually using each other to protect against their respective failures, which is sufficient to guard against a single but not two failures. Lancet prevents this by using more diverse paths to protect e_1 and e_2. Like [39], Lancet may suffer transient loops when links fail near simultaneously. However, the final protection variables are correctly restored based on the sequence of failures observed by the controller using the above adjustments. We have also proven that the adjustments on failures above can be applied in any order when multiple failures are involved, and would produce the same results.

7 RELATED WORK

Mechanisms that guarantee the network does not incur congestion when responding to failures have been developed recently in the context of both link-based protection [39], and path-based protection [29]. However, these schemes only consider worst-case performance over f failures. While we have extensively discussed link-based protection [39], FFC [29] performs path-based

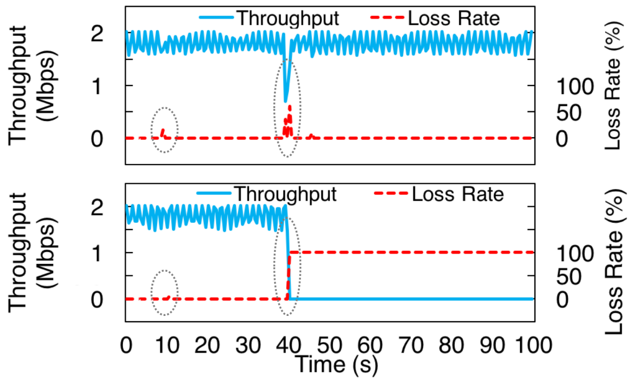

Fig. 8. Throughput and loss rate across UDP flows with Lancet (top) and Gen-R3 (bottom) on testbed.

protection. Here, backup paths are calculated in advance for traffic from each source s to every destination t. When a particular path fails (e.g., an underlying link, or a node fails), the traffic is diverted to those back-up paths. FFC assigns bandwidth to flows so the assignment can be handled under any scenario involving the simultaneous failure of up to f links and f_n nodes.

Considering all f failures with FFC may be conservative. A parallel work [12] uses financial risk theory to design FFC so as to minimize the expected loss function due to inadequate bandwidth assignment for the worst percentage, say $(100 - \beta)\%$, of failures. Lancet is complementary in that it considers link-based protection, which involves very different models. Besides the type of protection, there are two major differences. We attempt to find a routing that has network utilization below one for as many failures as possible. It is not easy to adapt the approach of [12] for this purpose, particularly because the expected loss function cannot be bounded as a constant multiple of β percentile. Second, the approach in [12] involves an optimization formulation that explicitly enumerates all scenarios. The number of failure states grows rapidly with the number of failures a networks may simultaneously experience. In fact, our results in §5.4 show that a formulation based on enumeration for link-based protection performs poorly even for GEANT, the smallest topology in our experiments. Enumerating failure scenarios leads to poor performance because the formulation simultaneously models exponentially many routing problems, one for each failure state. In contrast, Lancet's compact union representation enables a large number of scenarios to be represented using a small number of sets, which greatly helps in the tractability of the design problem. In the future, it will be interesting to explore whether Lancet's approach of avoiding explicit enumeration can offer benefits in the context of path-based mechanisms. Finally, we note that Lancet also allows analyzing scenarios that can be handled with an optimal Centralized approach, an important contribution in its own right.

Many earlier works [27, 28, 31, 34, 41] have focused on quickly re-routing traffic to restore connectivity when failures occur, but do not explicitly prevent congestion when traffic is re-routed. Other works [8, 10, 17, 21, 32, 36, 44] only consider robust design for a small number of failure states (e.g., single-link or node failures). Lancet considers congestion-free recovery mechanisms (not just restoring connectivity), and scales to consider the combinatorially many failure states arising from concurrent failures.

Determining optimal ways to route traffic in a demand-invariant manner while minimizing MLU has been well studied [7, 8, 38, 42]. These works could be viewed as providing guarantees on worst-case network performance, assuming adaptation is not permissible. Semi-oblivious traffic engineering [26] picks paths in a demand-invariant manner, but allows flexibility in how traffic is routed across tunnels. Recent works [13, 16] show how to design local failover algorithms to minimize the number of flows on each link under concurrent failures for different restrictions on network response, e.g. using backup arborescences or backup next-hops.

Lancet simultaneously handles discovery and exclusion of scenarios unlike prior work in the optimization community that focuses solely on excluding a pre-specified list of scenarios [6].

8 CONCLUSIONS

In this paper, we have shown that there exists a huge gap between the failures that can be handled by existing congestion free route protection schemes, and what a network can intrinsically tackle. The gap arises because existing methods only allow for design for the worst-case across all f or fewer failure scenarios. We have presented Lancet, a system that determines what scenarios to design for, and supports tractable design for general failure sets, in a manner guided by intrinsic network capability. The evaluations show the promise of Lancet. For Deltacom, Lancet's analysis reveals that Gen-R3 (a generalization of the state-of-the-art scheme) only handles 46.1% of 2-failure and no 3-failure scenario. In contrast, Lancet can support nearly all 2- and 3-failure scenarios (closely matching Centralized).

Lancet combines an offline protection routing design phase, and an online adjustment phase. The design time in the offline phase is acceptable even for the largest topologies, and when designing for a large number of failure scenarios because of Lancet's effectiveness in (i) expressing scenarios as a compact union of sets (e.g., for Deltacom it represents over 466K 3-failure scenarios with only 3 sets); and (ii) its LP that allows tractable design with such a representation. In contrast, design is intractable even for the moderate-sized GEANT topology, with the *Enumeration* approach. Further's Lancet's online phase triggered in response to failures allows for efficient adjustments that do not require running an LP. Emulation experiments on an SDN testbed show the benefits of Lancet are realizable in practice.

9 ACKNOWLEDGEMENTS

We thank our shepherd, Stefan Schmid, and the anonymous reviewers for their feedback which greatly helped improve the paper. This work was funded in part by the National Science Foundation (NSF) Award 1910234.

REFERENCES

[1] Topology zoo. http://www.topology-zoo.org/.

[2] Abilene traffic matrices. http://www.cs.utexas.edu/~yzhang/research/AbileneTM/, 2014.

[3] Inside AT&T's grand plans for SDN. https://www.networkworld.com/article/2866439/sdn/inside-atts-grand-plans-for-sdn.html, 2015.

[4] Cisco WAN automation engine (WAE), 2016. http://www.cisco.com/c/en/us/products/routers/wan-automation-engine/index.html.

[5] Building Express Backbone: Facebook's new long-haul network. https://code.facebook.com/posts/1782709872057497/building-express-backbone-facebook-s-new-long-haul-network/, 2017.

[6] Gustavo Angulo, Shabbir Ahmed, Santanu S. Dey, and Volker Kaibel. Forbidden vertices. *Mathematics of Operations Research*, 40(2):350–360, 2015.

[7] David Applegate and Edith Cohen. Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental tradeoffs. In *Proceedings of ACM SIGCOMM*, pages 313–324, 2003.

[8] David Applegate, Lee Breslau, and Edith Cohen. Coping with network failures: Routing strategies for optimal demand oblivious restoration. In *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '04/Performance '04, pages 270–281, 2004.

[9] Ajay Kumar Bangla, Alireza Ghaffarkhah, Ben Preskill, Bikash Koley, Christoph Albrecht, Emilie Danna, Joe Jiang, and Xiaoxue Zhao. Capacity planning for the google backbone network. In *ISMP 2015 (International Symposium on Mathematical Programming)*, 2015.

[10] Randeep S. Bhatia, Murali Kodialam, T. V. Lakshman, and Sudipta Sengupta. Bandwidth guaranteed routing with fast restoration against link and node failures. *IEEE/ACM Transactions on Networking*, 16(6):1321–1330, December 2008.

[11] Martin Birk, Gagan Choudhury, Bruce Cortez, Alvin Goddard, Narayan Padi, Aswatnarayan Raghuram, Kathy Tse, Simon Tse, Andrew Wallace, and Kang Xi. Evolving to an SDN-enabled isp backbone: key technologies and applications. *IEEE Communications Magazine*, 54(10):129–135, 2016.

[12] Jeremy Bogle, Nikhil Bhatia, Manya Ghobadi, Ishai Menache, Nikolaj Bjorner, Asaf Valadarsky, and Michael Schapira. Teavar: Striking the right utilization-availability balance in wan traffic engineering. In *Proceedings of ACM SIGCOMM*, 2019. (to appear).

[13] Michael Borokhovich, Yvonne-Anne Pignolet, Stefan Schmid, and Gilles Tredan. Load-optimal local fast rerouting for dense networks. *IEEE/ACM Transactions on Networking*, 26(6):2583–2597, 2018.

[14] Yiyang Chang, Sanjay Rao, and Mohit Tawarmalani. Robust validation of network designs under uncertain demands and failures. In *14^{th} USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 347–362, 2017.

[15] Michele Conforti, Gerard Cornuejols, and Giacomo Zambelli. *Integer Programming*. Springer Publishing Company, Incorporated, 2014.

[16] Klaus-Tycho Foerster, Yvonne-Anne Pignolet, Stefan Schmid, and Gilles Tredan. Casa: congestion and stretch aware static fast rerouting. In *Proceedings of IEEE INFOCOM*, pages 469–477, 2019.

[17] Bernard Fortz and Mikkel Thorup. Robust optimization of OSPF/IS-IS weights. In *Proceedings of International Network Optimization Conference*, pages 225–230, 2003.

[18] Monia Ghobadi and Ratul Mahajan. Optical layer failures in a large backbone. In *Proceedings of the 2016 Internet Measurement Conference*, pages 461–467, 2016.

[19] Phillipa Gill, Navendu Jain, and Nachiappan Nagappan. Understanding network failures in data centers: Measurement, analysis, and implications. In *Proceedings of ACM SIGCOMM*, pages 350–361, 2011.

[20] Ramesh Govindan, Ina Minei, Mahesh Kallahalla, Bikash Koley, and Amin Vahdat. Evolve or die: High-availability design principles drawn from googles network infrastructure. In *Proceedings of ACM SIGCOMM*, pages 58–72, 2016.

[21] Fang Hao, Murali Kodialam, and T. V. Lakshman. Optimizing restoration with segment routing. In *Proceedings of IEEE INFOCOM*, pages 1–9, April 2016.

[22] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. Achieving high utilization with software-driven wan. In *Proceedings of ACM SIGCOMM*, pages 15–26, 2013.

[23] Chi-Yao Hong, Subhasree Mandal, Mohammad Al-Fares, Min Zhu, Richard Alimi, Kondapa Naidu B., Chandan Bhagat, Sourabh Jain, Jay Kaimal, Shiyu Liang, Kirill Mendelev, Steve Padgett, Faro Rabe, Saikat Ray, Malveeka Tewari, Matt Tierney, Monika Zahn, Jonathan Zolla, Joon Ong, and Amin Vahdat. B4 and after: Managing hierarchy, partitioning, and asymmetry for availability and scale in google's software-defined wan. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, pages 74–87, 2018.

[24] Gurobi Optimization Inc. Gurobi optimizer reference manual, 2016. http://www.gurobi.com.

[25] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jon Zolla, Urs Hölzle, Stephen Stuart, and Amin Vahdat. B4: Experience with a globally-deployed software defined wan. In *Proceedings of ACM SIGCOMM*, pages 3–14, 2013.

[26] Praveen Kumar, Yang Yuan, Chris Yu, Nate Foster, Robert Kleinberg, Petr Lapukhov, Chiun Lin Lim, and Robert Soulé. Semi-oblivious traffic engineering: The road not taken. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 157–170, 2018.

[27] Kin-Wah Kwong, Lixin Gao, Roch Guérin, and Zhi-Li Zhang. On the feasibility and efficacy of protection routing in ip networks. *IEEE/ACM Transactions on Networking*, 19(5):1543–1556, October 2011.

[28] Karthik Lakshminarayanan, Matthew Caesar, Murali Rangan, Tom Anderson, Scott Shenker, and Ion Stoica. Achieving convergence-free routing using failure-carrying packets. In *Proceedings of ACM SIGCOMM*, pages 241–252, 2007.

[29] Hongqiang Harry Liu, Srikanth Kandula, Ratul Mahajan, Ming Zhang, and David Gelernter. Traffic engineering with forward fault correction. In *Proceedings of ACM SIGCOMM*, pages 527–538, 2014.

[30] Athina Markopoulou, Gianluca Iannaccone, Supratik Bhattacharyya, Chen-Nee Chuah, Yashar Ganjali, and Christophe Diot. Characterization of failures in an operational ip backbone network. *IEEE/ACM Trans. Netw.*, 16(4):749–762, 2008.

[31] P. Pan, G. Swallow, and A. Atlas. Fast Reroute Extensions to RSVP-TE for LSP Tunnels. RFC 4090, May 2005.

[32] Michal Pióro and Deepankar Medhi. *Routing, Flow, and Capacity Design in Communication and Computer Networks*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. ISBN 0125571895.

[33] Rahul Potharaju and Navendu Jain. When the network crumbles: An empirical study of cloud network failures and their impact on services. In *Proceedings of the 4th Annual Symposium on Cloud Computing*, SOCC '13, pages 15:1–15:17, 2013.

[34] M. Shand and S. Bryant. IP Fast Reroute Framework. RFC 5714, January 2010.

[35] R. K. Sinha, F. Ergun, K. N. Oikonomou, and K. K. Ramakrishnan. Network design for tolerating multiple link failures using Fast Re-route (FRR). In *2014 10th International Conference on the Design of Reliable Communication Networks (DRCN)*, pages 1–8, April 2014.
[36] Martin Suchara, Dahai Xu, Robert Doverspike, David Johnson, and Jennifer Rexford. Network architecture for joint failure recovery and traffic engineering. *SIGMETRICS Perform. Eval. Rev.*, 39(1):97–108, 2011.
[37] Daniel Turner, Kirill Levchenko, Alex C. Snoeren, and Stefan Savage. California fault lines: Understanding the causes and impact of network failures. In *Proceedings of the ACM SIGCOMM 2010 Conference*, pages 315–326, 2010.
[38] Hao Wang, Haiyong Xie, Lili Qiu, Yang Richard Yang, Yin Zhang, and Albert Greenberg. COPE: Traffic engineering in dynamic networks. In *Proceedings of ACM SIGCOMM*, pages 99–110, 2006.
[39] Ye Wang, Hao Wang, Ajay Mahimkar, Richard Alimi, Yin Zhang, Lili Qiu, and Yang Richard Yang. R3: Resilient routing reconfiguration. In *Proceedings of ACM SIGCOMM*, pages 291–302, 2010.
[40] R.Kevin Wood. Deterministic network interdiction. *Mathematical and Computer Modelling*, 17(2):1–18, January 1993.
[41] B. Yang, J. Liu, S. Shenker, J. Li, and K. Zheng. Keep forwarding: Towards k-link failure resilient routing. In *Proceedings of IEEE INFOCOM*, pages 1617–1625, April 2014.
[42] C. Zhang, Zihui Ge, J. Kurose, Y. Liu, and D. Towsley. Optimal routing with multiple traffic matrices tradeoff between average and worst case performance. In *Network Protocols, 2005. ICNP 2005. 13th IEEE International Conference on*, 2005.
[43] Yin Zhang, Zihui Ge, Albert Greenberg, and Matthew Roughan. Network anomography. In *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*, pages 30–30, 2005.
[44] Jiaqi Zheng, Hong Xu, Xiaojun Zhu, Guihai Chen, and Yanhui Geng. We've got you covered: Failure recovery with backup tunnels in traffic engineering. In *2016 IEEE 24th International Conference on Network Protocols (ICNP)*, pages 1–10, 2016.

A APPENDIX

Design with multiple traffic classes (§5.3). Designing a protection routing for multiple traffic classes involves minor changes to LP (H) and is presented below. Let d^h and d^l represent high and low priority traffic matrices, with d^h_{st} and d^l_{st} representing the relevant traffic from s to t. The formulation determines the largest Z such that the network can handle a traffic matrix D where D_{st} is $d^h_{st} + Zd^l_{st}$ (i.e., the network can carry all high priority traffic, and a fraction Z of low priority traffic). $Z \geq 1$ indicates all low priority traffic can be carried. Setting d^h to zero produces the special case where there is a single class of traffic, when Z would share an inverse relationship with the MLU metric. The protection routing has parameters (r^h, r^l, p, a), where r^h and r^l represent flows corresponding to high and low priority traffic from s to t. The formulation is similar to (H) except that r^l_{st} only need carry a fraction Z of the d^l_{st} traffic, however link capacity constraints must be strictly met.

$$
\begin{aligned}
\text{(G)} \quad & \max_{r^h, r^l, p, a, Z} \quad Z \\
\text{s.t.} \quad & r^h_{st} \text{ is a flow of } d^h_{st} \text{ from } s \text{ to } t. \quad \forall s, t \in V \\
& r^l_{st} \text{ is a flow of } Zd^l_{st} \text{ from } s \text{ to } t. \quad \forall s, t \in V \\
& p_l \text{ is a flow of } a_l \text{ from } i \text{ to } j. \quad \forall l \in E, l = \langle i, j \rangle \\
& \sum_{s,t} r^h_{st}(e) + \sum_{s,t} r^l_{st}(e) + \sum_{l \in E} x_l p_l(e) \\
& \quad \leq c_e(1 - x_e) + a_e x_e \quad \forall x \in X, e \in E \\
& a_e \geq 0 \quad \forall e \in E \\
& Z \geq 0
\end{aligned}
$$

Proof of Proposition 3 (§6). Since (r, p, a, U) (parameters prior to failure) is feasible for LP (H), it satisfies every constraint of (H) for $G = (V, E)$ when protecting against all failure scenarios defined in X. The proof proceeds by showing that (r', p', a', U) obtained after the failure of ϵ and adjustments described in §6 satisfies every constraint of LP (H) for the graph $G' = (V, E - \epsilon)$ and for failure scenarios in X with ϵ failing.

First, we show that r'_{st} is a unit flow from s to t for all s, t pairs. For convenience, let's define a function $g(i, s, t)$ as:

$$g(i,s,t) = \begin{cases} 1 & i = s \\ 0 & i \neq s, i' \neq t \\ -1 & i = t \end{cases} \tag{14}$$

Then, the flow balance constraint on r_{st} can be written as:

$$\sum_{j';\langle i',j'\rangle \in E} r_{st}(i',j') - \sum_{j';\langle j',i'\rangle \in E} r_{st}(j',i') = g(i',s,t) \tag{15}$$

We now verify that the flow balance constraint for r'_{st} is met:

$$\begin{aligned}
&\forall i', s, t, \\
&\sum_{j';\langle i',j'\rangle \in E\setminus\{\epsilon\}} r'_{st}(i',j') - \sum_{j';\langle j',i'\rangle \in E\setminus\{\epsilon\}} r'_{st}(j',i') \\
&= (\sum_{j';\langle i',j'\rangle \in E\setminus\{\epsilon\}} r_{st}(i',j') - \sum_{j';\langle j',i'\rangle \in E\setminus\{\epsilon\}} r_{st}(j',i')) + \\
&\frac{r_{st}(\epsilon)}{a_\epsilon}(\sum_{j';\langle i',j'\rangle \in E\setminus\{\epsilon\}} p_\epsilon(i',j') - \sum_{j';\langle j',i'\rangle \in E\setminus\{\epsilon\}} p_\epsilon(j',i')) \\
&= \begin{cases} g(i',s,t) - r_{st}(\epsilon) + \frac{r_{st}(\epsilon)}{a_\epsilon} * a_\epsilon & i' = u \\ g(i',s,t) + \frac{r_{st}(\epsilon)}{a_\epsilon} * 0 & i' \neq u, i' \neq v \\ g(i',s,t) + r_{st}(\epsilon) - \frac{r_{st}(\epsilon)}{a_\epsilon} * a_\epsilon & i' = v \end{cases} \\
&= g(i',s,t)
\end{aligned} \tag{16}$$

Similarly, we can show that p'_l is a flow of a'_l from i to j for all $l = \langle i, j\rangle \in E$.

Second, we show that updated routing and future protection will not traverse ϵ. By definition, $\forall s,t : r'_{st}(\epsilon) = 0$ and $\forall e \in E\setminus\{\epsilon\} : p'_e(\epsilon) = 0$. So link ϵ will not be used in the updated routing and protection.

Third, we show that (r', p', a', U) still satisfies the capacity constraint for all failure scenarios x with $x_\epsilon = 1$ in the certified failure set X. From the capacity constraint on ϵ for any failure scenario $x \in X$ with $x_\epsilon = 1$, we have:

$$\sum_{s,t} d_{st} r_{st}(\epsilon) + \sum_{l \in E\setminus\{\epsilon\}} x_l p_l(\epsilon) \leq a_\epsilon \tag{17}$$

From the capacity constraint for $e \neq \epsilon$ for any failure scenario $x \in X$ with $x_\epsilon = 1$, we have:

$$\sum_{s,t} d_{st} r_{st}(e) + p_\epsilon(e) + \sum_{l \in E\setminus\{\epsilon\}} x_l p_l(e) \leq c_e(1 - x_e) + a_e x_e \tag{18}$$

Then the steps below show that the capacity constraints on other links after updating continue to be met, thereby proving the proposition.

$$\begin{aligned}
&\forall e \in E \backslash \{\epsilon\}, \\
&\sum_{s,t} d_{st} r'_{st}(e) + \sum_{l \in E\backslash\{\epsilon\}} x_l p'_l(e) \\
&= \sum_{s,t} (d_{st} r_{st}(e) + d_{st} r_{st}(\epsilon) \frac{p_\epsilon(e)}{a_\epsilon}) + \\
&\sum_{l \in E\backslash\{\epsilon\}} (x_l p_l(e) + x_l p_l(\epsilon) \frac{p_\epsilon(e)}{a_\epsilon}) - x_e \tilde{p}_e(e) \\
&= \sum_{s,t} d_{st} r_{st}(e) + \frac{p_\epsilon(e)}{a_\epsilon} (\sum_{s,t} d_{st} r_{st}(\epsilon) + \sum_{l \in E\backslash\{\epsilon\}} x_l p_l(\epsilon)) + \\
&\sum_{l \in E\backslash\{\epsilon\}} x_l p_l(e) - x_e \tilde{p}_e(e) \\
&\leq \sum_{s,t} d_{st} r_{st}(e) + \frac{p_\epsilon(e)}{a_\epsilon} a_\epsilon + \sum_{l \in E\backslash\{\epsilon\}} x_l p_l(e) - x_e \tilde{p}_e(e) \\
&= (\sum_{s,t} d_{st} r_{st}(e) + p_\epsilon(e) + \sum_{l \in E\backslash\{\epsilon\}} x_l p_l(e)) - x_e \tilde{p}_e(e) \\
&\leq c_e(1 - x_e) + a_e x_e - x_e \tilde{p}_e(e) \\
&= c_e(1 - x_e) + a'_e x_e
\end{aligned} \tag{19}$$

Received August 2019; revised September 2019; accepted October 2019