On the Implications of Routing Models on Network Optimization

Yvonne-Anne Pignolet[1] Stefan Schmid[2] Gilles Tredan[3]
[1] DFINITY, Switzerland [2] Faculty of Computer Science, University of Vienna, Austria [3] CNRS-LAAS, France

***Abstract*—In network optimization problems, from traffic engineering to network monitoring, the routing model is typically considered as something given and frozen. This paper is motivated by the fundamental question how the ability to *change* and *optimize* the routing model itself influences the efficiency at which communication networks can be operated. To this end, we identify two main dimensions of a routing model: *consistency* (of a single route) and *coherence* (of sets of routes). We present analytical results on the impact of the routing model on the achievable route diversity as well as on the runtime of solving optimization problems underlying different case studies. We also uncover that it can sometimes be beneficial to *artificially* restrict the routing model, to significantly reduce the computational complexity without negatively affecting the route diversity much.**

***Index Terms*—Routing, path diversity, modelling, algorithms, complexity, NP-hardness, SDN, IP, MPLS**

I. Introduction

While most communication networks feature a routing mechanism that supports the delivery of packets from their source s to their destination t across a multi-hop network, they can differ significantly in how and to what extent the routes taken by packets can be controlled. Routing models have evolved dramatically over the last years, and indeed, more flexible routing models have been a main driver behind recent innovations in networking [16]. At one end of the spectrum lie traditional networks in which packets are routed along shortest paths and in a *(IP prefix) destination-based* manner (using protocols such as, e.g., OSPF or IS-IS). Such networks often provide only limited flexibility: destination-based routes are confluent, and flows towards the same destination remain on the same route once they meet. Furthermore, the only knob available to the operator to influence the routes is to set the *link weights* based on which the shortest paths are selected. In the late 1990s, increasing traffic volumes and the need for a more reliable performance led to the design of advanced traffic engineering schemes. For example, MPLS enabled traffic engineering on a *per-flow* basis, supporting the definition of more general (not necessarily shortest) routes [9], [29], [30], [39]. Offline traffic engineering engines, source-routing protocols, approaches which establish paths via a Path Computation Element (PCE), as well as emerging Software-Defined Networks (SDNs) and OpenFlow further facilitated a *direct* control over the forwarding tables. In particular, besides forwarding, SDN and OpenFlow also support a fairly general *modification* of packet headers, enabling more advanced routing services: packets cannot only be forwarded along simple paths, but may visit a given node multiple times along their route, i.e., form complex *walks* [2]. Also source routing and segment routing mechanisms [5] provide a fine-grained flow management capabilities, requiring only little state in the core network.

But not only the technology and specific protocols can influence the flexibility at which routes can be selected, but also the policy. For example, inter-domain routes computed by the Border Gateway Protocol (BGP) are usually restricted to be *valley-free* and often fulfill the Gao-Rexford conditions [19], meeting the business objectives of Internet Service Providers. But policies also restrict routes inside a network, e.g., forcing traffic to be routed via certain waypoints (e.g., a middlebox for deep packet inspection, a firewall, or a WAN optimizer) between source and destination [28], e.g. using service function chaining and IPv6 segment routing [1], [17], or to *explicitly avoid* such waypoints (e.g., routing via certain countries or via certain network elements such as route reflectors). Thus the routes taken by packets can depend on aspects beyond the pure network topology.

In addition to these classic examples, there are many additional important routing models. To just name a few, for datacenters there exist optimized routing protocols such as RIFT, a dynamic routing protocol tailored for Clos and fat-tree network topologies [12], or BGP-SPF [32], a shortest path first protocol similar to OSPF for routing both in the fabric of data centers and their interconnect, meeting requirements of massively scaled data centers. Layer-2 cryptocurrency networks (e.g., Lightning) usually rely on shortest paths routing where costs are described by an affine function [11]. Low-power and lossy networks may use RPL [41], a protocol which can be implemented in IPv6, using a centralized control point. And wireless mesh networks may use Babel [10], a shortest path protocol which relies on distance vectors and avoid loops.

This paper is motivated by the observation that while it is intuitively clear that different routing models have an impact on the efficiency at which a network can be operated, we lack a systematic comparison of existing and future routing models, not only with respect to the routing flexibility they provide and the path diversity they support, but also with respect to the cost resp. quality and time complexity at which network optimization problems can be solved in these different routing models. Indeed, as we show in this paper, and complementing the perspective usually taken in the literature, many aspects of network optimization to a large extent do not only depend

ISBN 978-3-903176-28-7

on the complexity of the underlying network topology but also on the routing model used. Also, some optimizations are for example only possible if the underlying routing supports a set of capabilities (e.g., multi-path, TE-engineered routes, constraint-based paths, support of objective functions, power-aware paths, etc.).

Our main ***contributions*** are a systematic study of the impact of the routing model on fundamental network optimization problems. To this end, we first propose a taxonomy of (unicast) routing models, along two dimensions, *consistency* and *coherence*:

1) *Consistency (Π):* Consistency relates to properties of a single route, e.g., some routing models require shortest paths or that a node only appears at most once along a route, or policies may dictate that packets should (not) go through a certain waypoint.
2) *Coherence (Σ):* Coherence refers to constraints on how multiple routes relate to each other. For example, packets must travel confluent routes towards the same destination.

We illustrate how the consistency and coherence influence the diversity of routes on a given network and propose a hierarchy of coherence models which impacts the achievable performance. Furthermore, we demonstrate how routing algebra properties can impact coherence. We consider two canonical problems as case studies. (i) *Traffic Engineering:* A main goal of Traffic Engineering is to optimize load over multiple paths, when available. Thus we consider the impact of routes on network load in this case study. (ii) *Monitoring:* Dependable communication networks require (automated) monitoring, e.g., to check the availability of links or entire paths. The objective in this case is hence to keep track of the network state at low monitoring cost. Moreover, we show how routing models can help to prove new properties using OSPF as an example.

Our approach leads to several interesting observations. For example, we find that the same optimization problem on the same network can be polynomial-time solvable under one routing model and NP-complete under another routing model. This uncovers an optimization opportunity: by restricting the routing model *artificially*, i.e., by introducing constraints on the routing which do not negatively affect the route diversity by much, the computational complexity may be reduced significantly.

The remainder of this paper is organized as follows. Section II introduces a taxonomy for routing models. Section III derives general properties and Section IV presents implications in two case studies. In Section V, we discuss a specific protocol, OSPF. After reviewing related work in Section VII, we conclude in Section VIII.

II. A Taxonomy of Routing Models

This section first introduces a basic network model and then presents and discusses a taxonomy to classify routing protocols. Throughout this section we will focus on unicast routing.

A. Preliminaries

We consider a basic model in which the network is represented as an undirected connected graph $G(V, E)$, with $n = |V|$ devices (nodes) and $m = |E|$ communication links connecting them. Nodes and links may have attributes representing costs, constraints, etc assigned to them. An (s, t)-*route* is a sequence of k nodes $r = (v_1 = s, v_2, \ldots, v_k = t)$ such that $(v_i, v_{i+1}) \in E$. Note that a route r does not necessarily have to follow a shortest path and not even a simple path: the route may contain certain nodes multiple times. For example, in the context of service function chaining [23], in order to visit a certain waypoint, such as a firewall, a route may traverse a node to/from the waypoint multiple times, and hence constitutes a *walk* in graph theory terminology. However, for ease of presentation, we first focus on simple (but not shortest) paths, and write $r[v_i, v_j]$ to denote the subsequence of r between nodes v_i and v_j. Section II-E describes how to extend our model to routes in which a node can appear multiple times.

Given two routes r_1, r_2, we denote by $r_1 \cap r_2$ the set of links appearing in both routes. Furthermore, we consider the graph induced by the union of two or more routes, ignoring the order of the links: thus $r \cup r'$ implies an induced network $G(V, E)$, where $V = \{v_1, \ldots, v_k, v'_1, v'_{k'}\}$ and $E = \{(v_i, v_{i+1}) | (v_i, v_{i+1}) \in r \ \vee \ (v_i, v_{i+1}) \in r'\}$.

We refer to the set of all possible routes which differ by at least one edge between any two nodes $x, y \in V$ in G by $R(x, y)$. In addition, we denote by $R(x, .)$ (resp. $R(., y)$) the set of all routes that start at node x (resp. finish at node y). Finally, to emphasize when we consider *all* routes in a network G (not only between certain endpoints), we write $R(G)$.

Given a set of source-destination pairs, a *routing algorithm* produces a set of routes for these pairs. In this paper we do not focus on the algorithms constructing these routes but rather on the properties the resulting route sets exhibit: Hence the routing algorithms are abstract providers of routes. This allows us to first compare all algorithms on a generic basis, and second to classify algorithms using properties on their route sets that are useful from a theoretical and practical perspective.

Definition 1 (Route Set $\mathcal{S}$)**.** *A* route set $S \subset R(G)$ *contains zero, one or several routes* $r_i \in R(G)$ *for each source-destination pair.*

Consistent with this approach, when we study properties of routing models, we will distinguish between *blackbox* properties which can be checked by only observing the routes themselves, as opposed to *whitebox* properties that require additional knowledge on the "infrastructure", e.g., on the network topology, link weights, algorithm parameters, etc.

B. Dimension 1: Consistency

The first dimension of our taxonomy is consistency: a property defined on *individual route* of a route set. Formally, a route r on a graph G is Π-consistent if the predicate $\Pi(r, G)$ defined over the links and nodes of the route r on G evaluates to true. For example, a basic consistency predicate is that all

routes of a route set are of minimum length: shortest path routing is employed by well-known protocols, e.g., OSPF, which ensure that all flows are routed along shortest paths with respect to edge lengths (routing weights). Other examples of consistency properties are related to the policy-compliance of a given flow, e.g., ensuring that a route did traverse certain waypoints, did *not* traverse blacklisted parts of the network, or conforms to business relationships (like valley-freedom).

In general, a consistency predicate Π can be used as a filter: only a subset of all possible routes $R(x, y)$ between x and y may fulfill Π. In the following, let $R_\Pi(x, y) \subset R(x, y)$ denote the set of routes from x to y that are consistent w.r.t. Π. We write $R_\Pi(G)$ and call it the set of all Π-*consistent routes* of a graph G.

Note that many of the consistency predicates used in practice depend on properties of the elements of the underlying network *infrastructure*, such as waypoints, edge type classes or weights representing the cost or latency incurred when using them. In particular, predicates such as "is a shortest path" even require additional information about the infrastructure, beyond the links of the current path: in order to be able to verify that this path is indeed the shortest between a given source and a given destination, we need to know the alternative links (and their weights) in G. However, there are also consistency properties which do not require such additional information, for example a predicate of the form "the route length is at most ℓ" or "a node appears at most once in the route". We will refer to consistency properties which depend on the infrastructure as *whitebox* consistency properties, and to consistency properties which do not require such information, as *blackbox* consistency properties.

Consistency properties (e.g., valley-freedom, waypoint routing, multipathing, etc.) are often described using regular languages [38], [28], [27], e.g., over labels on links and nodes: it is required that all valid routes in the graph adhere to this regular expression. For example, $s.^*w.^*t$ could express that a route from s to t should traverse a waypoint w. Or $(c2p)^*(p2p)?(p2c)^*$ could express a valley-free routing policy where edge labels are used to denote peer-to-peer ($p2p$), provider-to-customer ($p2c$), or customer-to-provider ($c2p$) relationships [28].

Another approach, based on algebraic methods, considers routing policies as a function that selects, from the set of all paths from a source to a destination, preferred paths according to predefined rules, specified by an algebra. In other words, a routing algebra defines a set of "legal" or policy-compliant routes. This definition is broad enough to contain many routing policies, e.g., shortest paths, widest paths, most reliable paths, widest-shortest paths, shortest-widest paths, valley-free paths, etc. A large body of literature analysed routing protocols in such a framework, e.g., [37], [35], [22], [3], to just give some examples.

The crucial components of a *routing algebra* are a partially-ordered commutative semi-group with a compatible infinity element: $A = (W, \phi, \bigoplus, \preceq)$, where W is the set of possible edge weights (i.e., different edges can have different costs), ϕ ($\phi \notin W$) denotes an infinity element assigned to unusable edges/routes, and $\bigoplus$ is a composition operator for weights (e.g., latency related edge costs add up while bandwidth-related edge costs are naturally subject to min/max operations). Given a route we obtain its weight by combining the weights of its constituent edges with $\bigoplus$. $\preceq$ is a partial order for weight comparison of edges and routes. A *preferred route* in the algebra A between two nodes is one with the smallest weight according to $\preceq$. The infinity weight ϕ is compatible with $(W, \bigoplus)$ according to $\preceq$ in the sense that it is absorptive $w \bigoplus \phi = \phi,\ \forall w \in W$ (a route with contains an unusable edge is unusable), and maximal $w \prec \phi,\ \forall w \in W$ (any route without an unusable edge is preferred over a route with an unusable edge).

To give an example, shortest path routing, where valid paths between two nodes minimise the sum of the weights of its constituent edges, corresponds to the algebra $(\mathbb{R}+, \infty, +, \leq)$, where positive edge weights ($\mathbb{R}+$) may describe a property like the latency or cost of this edge, which is added up ($+$) along a route, and shorter routes are better ($\leq$). Widest-path routing prefers paths which have the largest bottleneck capacity, i.e., $(\mathbb{R}+, 0, \min, \geq)$ where positive edge weights ($\mathbb{R}+$) describe the link bandwidth, the total bandwidth provided along a route is the minimum ($\min$) offered on any of its constituent links, and wider paths are better ($\geq$).

C. Dimension 2: Coherence

The second dimension concerns the coherence $\Sigma(R, G)$ of route sets R produced by a routing algorithm. Similarly to the above, we denote by $\mathcal{R}^\Sigma$ the set of route sets that fulfill a coherence predicate Σ. Since Σ describes a relationship between multiple routes, $\mathcal{R}^\Sigma$ is a *set of sets*: each set of routes $R \in \mathcal{R}^\Sigma$ satisfies the coherence predicate Σ.

Note that a coherence predicate compares multiple routes to each other (e.g., if and in which nodes and links they are the same or different). Many important coherence properties do not require references to the infrastructure network G (i.e., they are *blackbox* coherence properties). Take for example destination-based routing, which can be expressed generally as "once two routes towards the same destination meet, they will follow the same route from then onward". Indeed, as we will see, when describing coherence properties, it often matters when two routes *meet*.

There are also *whitebox* coherence properties, which require knowledge about the infrastructure. For example, consider a network which includes two waypoints w_1, w_2 of the same type (e.g., an intrusion detection system). A coherence property may require that two flows, one from s_1 to t_1 and one from s_2 to t_2, either *both* go through w_1 or *both* go through w_2.

Some natural coherence properties are the following:

Definition 2 (Basic Coherence Property Examples)**.** *Let G be a graph and Σ a coherence predicate. Basic coherence properties include:*

(i) ***Multi*** *($\star$): In this model, an arbitrary subset of $R(G)$ is valid: for each set of routes $R \in \mathcal{R}^*$, it holds that $R \subseteq R(G)$. In particular, more than one route between a source and destination node may be included in R.*

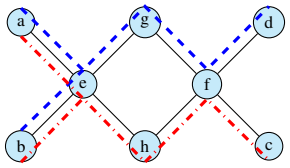

Fig. 1. Example: The routes from a and b to c and d are confluent: for each destination they follow the same route once they meet. The routes are not contained as paths between e and f differ.

(ii) ***Any*** *(!): In this model, we only require that at most one route between any source-destination pair exists, and there are no other constraints on the route: for each set of routes* $R \in \mathcal{R}^!$*, it holds that* $|R(x,y)| \leq 1\ \forall x,y \in V(G)$*.*
(iii) ***Confluent*** *(*$>$*): In the confluent model, the route choice at each node is source-invariant. I.e., the next hop is determined by the destination. Let* $R \in \mathcal{R}^{>}(G)\ \forall w,y \in V, \forall r,r' \in (R \cap \mathcal{R}(.,y))$ *it holds that* $w \in r \cap r' \Rightarrow r[w,y] = r'[w,y]$*. Note that such an element* $w \in r' \cap r$ *has to carry the same inport and attribute in both sequences. This also holds for all elements in* $r[w,y]$ *and* $r'[w,y]$*.*
(iv) ***Contained*** *(*$\subseteq$*): In the contained model, any two routes share at most one contiguous subsequence* $\forall z,w \in V, \forall r,r' \in R$ *it holds that* $\{z,w\} \in r \cap r' \Rightarrow r[z,w] = r'[z,w]$*.*
(v) ***Forest*** *(*T*)* ***and Graph*** *(*$\mathcal{G}$ *): In the most constrained model, the union of* all *routes in any set* $R \in \mathcal{R}^T$ *is a forest. More generally, the coherence restriction could be extended* $R \in \mathcal{R}^{\mathcal{G}}$ *for other graph classes, e.g., DAGs or planar graphs.*
(vi) ***Symmetric Routing*** *(*$\leftrightarrow$*): A set of routes* R *is symmetric if it holds for all source-destination pairs* (s,t)*, if a route* r *from* s *to* t *is in* R *then the reverse route from* t *to* s *is in* R *too.*

Note that these coherence properties differ in terms of the path subsequences shared by the different routes. E.g., multi $\star$ and any ! do not have any constraints on shared subsequences, while confluent $>$, contained $\subseteq$, forest T, symmetric $\leftrightarrow$ require shared subsequences to adhere to rules. In symmetric $\leftrightarrow$ routing, the same routes between two nodes are used in both directions. It is easy to see that for example routes adhering to T routing models are always symmetric. For an example where the confluent $>$ and the contained $\subseteq$ routing models are different, see Figure 1 with valid confluent routes which are not contained.

D. Combining Consistency and Coherence

There can hence be two different types of restrictions on a route set: related to consistency Π and related to coherence Σ. Both limit the classes of route sets $R^{\Sigma}_{\Pi}(G)$ through a given network G. For any set of routes $R \in R^{\Sigma}_{\Pi}(G)$, the routes R *jointly* fulfill Σ, and each route *individually* fulfills Π. For many networking problems $R^{\Sigma}_{\Pi}(G)$ serves as a better class definition when reasoning about network algorithms (e.g., for traffic engineering, monitoring, etc.) than the network topology or consistency and coherence properties individually (as discussed in more details later). Motivated by our observations, we will define a route set to adhere to a routing model as follows.

Given a message m **from** s **to** t **(code for node** v**)**

1: $C = \arg\min_{i \in V(v)}(d(i,t))$ /*find relay candidates*/
2: **if** $|C| > 1$ /* more than one relay candidate */ **then**
3: $next\_hop =$ $argmin_{i \in C}(i)$ | $argmin_{i \in C}(|i-s|)$ | $random(i \in C)$
4: forward packet to $next\_hop$
5: **else**
6: /* algorithm continues */

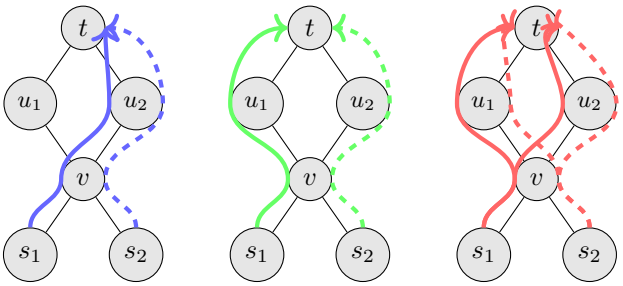

Fig. 2. Prototype of a shortest path routing algorithm, and impact of the tie-breaking. Produced routes are always consistent. Depending on the nature of the tie breaking, the set of produced routes will have different properties. In the blue example, routes will always be confluent, whereas in the green examples, the produced route set might be *Any* (!). A non-deterministic tie-breaking like in red may produce *Multi* ($\star$).

Definition 3 (Routing Model $\mathcal{M}$)**.** *Let* Π *be a consistency criterion and* Σ *a coherence criterion. The* routing model $\mathcal{M}^{\Sigma}_{\Pi}(G)$ *consists of all route sets* S *that satisfy* Π *and* Σ*, i.e.,* $S \subset R_{\Pi}$ *and* $S \in \mathcal{R}^{\Sigma}$*.*

The fewer constraints we have on coherence and consistency for a routing model, the higher the number of schemes satisfying the predicates. In other words, less constrained models $\mathcal{M}^{\Sigma}_{\Pi}$ contain route sets of larger size and more route sets.

In Figure 2 we illustrate the connection between a routing algorithm and the resulting routing model. It presents a partial prototype of a shortest path routing algorithm: line 1 ensures that paths are selected according to this consistency criteria. However, depending on the contents of line 3, the resulting routing model might end up having various coherence properties. Below, we depict the consequences of 3 example implementations, together with the possible resulting routes.

This example demonstrates the importance and impact of tie-breaking in such protocols. Other protocols may not fit this prototype algorithm, yet produce route sets that obey the same logic. Let us also underline that many routes are affected by line 3, for instance in a regular hypercube there are 2^{k-1} shortest paths to nodes at distance k: the actual route will be selected through $k-1$ successive evaluations of line 3.

E. Generalization Beyond Simple Paths

Our definitions can easily be extended beyond simple paths. In this case, we do not only have to account for the current

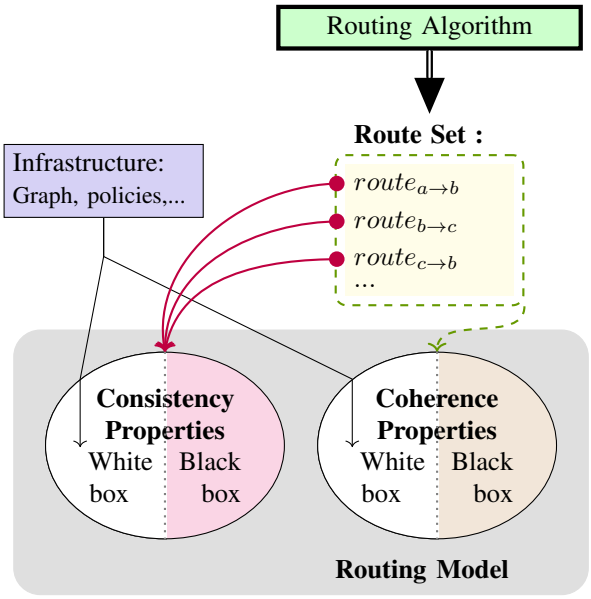

Fig. 3. Overview of our taxonomy: A routing algorithm produces a set of routes. Standard approaches mostly focus on the properties of routes taken *individually*, which we refer to as consistency properties. Conversely, coherence relates to properties of the set of routes taken as a whole. For each of these properties, we distinguish black box properties (that can be checked without infrastructure knowledge, e.g., whether routes form simple paths) and white box properties (that refer to a specific topology, e.g., shortest paths). The combination of consistency and coherence properties defines the routing model.

node and destination, but also have to consider the *packet's state*, e.g., a flag denoting whether a packet is on its route "before or after the waypoint", as well as the *router's state* (e.g., counters). These states can be modelled as additional *attributes*, which can be matched in forwarding rules of routing protocols. Analogously, the resulting routes can be annotated with these attributes. Thus the notion of two routes "meeting" at a node v in our taxonomy needs to be generalised to refer to these annotations. I.e., two routes meet if in addition to visiting the same node they feature the same attributes at this node. Another way of looking at this, is to consider a node v occurring on routes with attributes a_1 and a_2 as two different instances of a node, v_1 and v_2 respectively. On the multi graph induced by the set of annotated nodes and links the routes are thus simple paths and the original definitions can be used.

F. Summary and Taxonomy

In summary, we propose to study sets of routes as the generic consequence of any routing algorithm. This allows us to focus only on the consequences of these algorithms in our taxonomy (Figure 3), regardless of the internal logic that led a given algorithm to produce a particular set of routes. Inside this model, we identify two canonical categories of properties to describe those route sets: *(i)* consistency properties, describing properties satisfied by each individual route of the considered set (e.g., properties that paths are *simple*, *shortest*, or at most k hops long). And *(ii)* coherence properties, describing properties satisfied by the route set as a whole (e.g., the routes are *symmetric*, *confluent*). We can further distinguish two types of coherence properties. First, internal coherence properties, that can be expressed using only elements of the route set (e.g., the route set is *contained*) and can be seen as the counterpart of consistency (expressed as predicates involving a single route against elements of the infrastructure). Second, generalized coherence properties, that can only be expressed using predicates involving both multiple routes and elements of the infrastructure. We can then define the routing model as the combination of consistency and coherence properties fulfilled by a route set. We illustrate in Figure 2 how a simple shortest path routing algorithm (designed with a predefined consistency criteria) can provide route sets belonging to different routing models because of the different coherence properties induced by its tie-breaking behavior.

III. General Analysis and Implications

This section presents an analysis of the impact of the routing model, based on our taxonomy, namely route diversity, hierarchies and the interdepence of routing algebra properties and coherence.

A. Notions of Route Diversity

Given our taxonomy, we can refine the intuitive notion of "path diversity". First, the term *route diversity* is more accurate to represent the flexibility offered by a variety of routes between a source and destination since routes do not necessarily have to follow simple paths and may contain certain nodes repeatedly, as discussed earlier. Route diversity can come in different flavors.

If we consider consistency only, we can define $(s,t)_\Pi$-*route-diversity* to count the number of different Π-consistent routes a packet travelling from source s can take to reach its destination t on a graph G. E.g., there might be several shortest paths between s and t, or several valley-free paths. The higher this number, the more distinct Π-consistent route sets exist.

For a given route set R, we can define the (s,t)-*route-set-diversity* to be $|R(s,t)|$, the number of distinct routes between s and t in R. We further define (s,t)-*subsequence-route-set-diversity* as the number of different routes a packet travelling through s and t can take, according to a route set R (regardless of the source and destination of packets), i.e., $|\{r | r \in R \ \wedge \ len(r[s,t]|) > 0\}|$.

Note that the subsequence-route-set-diversity definition is more general in the sense that depending on a routing model certain paths between s and t may only be traversed by packets emitted by s' and not by packets originating at s. To indicate the complexity and quality of some problems one of the two may be more appropriate. E.g., for the monitoring problem described in our case study (see Section IV) the subsequence-route-diversity of a routing model matters. This is due to the fact that route measurements can be used to infer metrics of the links they contain and thus a $r[u,v]$ subsequence of a (s,t)-route r traversing u and v can monitor links in $r[u,v]$, even though there might be no (u,v)-route that contains $r[u,v]$.

To analyze the impact of Π-consistency and Σ-coherence not just on a pair of nodes and of single route set but on the

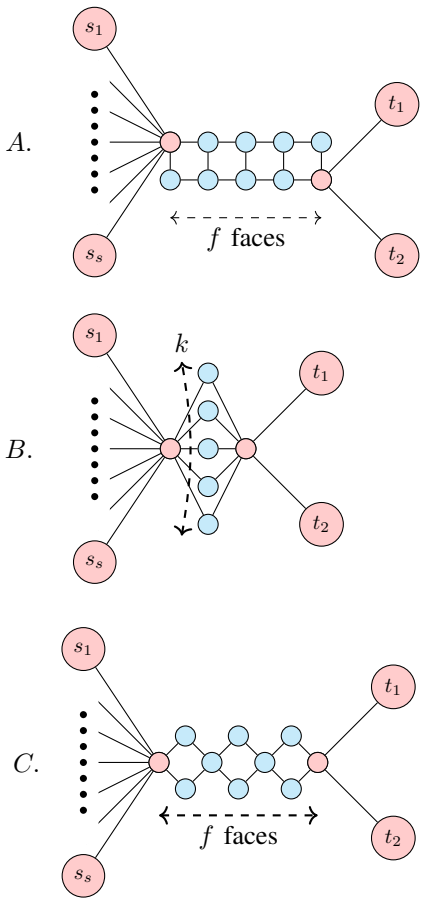

Fig. 4. A. ladder graph, B. k-connected subgraph, C. sausage graph. These graphs illustrate the impact of the choice of routing policies on the route diversity. The route diversity in these graph is high for some models and low for others, implying a high variance in network problem solution quality for different routing models.

number of route sets that conform with Π and Σ , we can define the *routing model diversity* as $diversity^{\Sigma}_{\Pi} := |\mathcal{M}^{\Sigma}_{\Pi}|$, the number of route sets that adhere to $\mathcal{M}^{\Sigma}_{\Pi}$.

To illustrate how the routing model affects route diversity, let us consider a few fundamental examples, depicted in Figure 4. Let S be the set of route sets with routes between a set of s sources $s_1, \dots s_s$ and two destinations t_1 and t_2. The three networks we study only differ from one another in the blue central set of nodes connecting sources and targets. In A, this central part is organized as a ladder spanning f faces; it is 2-connected. In B, the blue nodes connect sources and destinations by k parallel routes of the same length.In C, this central part is a chain of f faces only connected through single nodes, and is 2-(edge-)connected. Table I presents the number of possible routing schemes between sources and targets that follow a specific routing model in those graphs.

The first observation drawn from these results concerns the restricting power of routing models. While all three networks exhibit a number of simple-path route sets growing exponentially with the number of sources in the ! model, this combinatorial explosion is no longer possible under more restrictive routing models.

Routing Model \ Graph	A	B	C
$\lvert R^{!} \cap S\rvert$	$(f+2)^{s+2}$	$(k)^{s+2}$	$(2)^{f+s+2}$
$\lvert R^{>} \cap S\rvert$	$c(f+2)^2$	k^2	2^{2+f}
$\lvert R^{\subseteq} \cap S\rvert$	$(f+2)$	k^2	2^f
$\lvert R^{T} \cap S\rvert$	$(f+2)$	k	2^f

TABLE I
IMPACT OF COHERENCE ON THE NUMBER OF POSSIBLE SHORTEST PATH ROUTE SETS FOR GRAPHS IN FIGURE 4.

The impact of the topology on those restrictions can be seen very well on outerplanar graphs: while graph B presenting parallel routes quickly becomes quadratic and then linear under the most restrictive tree T routing model, the chain of faces C still allows for the number of tree route sets to be exponential in the number of faces (which in turn can be as large as $\Omega(n)$, even on (sparse) outerplanar graphs).

Similarly to the coherence example above, consistency influences the route model diversity. Consider two nodes s, t on an odd cycle. Under shortest path consistency there is exactly one route between s and t possible and thus every any !-coherent route set is also confluent $>$. A more relaxed consistency criteria may allow two routes. In this case, we could construct an !-coherent route set that is not confluent by adding the long routes for the two neighbors of t.

B. Coherence Hierarchy

We first observe that some of the properties defined in the previous section form a hierarchy of increasingly flexible routing. In particular the number of possible route sets $\mathcal{R}^{\Sigma}_{\Pi}(x, y)$ between two nodes x and y (or on G in general), depends, besides the topology, on the routing model defined by Σ and Π.

In the following, we prove the *hierarchy of coherence* (for any consistency property Π as it affects all sets the same way).

Theorem 1. *Let G be a graph. We have $\mathcal{R}^{T}_{\Pi}(G) \subseteq \mathcal{R}^{\subseteq}_{\Pi}(G) \subseteq \mathcal{R}^{>}_{\Pi}(G) \subseteq \mathcal{R}^{!}_{\Pi}(G) \subseteq \mathcal{R}^{\star}_{\Pi}(G)$, for any consistency Π.*

Proof. To improve readability, we omit the subscript Π in the proof. Only routes that satisfy Π are considered in the following. *Containment*: Let $S \in R^{T}(G)$ be the union of all routes, forming a tree. In particular all routes that pass through a particular node w form a tree and thus at most one contiguous subsequence for each pair of routes, hence $S \in R^{\subseteq}(G)$.

Let $S \in R^{\subseteq}(G)$ and let w be a node. All pairs of routes containing w have at most one contiguous subsequence. This holds in particular for routes destined for w: $S \in R^{<}(G)$.

Let $S \in R^{<}(G)$. By contradiction assume that there exists x, y such that $|R(x, y) \cap S| > 1$. Let r and r' two such routes. Since r and r' are in $R(., y)$, we know that $r \cup r'$ do not split after they meet. Since $x \in r \cap r'$, we conclude that $r[x, y] = r'[x, y] = r = r'$.

We note that if the network is a tree, all models become the same (*possible equality*). We construct an example where all those coherence sets are equal. Let T be a tree, and Σ

a classic "shortest path" consistency criteria. Observe that $\forall S \in R^\star(T), \bigcup_{r\in S} r$: we have $R^\star(T) \subseteq R^T(T)$, from which we conclude $R^\star(T) = R^T(T)$, settling the case for the intermediary models any !, confluent $<$ and confluent $\subseteq$. □

The tree example in the proof shows that topologies have an impact on the route diversity. More precisely, a greater link density allows for many routes, and hence allows for many route combinations that populate the routing models hierarchy.

Intuitively, a higher route diversity allows for more possible configurations, some of which may provide more desirable output. This intuition follows from Theorem 1.

Corollary 1. *For any optimization problem, let* $\text{qual}(S, G)$ *denote the quality measure of the best solution achievable on a graph* G *for a given route set* S*. By extension, let* $\text{qual}(R^\Sigma_\Pi, G) = \max_{S\in R^\Sigma_\Pi}(\text{qual}(S, G))$*. Then* $\text{qual}(R^T_\Pi(G), G) \leq \text{qual}(R^\subseteq_\Pi(G), G) \leq \text{qual}(R^>_\Pi(G), G) \leq \text{qual}(R^!_\Pi(G), G) \leq \text{qual}(R^\star_\Pi(G), G)$.

Proof. Consider $qual(R^T_\Pi(G), G)$ and $qual(R^\subseteq_\Pi(G), G)$, and let $S_1 \in R^T_\Pi(G)$ and $S_2 \in R^\subseteq_\Pi(G)$ be (one of) the route sets realizing this optimum. Since $S_1 \in R^T_\Pi(G) \subseteq R^\subseteq_\Pi(G)$, we have in particular $S_1 \in R^\subseteq_\Pi(G)$ and therefore $qual(S_1) \leq \max_{S\in R^\subseteq_\Pi(G)}(qual(S, G)) = qual(R^\subseteq_\Pi(G), G)$. This settles the case for $R^T_\Pi(G)$ and $R^\subseteq_\Pi(G)$. The inequalities for the other cases can be derived analogously. □

C. Equivalence Under Symmetry

If routes are symmetric (using the same links in both directions), confluent routes are contained.

Theorem 2. *A symmetric and confluent routing model ensures that valid routes are contained.*

Proof. For the sake of contradiction, assume the opposite for simple paths, i.e., two valid routes between nodes e and f, where $a - h \in V$ s.t. $g \in R(e, f)$; nodes e, g, f are incident to links in $R(a, d)$, node g is not on any route in $R(b, c)$, and e, f are on routes in $R(b, c)$, e.g., like in Fig. 1. Consider $T_f = \cup_{v\in V} R(v, f)$, the confluent tree leading to f. Since T_f is a tree, and since $g \in R(e, f) \wedge (g, e, f) \in R(a, d)$ we deduce $g \in R(f, a)$ (or $g \in R(f, e)$ w.l.o.g.) due to symmetry. Since $T_b \cup_{v\in V} R(v, b)$ is also a tree, we have $R(f, b) \subset R(e, b) \subset R(c, b) \subset T_b$. Thus $g \in R(c, b)$ and due to symmetry we have a contradiction. The same argument can be generalized for walks using inports and attributes instead of simple paths. □

D. Routing Algebras Can Impact Coherence

As mentioned earlier, a routing algebra can be used to describe consistency properties: for each source-destination pair we can determine whether a route is preferred. Yet most routing algebras do not have an impact on coherence properties of sets of routes. However, some classes of routing algebras can be used to make statements about the kind of coherence properties a route set may *not* be able to satisfy. For example, results from [37], [35] can be interpreted with respect to coherence properties. To this end, we define the set of routes that represent all preferred paths of an algebra A by $\mathcal{R}_A$. An !-coherent route set $S \subseteq \mathcal{R}_A$ *derived* from A contains at most one route per source-destination pair. Furthermore, we need the definition of *regular* routing algebras, which feature a total order $\preceq$ and satisfy *monotonicity*: $w_1 \preceq w_2 \bigoplus w_1,\ \forall w_1, w_2 \in W$ and *isotonicity*: $w_1 \preceq w_2 \implies w_3 \bigoplus w_1 \preceq w_3 \bigoplus w_2,\ \forall w_1, w_2, w_3 \in W$. Monotonicity requires that prepending an edge (or path) of weight w_1 to another edge (or path) of w_2 can only make it less preferred. By commutativity, the same applies to appending edges/paths. Isotonicity, on the other hand, requires $\preceq$ to be compatible with the semigroup $(W, \bigoplus)$ in the following sense: if an edge/path is preferred over some other one, then prepending or suffixing both with a common edge or path maintains this relation. As an example, BGP and IGRP can both be represented by routing algebras. BGP is regular while IGRP is not isotonic and thus not regular [37]; a fact that illustrates how the above definitions can classify real-world routing policies. It also follows from [37] that an algebra A can be implemented by a destination-based routing function on any graph, if and only if A is regular. These results imply that a $\mathcal{R}_A$ representing an algebra A can be turned into a confluent route set by removing some of the routes. In other words, a routing algorithm which selects one next hop based on the destination only and which conforms with algebra A, can exist if and only if A is regular and always produces a confluent route set. In other words, we can make the following observation.

Observation 1. *A routing algorithm that turns a preferred route set* $\mathcal{R}_A$ *into an any* !*-coherent route set* $\mathcal{R}' \subseteq \mathcal{R}_A$ *which is* $confluent >$ *on every network exists if and only if* A *is regular.*

Other aspects that have been studied for routing algebras are their scalability and memory requirements [37], [35]. Route sets that do not represent a regular algebra are incompressible in the sense that their policy does not scale well, as the memory needed to represent the local routing behavior of some node increases with the number of nodes in at least one network topology.

Observation 2. *Given a non-regular routing algebra* A*, there are networks where no* !*-coherent route set derived from* $\mathcal{R}_A$ *for all source-destination pairs can be confluent.*

On the other end of the spectrum, Retvari et al. [35] prove that if and only if a route set represents a monotonic and *selective* algebra, i.e., $w_1 \bigoplus w_2 \in \{w_1, w_2\}$ for each $w_1, w_2 \in W$, this route set adheres to tree coherence and is thus highly compressible.

Observation 3. *A route set* $\mathcal{R}_A$ *representing a monotonic and selective algebra* A *is* T*-coherent.*

E. Summary

To summarize our observations, coherence and consistency can have a large impact on the number of possible route

sets on a given graph. As many networking problems involve exploring the space of possible route sets to find an optimal one, the impact of the routing model on the structure of this "potential solutions space" is twofold. First, by restricting the size of the solution space, constraining routing models can forbid the most optimal solutions (see Corollary 1). Second, by changing the nature of the solution space (e.g., its size), coherence also impacts the time complexity of algorithms, as well as the cost or performance of the solution. The next section provides examples that instantiate these differences on specific algorithmic problems.

IV. Implications for Fundamental Case Studies

So far we have shown how the model can affect the diversity and complexity of routing. Depending on the specific application, the diversity in turn affects the runtime and quality/cost of the solutions of the corresponding optimization problems. In this section, we describe two case studies demonstrating these impacts of the routing model.

Network Monitoring. When deploying a set of monitoring equipment on a subset of all nodes to observe the status of links, one possible optimization objective might be to use the minimum number of equipment (i.e., minimize deployment *cost*). Finding the right nodes for a deployment is NP-hard in many settings, while for some assumptions efficient exact or approximation algorithms exist [7], [34] for two versions of the problem with one or two types of monitoring equipment.

Note, that this monitoring problem cannot be addressed with tree routing, as not all links of a non-tree graph are used in this case and hence not all links can be monitored. Other routing models can be applied, with varying complexity and cost, provided each link is used in at least one route. In this context, the solution quality refers to the amount of equipment to be deployed: the lower the better.

Traffic Engineering. Consider a graph with capacitated links, i.e., each link has a maximum amount of traffic it can carry. We define congestion to be the ratio between the number of flows using an edge and its capacity. Given a set of requests (flows from v_i to v_j, for $v_i, v_j \in V$), the traffic engineering problem comes in different flavors: assigning routes to each flow, such that either (1) the maxmimum congestion is minimized or that (2) the routes are as short as possible and do not violate the capacity constraints. Both are multicommodity flow problems [15]. The routing model to be used restricts solutions, i.e., the model is expressed as additional constraints in the multicommodity flow problem formulation. In this context, the notion of quality refers to the maximum congestion on a link resp. the length of a capacity-respecting path: lower is better.

A. Runtime

We first discuss the influence on runtime.

Consistency Influences Runtime. One simple example showing that consistency influences complexity regards the traffic engineering of a *single* flow: in a directed network, it is easy and fast to compute a shortest capacity-respecting flow between a given source s and a destination t, e.g., by using Dijkstra's algorithm. However, computing a shortest capacity-respecting route from s to t that fulfills the policy that traffic must go through a single and given waypoint w, is NP-hard [2].

Coherence Influences Runtime. One of the most important dimensions in this problem concerns the decision whether packets of the same flow always need to take the same route (unsplittable flows) or if they can choose among several routes. The general unsplittable version of the load minimization problem (i.e., ! model) is NP-complete while optimal routes for arbitrarily splittable flows ($*$ model) can be found in polynomial time [15]. The requirement of unsplittable flows is often imposed to increase the traceability of end-to-end traffic flows and to prevent package reordering and other unwanted effects of multipath routing in practice. For a variant of the traffic engineering problem, where instead of the concrete traffic matrix upper bounds on the weight of flows from and to nodes are given, a polynomial time algorithm computes an optimal tree routing scheme coinciding with the best possible simple-path confluent routing [18], [21].

We can show similar results when considering the monitoring problem for a graph where the routes are given. We rely on the consistency assumption that the routes use symmetric shortest paths below, regardless of the coherence model applied. For this setting, we discuss how coherence can influence the complexity of problems (an asymmetric version of this problem with two different types of equipment has been studied in [34]). It turns out that finding a monitoring deployment is NP-hard for the *any*, *confluent* and *contained* routing model on general graphs.

Theorem 3. *Symmetric network monitoring is NP-complete for routing models any* !*, contained* $\subseteq$*, and confluent* $>$*.* [*Proof in Appendix*]

For some restricted graph classes, differences in the complexity can be observed. In particular, a polynomial time algorithm can find an optimal assignment for cactus graphs for many routing policies, e.g., for confluent and contained routing (coinciding in this scenario, due to the symmetry), while it is NP-hard even in these graphs under the ! routing model.

Theorem 4. *Symmetric network monitoring is NP-complete under the* !*-routing model for cactus graphs.* [*Proof in Appendix*]

Theorem 5. *A polynomial algorithm exists solving the symmetric network monitoring problem under the contained* $\subseteq$*-routing model for cactus graphs.* [*Proof in Appendix*]

Observe that in this case study, the routes are assumed to be given (or chosen by the adversary). In other words, solving the problem only consists in finding a deployment and not in finding a good set of routes as well. The combined routing and monitoring problem is still NP-hard on general graphs, using the same reduction as described above. However, the cactus graph reduction cannot be used for ! routing. In this scenario, only three pieces of monitoring equipment would be necessary, one at each end and one on a node on the upper part of a cycle. The routes between them would then be chosen such that the route on the lower part of the cycles would be

used between the two end nodes and the upper part of the cycles would be used for communication to the third node with equipment. Thus the additional degree of freedom makes the problem easier to solve for these graphs.

B. Quality

Next we study the quality of the optimizations.

Consistency Influences Quality. To study how consistency influences the admissible solution quality, we compare load minimization for different routing models. There exist examples where the minimum congestion achievable with shortest path routing for a given set of commodities exceeds the congestion achievable with other routing models by large factors.

For simple-path routing, Bley [6] shows that the load obtained with shortest path routing can be up to $\Omega(|V|^2)$ times larger than the minimum congestion achievable without this restriction. Furthermore, it is also a factor of $\Omega(|V|)$ larger than the congestion of an optimal confluent routing.

Coherence Influences Quality. For the network monitoring example, we can observe on very simple graphs that the solution quality for different coherence models varies significantly, e.g., for the graph in Figure 4 due to the route diversity. For the model !, we can monitor at most x^2 different routes with shortest path monitoring pairs composed out of x nodes with monitoring equipment. If we can select the routes (i.e., when they are not chosen in an adversarial manner), we thus need at least$\lceil\sqrt{k}/2\rceil$ monitoring equipment for this scenario, deploying half of them on the source nodes on the left and the other half on the destination nodes on the right.

For the routing models confluent $>$ and contained $\subseteq$ the minimum number of equipment we need is linear in k, even when selecting routes is possible.

For traffic engineering, Lorenz et al. [31] show that finding a minimum congestion confluent $>$ route set for simple paths is NP-hard. They also show that the minimum congestion may be factor $\Omega(|V|)$ higher for confluent routing than for the any ! routing model. An intuitive explanation for this is the observation that Valiant's trick [40] cannot be used in confluent routing schemes. Furthermore it is easy to see that a traffic engineering solution that is restricted to a tree can lead to a solution that is a factor of $\Omega(n)$ worse than a confluent and contained solution, e.g. in a clique with uniform capacities and uniform all-to-all traffic demands.

For some variants of the traffic engineering problem with simple-path routing, parts of Corollary 1 have been shown in [25] namely $\mathit{qual}(R^T_\Pi(G),G) \leq \mathit{qual}(R^>_\Pi(G),G) \leq \mathit{qual}(R^!_\Pi(G),G) \leq \mathit{qual}(R^\star_{G,Q},G)$. For these problems, there also exists a polynomial time approximation algorithm that computes an optimal tree routing which coincides with the best possible confluent routing [18], [21]. The approximation bound of this solution is at most $2(1-1/n)$ times best possible fractional $\star$ solution and this also holds for !. Formally, $\mathit{qual}(R^T_\Pi(G),G) = \mathit{qual}(R^>_\Pi(G),G) \leq 2(1-1/n)\mathit{qual}(R^\star_{G,Q},G) \leq 2(1-1/n)\mathit{qual}(R^!_\Pi(G),G)$.

V. Example: OSPF

Most Internet routing protocols like OSPF and IS-IS conform to the shortest path consistency model, where each link is assigned a weight to represent its cost or length. Furthermore, unsplittable OSPF is confluent and depending on the link weight assignment and/or the selection of one of multiple shortest paths to a destination, OSPF also adheres to the contained routing model. Weights for a contained scheme can be determined efficiently if they exist.

Theorem 6. *(i) Unsplittable recursion-conform OSPF is contained. (ii) Given a contained scheme for simple-path routing, we can find OSPF weights in polynomial time if they exist. (iii) There are contained schemes for simple-path routing for which no OSPF weights exist.*

Proof. (i) Unique shortest path for all source-destination pairs imply that between two nodes v, w there is exactly one possible route that can be followed according to OSPF. OSPF then guarantees that at every node packets destined to node v will take the same outgoing link, i.e., it is source invariant and hence confluent. Hence, the condition for contained routing is satisfied and as a consequence OSPF is contained. This also holds if the choice of shortest paths is recursion-conform. (ii) Assigning weights to links in a set of routes such that unsplittable OSPF routing can be applied with unique shortest paths is known as the *inverse unique shortest paths problem*. Using linear programming techniques this problem can be solved in polynomial time [4]. With this approach, the ratio between the smallest and highest weight is bounded by the minimum of the number of nodes divided by two and the number of routes in the routing scheme. Any walk including a cycle obviously cannot be described using weights as it is not a shortest path. (iii) Consider the set $S = \{(5,6),(2,3,5),(1,4,5),(4,2,6),(3,1,6)\}$. Since no two paths share two nodes, S constitutes a contained set. However it does not satisfy cyclic compatibility and can hence not be implemented with OSPF [4]. □

Table II summarizes this and some further examples. For instance, in SDN, without any further constraints and using the most general matching scheme (e.g., matching also IP source addresses or application-specific ports), arbitrary coherence and consistency properties can be implemented. In geographic routing, a typical objective is to minimize the stretch, and the routes are contained. Routes in BGP are valley-free, and likely destination-based (although further specifications may be required).

VI. Empirical Model Diversity Analysis

In this section, we take a closer look at the routing model diversity available in various contemporary network topologies. More precisely, we here focus on shortest path consistent routing, and evaluate the number of *complete* route sets that are !-coherent and $>$-coherent and contain a route for each node pair, denoted by $\kappa^!$ and $\kappa^>$.

Name	Coherence	Consistency
SDN	arbitrary	arbitrary
Unsp. OSPF	confluent	shortest path
Unsp. recursion-conform OSPF	contained	shortest path
Babel	confluent	shortest path
Lightning	confluent	shortest path
Geographic routing	contained	stretch
BGP	confluent	valley-free

TABLE II
EXAMPLES OF ROUTING MODELS WITH COHERENCE AND CONSISTENCY PROPERTIES.

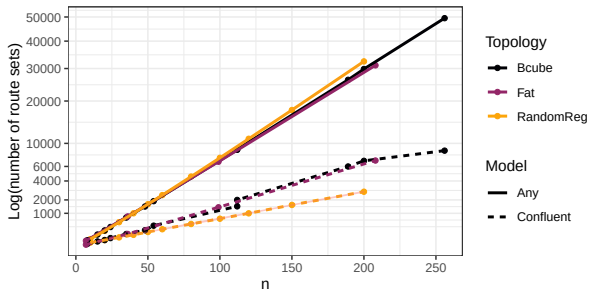

Fig. 5. Number of complete route sets adhering to *shortest paths confluent* and *any* routing models as a function of the topology size, for Fat Trees, BCubes and 3-regular random networks. The ordinate scale is square rooted.

As we have seen in the previous section, the choice of a coherence model modifies the solution space of related applications, which in turn impacts both runtime and quality of the obtained solutions. We here empirically compare the size of those solution spaces, namely $\kappa^!$ and $\kappa^>$.

To conduct this numerical evaluation, we focus on 3 types of topologies. First, (LAN) datacenter topologies: we generate all Bcubes and Fat trees of size $n \leq 256$ [1]. Second, (WAN) *internet topology zoo* networks[2]: we consider all zoo topologies of size $n \leq 256$. Finally, d-regular random graphs are used to provide a synthetic baseline, with $d = 3$ (average degree of zoo topologies).

Figure 5 plots the diversity of both confluent $>$ and any $!$ routing models for datacenter and zoo topologies. A first observation is the tremendous growth of both values. E.g., for $Fat(4)$ topologies ($n = 99$ nodes), the number of confluent route sets is in the order of 10^{1408} and the number of any route sets in the order of 10^{6740}. This observation holds for all topologies. The straight lines on the square-rooted scale suggest that the number of complete shortest path models is $\approx 10^{n^2}$.

A second observation concerns the fraction of (any $!$) route sets that are confluent $>$, formally the ratio $\kappa^!/\kappa^>$. For instance, on $Fat(4)$, this ratio is 10^{5332} which means that by sampling uniformly any-models, the probability of finding a confluent one is extremely low. This observation is confirmed also for the zoo datasets in Figure 6, which directly plots this ratio. This gap illustrates the considerable scale difference between the *any route* set space, and the *confluent route* set space. As

[1]We use FNSS for topology generation [36].

[2]The Internet Topology Zoo http://www.topology-zoo.org/dataset.html.

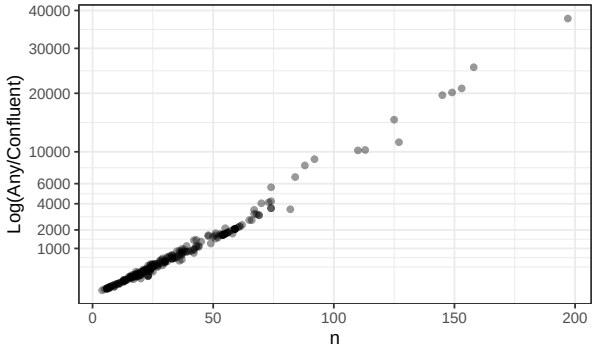

Fig. 6. The ratio between the number of *any* route sets and *shortest path confluent* route sets on zoo topologies. The ordinate scale is square rooted.

illustrated before, when optimizing routes for an application goal, this gap impacts both the quality of the optimal route set, and the exploration strategy of the solution space (which in turn impacts the runtime).

So far, we estimated the quantity of route sets for different coherence models. This quantity potentially impacts both the optimum and the runtime of any route optimization process applied to a target topology.

To further understand the impact of coherence models, we need to estimate the quality of routes sets beyond their mere quantity. The notion of quality is however application dependent. To circumvent this difficulty, we generate for each link e of a target topology a $cost(e)$ value, that represents an application's preference to route through this edge e, and we restrict ourselves to shortest path consistency..

Concretely, we study three cost functions that each have an average value of $1/2$ to allow their comparison:

- `Constant` considers all links equal (at a cost of $1/2$), hence only expressing preferences towards shortest path consistency: $\forall e \in E, cost(e) = 1/2$.
- `Uniform` has a preference drawn uniformly at random $\forall e \in E, cost(e) = \texttt{random(0, 1)}$.
- `Source-dependent` has different preferences for each edge depending on the packet source: $\forall e \in E, s \in V, cost_s(e) = \texttt{random(0, 1)}$.

We focus on zoo topologies for this experiment. Given a target topology, we assign edge costs according to the above cost functions to all edges and pick random destinations. We then explore the route sets available from all sources to the selected destinations for the coherence models *any* $!$, *confluent* $>$ and *tree* T, seeking a coherent route set that minimises the total cost, defined as the sum of costs of the routes taken by each source towards the destination. More precisely, for this experiment we define $qual(R^{\Sigma}_{\Pi}(G), G) = \sum_{R[s,d] \in R^{\Sigma}_{\Pi}(G)} \sum_{e \in R[s,d]} cost_s(e)$.

For the any routing model, minimizing the total cost is straight-forward, as any shortest path from a source to a destination under the cost function can be chosen. For the confluent routing model, shortest path consistency with cost functions `Constant` and `Uniform` lead to a regular routing algebra and thus shortest path spanning trees for the given

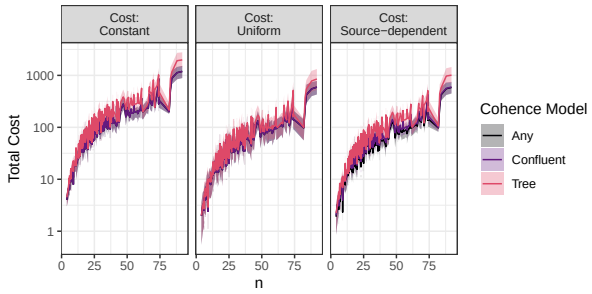

Fig. 7. Quality (lower is better) of the obtained route sets as a function of the number of nodes (n) in the topology zoo graph, for different routing models. The shaded area corresponds to the $[5, 95]$ percentile interval.

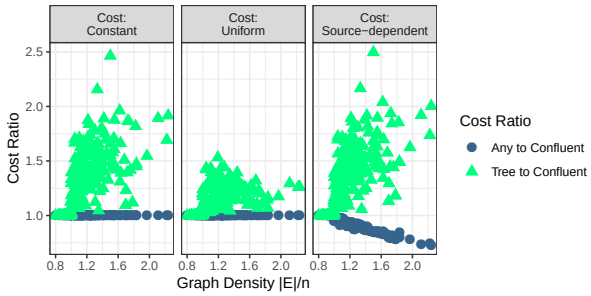

Fig. 8. Relative quality of the tree and any coherence compared to confluent coherence, as a function of the graph density (number of edges per node: m/n) for the topology zoo graphs. One topology ($n = 9, m = 36$) omitted for readability.

destinations can be constructed easily. However, for source-dependent edge costs, all possible spanning trees need to be considered to find the minimum. Since the sets of spanning trees for the topology zoo graphs are too large, we use a heuristic which sums up the source-dependent costs per edge and then constructs confluent shortest paths based on these sums for each destination. Due to the computational complexity to find the best spanning tree, we use one arbitrary minimum spanning tree for all destinations for the tree routing model. Note that the chosen tree might not lead to the minimum costs possible, yet the associated costs can be seen as an upper bound of the true minimum cost.

We repeat this process for all zoo graphs, for 10 randomly sampled destinations, 30 times, and obtain the results presented below. Since the broad majority of topology zoo graphs have less than 100 nodes, our plots only depict those for readability; however, the results also hold for the other graphs.

As one would expect, Figure 7 shows that the absolute value of the total cost increases with the number of nodes and the total costs for the tree coherence model is highest for all cost functions. Moreover, a close inspection confirms that tree routing is always above confluent and any routings. The any routing set improves over confluent routing only in the source-dependent cost function case: this illustrates Observation 1 where uniform and constant costs (together with the standard addition operation) imply a regular routing algebra.

To study the differences between the models in more detail, we look at the relative costs in Figure 8. Here we see that for the constant and uniform cost function the total cost of the any and confluent routing set coincide while the cost with an arbitrary spanning tree are up to 2.5 times higher. Interestingly, we observe that a higher topology density results in a higher cost ratio: intuitively, more edges provide paths which in turn provide more opportunities for a less coherent route set to optimise over a more coherent route set (like any routes over confluent routes in the source-dependent cost case, or confluent routes over tree routes in all cost cases). In the plot for the source-dependent cost function we observe that the heuristic used to compute a confluent route set does not perform much worse than the optimal routing set in the any routing model. Moreover we see a linear dependency on the density with a much lower deviation than for tree routing with a maximum of 25%.

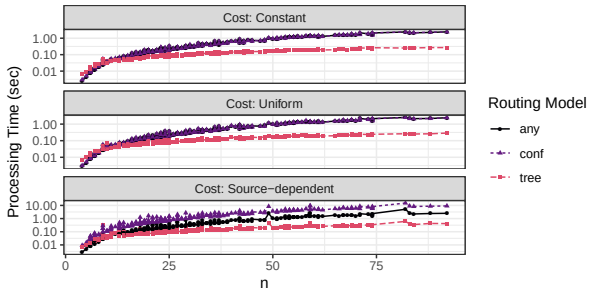

Fig. 9. The time to compute the route sets as a function of the number of nodes (n) in the topology zoo graph.

In Figure 9 we plot the time to compute the route sets. While computing a minimum spanning tree never takes on average $121ms$, finding the optimal route set take $540ms$. The heuristic for confluent routing under source-dependent edge costs take $2.02s$ on average over all topology zoo graphs. While the confluent routes result in higher costs, they have the advantage that the corresponding routing tables are a factor of n smaller than for the any coherence model. In practice this is very important and the overhead in the routing cost is typically more acceptable.

VII. Related Work

The need for a more direct control over traffic routes became evident the latest in the 1990s [16], leading to novel networking technologies such as MPLS (e.g., [39]). Since then, the goal of a more efficient traffic engineering remained an important innovation motor in networking, also motivating recent technologies such as software-defined networking [24], [26] and segment routing [5], among other [9], [29], [30].

Interestingly, however, there does not exist much analytical work on the impact of the routing model on aspects beyond congestion and path diversity. A notable exception is the work by Erlebach et al. [13], [14] who study the computational complexity of routing, under consistency constraints, valid s-t-routes and s-t-cuts, in the valley-free model. Kloeti et al. [28] proposed a generic method applicable to arbitrary graphs for

policies which can be formulated as regular expressions (e.g., valley-free, (negative) waypoint routing multipath TCP). Most closely related to our work are [8] and [34]. Chekuri [8] considers the problem of choosing routes for certain demand classes to minimize the resulting congestion and further compares different coherence classes in relation to each other. Chekuri also proposes a hierarchy which we extend in this paper. His *any* (!) property is called *Single Path Routing (SPR)*, his confluent ($>$) property is called *Terminal Tree Routing (TTR)* and *tree routing* (T) has the same name. We extend this hierarchy with *contained* ($\subseteq$), which lies between the confluent and tree model. To avoid confusion with routing on trees, we use the term confluent in this context, instead of terminal tree routing. We also generalize the routing models by allowing to visit nodes multiple times. Pignolet et al. [34] also introduce and compare routing models, however their work is restricted to coherence and does not account for consistency aspects; their results revolve around a specific case study.

Bibliographic note. A preliminary version of this paper has been presented at the IFIP Networking 2020 conference [33]. This journal version includes all technical details (including proofs), an extended experimental section and discussion of real-world examples and an updated related work section.

VIII. Conclusion and Future Work

This paper was motivated by the observation that the routing model, due to its constraints, can significantly influence the complexity and efficiency at which networks can be operated. We introduced a taxonomy of routing models accordingly and studied different implications. This also led us to question reasoning about network optimization problems in terms of the underlying physical topology only: this graph-centric view ignores the influence of the routing model. Sometimes a more restrictive coherence model does not affect route diversity much while introducing opportunities to speed up algorithms. We also found that tie-breaking among a set of consistent rules can have an impact on the efficiency of some protocols.

We hope that our perspective on consistent and coherent routing models can help the networking community identify novel optimization opportunities and reason about algorithms. In this regard, our paper also opens several interesting avenues for future research. In particular, we have so far focused on unicast routing and it will be interesting to extend our work to multicast routing. It also remains to explore the impact of the routing model on other performance criteria, e.g., memory/space complexity and on alternative applications, beyond traffic engineering and monitoring.

Acknowledgments. Research supported by the ERC Consolidator project AdjustNet (grant no. 864228).

References

[1] A. AbdelSalam, F. Clad, C. Filsfils, S. Salsano, G. Siracusano, and L. Veltri. Implementation of virtual network function chaining through segment routing in a linux-based nfv infrastructure. In *2017 IEEE Conference on Network Softwarization (NetSoft)*, pages 1–5. IEEE, 2017.

[2] Amiri et al. Charting the algorithmic complexity of waypoint routing. In *Proc. ACM SIGCOMM Computer Communication Review (CCR)*, 2018.

[3] Anderson et al. Netkat: Semantic foundations for networks. In *ACM SIGPLAN Notices*, 2014.

[4] W. Ben-Ameur and E. Gourdin. Internet routing and related topology issues. *SIAM Journal on Discrete Mathematics*, 17(1):18–49, 2003.

[5] R. Bhatia, F. Hao, M. Kodialam, and T. Lakshman. Optimized network traffic engineering using segment routing. In *Proc. IEEE INFOCOM*, 2015.

[6] A. Bley. Approximability of unsplittable shortest path routing problems. *Networks*, 54(1):23–46, 2009.

[7] P. Boothe, Z. Dvorak, A. Farley, and A. Proskurowski. Graph covering via shortest paths. In *Congressus Numerantium*, 2007.

[8] C. Chekuri. Routing and Network Design with Robustness to Changing or Uncertain Traffic Demands. *SIGACT NEWS*, page 2007, 2007.

[9] M. Chiesa, G. Rétvári, and M. Schapira. Lying your way to better traffic engineering. In *Proc. ACM CONEXT*, 2016.

[10] J. Chroboczek et al. The babel routing protocol. Technical report, RFC 6126, April, 2011.

[11] M. Dotan, Y.-A. Pignolet, S. Schmid, S. Tochner, and A. Zohar. Survey on blockchain networking: Context, state-of-the-art, challenges. In *Proc. ACM Computing Surveys (CSUR)*, 2021.

[12] J. Drake. Rift working group t. przygienda, ed. internet-draft juniper networks intended status: Standards track a. sharma expires: September 2, 2018 comcast a. atlas. 2018.

[13] T. Erlebach, A. Hall, A. Panconesi, and D. Vukadinović. Cuts and disjoint paths in the valley-free path model of internet bgp routing. In *Proc. CAAN*, 2005.

[14] T. Erlebach, L. S. Moonen, F. C. R. Spieksma, and D. Vukadinović. Connectivity measures for internet topologies on the level of autonomous systems. *Oper. Res.*, 57(4), 2009.

[15] S. Even, A. Itai, and A. Shamir. On the complexity of time table and multi-commodity flow problems. In *Proc. IEEE FOCS*, 1975.

[16] N. Feamster, J. Rexford, and E. Zegura. The road to sdn: an intellectual history of programmable networks. *ACM SIGCOMM Computer Communication Review (CCR)*, 2014.

[17] C. Filsfils, N. K. Nainar, C. Pignataro, J. C. Cardona, and P. Francois. The segment routing architecture. In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2015.

[18] J. A. Fingerhut, S. Suri, and J. S. Turner. Designing least-cost nonblocking broadband networks. *Journal of Algorithms*, 24(2):287–309, 1997.

[19] L. Gao and J. Rexford. Stable internet routing without global coordination. *IEEE/ACM Transactions on networking*, 9(6):681–692, 2001.

[20] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[21] A. Gupta, J. Kleinberg, A. Kumar, R. Rastogi, and B. Yener. Provisioning a virtual private network: a network design problem for multicommodity flow. In *Proc. ACM STOC*, 2001.

[22] A. J. Gurney and T. G. Griffin. Lexicographic products in metarouting. In *2007 IEEE International Conference on Network Protocols*, pages 113–122. IEEE, 2007.

[23] J. Halpern, C. Pignataro, et al. Service function chaining (sfc) architecture. In *RFC 7665*. 2015.

[24] C.-Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, and R. Wattenhofer. Achieving high utilization with software-driven wan. *SIGCOMM Comput. Commun. Rev.*, 43(4):15–26, Aug. 2013.

[25] C. A. Hurkens, J. C. M. Keijsper, and L. Stougie. Virtual private network design: A proof of the tree routing conjecture on ring networks. *SIAM Journal on Discrete Mathematics*, 21(2):482–503, 2007.

[26] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, and A. Vahdat. B4: Experience with a Globally-Deployed Software Defined WAN. In *SIGCOMM*, 2013.

[27] J. S. Jensen, T. B. Krogh, J. S. Madsen, S. Schmid, J. Srba, and M. T. Thorgersen. P-rex: Fast verification of mpls networks with multiple link failures. In *14th ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2018.

[28] R. Kloti, V. Kotronis, B. Ager, and X. Dimitropoulos. Policy-compliant path diversity and bisection bandwidth. In *Proc. IEEE INFOCOM*, 2015.

[29] Kotronis et al. Stitching inter-domain paths over ixps. In *Proc. ACM SOSR*, 2016.

[30] P. Kumar, Y. Yuan, C. Yu, N. Foster, R. Kleinberg, and R. Soulé. Kulfi: Robust traffic engineering using semi-oblivious routing. *arXiv*, 2016.

[31] D. Lorenz, A. Orda, D. Raz, and Y. Shavitt. How good can IP routing be?, 2001. Technical Report 2001-17, DIMACS.

[32] K. Patel, A. Lindem, and W. Henderickx. Shortest path routing extensions for bgp protocol. *Network Working Group Draft*, 2018.

[33] Y.-A. Pignolet, S. Schmid, and G. Tredan. Implications of routing coherence and consistency on network optimization. In *Proc. IFIP Networking*, 2020.
[34] Pignolet et al. Tomographic node placement strategies and the impact of the routing model. In *Proc. ACM SIGMETRICS*, 2018.
[35] G. Rétvári, A. Gulyás, Z. Heszberger, M. Csernai, and J. J. Bíró. Compact policy routing. *Distributed computing*, 26(5-6):309–320, 2013.
[36] L. Saino, C. Cocora, and G. Pavlou. A toolchain for simplifying network simulation setup. In *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques*, SIMUTOOLS '13, ICST, Brussels, Belgium, Belgium, 2013. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).
[37] J. L. Sobrinho. Network routing with path vector protocols: Theory and applications. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 49–60. ACM, 2003.
[38] R. Soulé, S. Basu, R. Kleinberg, E. G. Sirer, and N. Foster. Managing the network with merlin. In *Proc. ACM HotNets*, 2013.
[39] S. Suri, M. Waldvogel, and P. R. Warkhede. Profile-based routing: A new framework for mpls traffic engineering. In *International Workshop on Quality of Future Internet Services*, pages 138–157. Springer, 2001.
[40] L. G. Valiant. A scheme for fast parallel communication. *SIAM Journal on Computing*, 11(2):350–361, 1982.
[41] T. Winter, P. Thubert, A. Brandt, J. W. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J.-P. Vasseur, R. K. Alexander, et al. Rpl: Ipv6 routing protocol for low-power and lossy networks. *rfc*, 6550:1–157, 2012.

APPENDIX

Deferred theorems and proofs on the hardness of symmetric network monitoring. The analysis follows the arguments [34], which however considers the asymmetric case, while we prove results for the symmetric version in this article.

Theorem 7. *Symmetric network monitoring is NP-complete for routing models any* !*, contained* $\subseteq$*, and confluent* $>$*.*

Proof. The proof relies on a reduction from the NP-hard node cover problem [20]. The node cover problem asks whether for a given graph $G(V, E)$ and a threshold k, there exists a subset $S \subseteq V(G)$ such that $\forall (u, v) \in E(G),\ \{u, v\} \cap S \neq \emptyset$ and $|S| \leq k$. Note that this problem is slightly different than the monitoring problem. E.g., to monitor a chain of links, only the first and the last node require monitoring equipment, while every second node needs to be in a valid node cover.

Given such a node cover problem instance, one can build a graph $G'(V', E')$ such that a solution of the monitoring problem on G' can be used to derive a solution to the node cover problem. We can show that in G', a monitoring solution with $n + k$ monitoring equipment exists if and only if a node cover of G with k nodes exists.

In short, G' consists of two copies of the original graph G, called H and H', with additional nodes inserted on the edges between the original node. In copy H the original edges are replaced by 3-hop paths, in H' with four hop paths. The original nodes of the two copies are connect with edges between them. H' is extended with 3-hop paths between original nodes which are not connected in G. Moreover, one more node is added to each original nodes in H', as well as an edge connecting it. This last addition of nodes of degree one ensures that they obtain monitoring equipment and thus this copy of the graph is fully monitored. For the other copy and the edges connecting the two copies, we can show that choosing a node cover set of $G(V, E)$ and assigning equipment to it, guarantees that all edges are monitored. On the other hand, any set of nodes that does not cover all edges also fails to monitor all edges of E'. As a consequence, given a valid monitoring solution of $G'(V', E')$ with $n + k$ nodes the assigned monitoring equipment can be used to construct a node cover for $G(V, E)$ with k nodes. □

Theorem 8. *A polynomial algorithm exists solving the symmetric network monitoring problem under the confluent* $>$*-routing model for cactus graphs.*

Proof. The polynomial algorithm for confluent routing is based on the observation that valid solutions for graphs with articulation points (nodes which disconnect the graph when removed) can be stitched together for the confluent model from valid solutions of the subgraphs on both sides of the articulation points. This relies on the fact that equipment in one subgraph can also be used for monitoring the other subgraph. As a consequence, we can build a valid solution for one side of the articulation point by putting "virtual" monitoring equipment on the articulation point and deploying the remaining necessary equipment on each side separately. Afterwards we can remove the "virtual" equipment and still have a valid solution.

This approach cannot work for !, since in this routing model the source node can influence the path taken. This can be observed in a ring of even length n with two additional nodes u, v connected to a node w. These two nodes u, v need to be assigned monitoring equipment due to the fact that they are of degree 1. Put another piece of monitoring equipment on w', the node at distance $n/2$ from w on this ring. In the ! routing

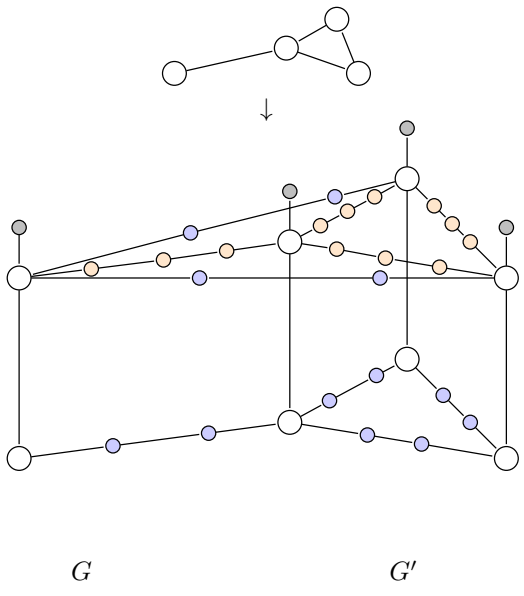

Fig. 10. Illustration of construction of $G'(V', E')$ given a graph $G(V, E)$. In the proof we show that a k node cover of G is equivalent to a $(n + k)$ monitoring solution in G'.

model it is possible that the whole graph is already monitored like this, as the routes from to u, v to w' could take the paths on either side of the ring. In confluent $>$ routing on the other hand, only one part of the ring is monitored and another piece of monitoring equipment is definitely necessary.

This allows us to consider each cycle of the cactus once and determine the necessary and sufficient number of equipment for it. In a first step we contract the cactus graph to a tree T such that every cycle is represented by a node. Then we put monitoring equipment on the leafs of T, one in case the leaf is a single node in the original graph and two in case the leaf represents a cycle, at the opposite side of the articulation point. We observe that this guarantees that all edges in the leaf cycles are monitored, so we can consider these nodes in T to be monitored. We now process each parent cycle of the monitored nodes in T individually, in a recursive fashion: we place a "virtual" monitoring equipment on each of the articulation points. If not all edges are monitored by the virtual equipment already we add "real" equipment to this cycle as necessary and proceed to the next node in T that needs additional equipment.

This algorithm works in polynomial time as there are at most $n/3$ cycles, each containing at most n nodes and determining if and and where additional "real" equipment is needed can be decided in polynomial time of the cycle size. The produced deployment monitors all edges due to the fact that stitching together preserves correctness. Optimality can be shown by contradiction using a recursive argument. □

Theorem 9. *Symmetric network monitoring is NP-complete under the* !*-routing model for cactus graphs.*

Proof. To show hardness, we devise a reduction from the NP-hard problem set cover [20]. The input of the set cover problem is a set of n elements and m subsets $S_1, \ldots, S_m$ containing some of the elements, and an integer k. The output should be $true$ iff there is a selection of k of the subsets such that their union includes all n elements.

Given a set cover problem instance, we construct a cactus graph $G(V, E)$ and a shortest path routing on it such that a solution of the monitoring problem on G can be used to derive a solution to the set cover problem.

We take the "sausage" graph depicted as graph B in Figure 4 with n even length cycles and a line of length $m+1$ on the right end. Each of the n cycles stands for an element. A shortest routing path from node v_l at distance l from the right end to the left-most node encodes the subset S_l: for each element that is in the set, the path on the top of its corresponding cycle is taken, whereas the lower path is used otherwise. For all other shortest path pairs the lower path is taken whenever feasible.

Assigning monitoring equipment to the nodes on both ends all edges of the lower paths are monitored. To monitor the top edges, either a piece of monitoring equipment has to be placed on the corresponding top node, or on one of the nodes at distance 1 to l from the right. It is easy to see that iff there is monitoring solution with k pieces of monitoring equipment, there is also a solution to the set cover problem. □