A curated dataset of real and synthetic quasi-planar Internet backbone network topologies

Tamás-Zsolt Képes[1], Noémi Gaskó[1], János Tapolcai[2], and Balázs Vass[1,2]

[1]Faculty of Mathematics and Computer Science, Babeș-Bolyai University, Cluj Napoca, Romania
[2]Department of Telecommunications and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Műegyetem rkp. 3., H-1111 Budapest, Hungary
email: {*tamas.kepes,noemi.gasko,balazs.vass*}*@ubbcluj.ro, tapolcai@tmit.bme.hu*

Abstract

Open datasets of Internet backbone topologies are essential for the reproducible evaluation of various types of routing algorithms. Thus, numerous research papers have been using available network repositories such as Topology Zoo and SNDlib. However, a lesson learned in the past years is that, in many cases, it is not enough to treat backbone networks as purely logical topologies, but they should be treated as geometric graphs instead (e.g., natural or man-made disasters fail network elements sharing the same geographic region). While TopoHub, a recent repository, makes a significant step in solving this issue, its graph representations still neglect the (almost) planar nature of backbone networks, a feature many recent routing algorithms rely on fundamentally. This paper introduces T-Garden, a curated and openly accessible collection of (nearly) planar Internet backbone topologies. The dataset consists of (i) fixed and standardized versions of selected physical-layer backbone networks sourced from available repositories, and (ii) synthetic planar topologies that scale to several thousand nodes while preserving key structural and geometric properties observed in real networks. By providing full geometric information on over 120 (nearly) planar backbones, T-Garden provides a comprehensive benchmark dataset for research on geography-aware telecommunication networks.

Background & Summary

By now, telecommunication networks, primarily served by optical backbones, have long become topmost critical infrastructures. Their near-continuous operation underpins essential services for citizens, industry, economy, and public administration. This fundamental societal importance has been acknowledged by governmental and regulatory bodies in both the United States and the European Union[1,2]. These networks have been extensively studied in the networking literature, reflecting their central role in global communication systems[3–10].

A long-standing challenge in this research area is the limited availability of openly accessible and well-documented Internet backbone topologies, stored in standardized repositories[11–13]. Detailed information on physical backbone infrastructures is often scarce, fragmented, or outdated, largely due to commercial sensitivity and security considerations on the part of network operators. As a result, the research community has relied on a small number of publicly available datasets, each of which only partially satisfies the requirements of contemporary algorithmic evaluation and benchmarking.

An important geometric property of the backbone networks is their planarity or quasi-planarity. Unfortunately, it is being at least partially overlooked by current repositories. However, it is well-known that planarity enables the efficient solvability of various problems that are otherwise hard to tackle (e.g., $\mathcal{NP}$-hard, or even inapproximable in polynomial time unless $\mathcal{P}$=$\mathcal{NP}$) in routing[14–19] and beyond[20–24]. Recognizing the utmost algorithmic importance of preserving the (near) planarity of backbone networks in their geometric graph representations, T-Garden lays solid foundations for the advent of a programmatic leverage of the combination of two exciting and seemingly distant fields: networking applications and the theory of efficient algorithms on planar graphs.

Preexisting datasets Several repositories have previously attempted to make Internet backbone and transport-network topologies available to the research community. Among these, the Topology Zoo[25] is probably the most widely used collection. A key to its success was that it consolidated a large number of

heterogeneous topology descriptions originating from earlier measurement efforts and operator disclosures. Despite its historical importance and continued popularity, the dataset exhibits a number of practical limitations, including occasional absence of geometric information on network nodes, or heterogeneous graph semantics. Another issue is the absence or incompleteness of the metadata, which often necessitates substantial preprocessing before the networks can be used in simulation or algorithmic evaluation.

Another established source of openly available backbone-related topologies is SNDlib[26,27], which was primarily developed to 'provide data about realistic telecommunication networks (including traffic) available to the research community'. While SNDlib provides well-defined instances and additional information such as traffic demands and capacities, its scope is broader than physical-layer fiber backbones, and the representation of network elements is not fully aligned with the requirements of geometry-aware analyses. Consequently, only a subset of its networks can be directly employed for studies focusing on the physically embedded backbone structure, often after additional adaptation.

More recently, TopoHub[28] has aimed to provide an easy-to-use collection of physical-layer Internet topologies by aggregating data from existing repositories and supplementing them with synthetic instances. However, the aggregation process selectively omits several real-world backbones and does not apply a systematic curation or standardization pipeline to the retained topologies (see e.g., upcoming Fig. 1), nor does it take into account the (almost) planarity of the backbones. Moreover, while TopoHub includes artificially generated networks, these instances exhibit structural characteristics that deviate substantially from those observed in operational backbones, reducing their relevance for a range of algorithmic and modeling tasks. Most importantly, it introduces an unrealistically and unnecessarily high number of edge crossings, which renders it unusable in case of numerous algorithms relying on the almost-planar nature of backbones[15–19].

It is important to distinguish the scope of T-Garden from that of widely used general-purpose graph and network repositories. Popular collections such as KONECT[29], the Network Data Repository[30], UCINET[31], SNAP[32], or the SuiteSparse Matrix Collection[33] provide a broad spectrum of graphs originating from diverse application domains, including social networks, economic networks, web graphs, biological interaction networks, and combinatorial benchmarks. Similarly, measurement platforms and data portals such as CAIDA[34], RIPE Atlas[35] and RIPE Routing Information Service (RIS)[36] focus on traffic measurements, routing data, or Internet-scale observations rather than explicit physical-layer backbone topology representations. In contrast to these resources, T-Garden is deliberately restricted to physical-layer Internet backbone topologies and their realistic abstractions, with an emphasis on complete geographic information, consistent semantics, and immediate usability for topology-centric analysis. As such, T-Garden is complementary to existing graph repositories rather than overlapping with them, addressing a specific and previously underserved need within the networking research community.

Aim of T-Garden The main objective of this paper is the introduction of T-Garden, a freely available quasi-planar backbone topology data source. Firstly, we aimed to provide the best of the already established main academic sources, as a means of conservation. Secondly, we generated synthetic networks to help the research community in designing and testing different algorithms. The generation methods are described in detail in Sec. Methods. Data records are described in Sec. Data Records. As a validation, we compared the generated synthetic networks with existing real networks based on several relevant metrics in Sec. Technical validation. We provide realistic networks up to several thousand nodes, which can be viewed as forecasted backbone networks, and are also handy in testing the scalability of various algorithms. Data and code availability are disclosed in Sections Data Availability and Code Availability, respectively.

Methods

In this section, we first describe the real-world networks collected from various sources (mainly from Topology Zoo and SNDlib). Then, we turn to presenting the approach we took for constructing realistic synthetic backbone networks.

Real-world networks

Topology Zoo

Topology Zoo[25] was a project that resulted in a collection of networks with topological origins. The Zoo currently hosts over 250 networks. Its website states that it is still an ongoing project. However, the

Topology Zoo appears to be no longer well-maintained, with the site being unavailable most of the time (as of 2025). We used the Internet Archive (`https://archive.org/search?query=topology+zoo`) to access earlier available versions of the site, with most networks being available to download.

All Topology Zoo networks were available in two main formats: GML and GraphML. As a starting point, we used the GML-formatted graph versions, as this format is both more human-readable and can be automatically parsed by several popular libraries, such as the NetworkX[37] Python library.

Topology Zoo classifies networks using network structure as a criterion. As our T-Garden focuses on physical networks, we manually gathered networks labeled to be *Fiber* or *Simplified Fiber* networks. On the other hand, we discarded the graphs labeled as *Logical*. This substantially reduced the number of Topology Zoo networks that were available for our use case. In the end, 34 networks were selected using this initial filtering. These networks were later modified to fit our use case, but also to fix some inconsistencies present in them, to unify their format, and to fix possible errors. Some of these modifications were major hurdles in the pursuit of standardization, as discussed in the following.

Parsing and formatting Although the GML format was universally applied to all networks on the Topology Zoo site, some inconsistencies led to the files failing to load with a base NetworkX loading process. While these errors could be circumvented, we decided to modify the networks to be more standardized and to be readable by NetworkX on the fly (for the final standard format, see also Sec. Data Records).

All networks' nodes contained both a *label* field and an *ID* field. By default, NetworkX uses the label field to identify nodes, but this leads to duplicates since multiple identical place names can coexist. This can be easily circumvented by setting the label parameter of the reader function to identify nodes by their ID field instead. From now on, this will be the default notation, and all future networks will be modified to fit this criterion.

Duplicate edges also exist in the original network files; these edges have the same starting and ending nodes, and as such, they are labeled as duplicates in the NetworkX reading process. This error message halts execution. In[25] the authors state that networks should be considered multigraphs: "...In fact, we really mean a multigraph, as multiple edges are allowed between a single pair of nodes...". This consideration is only taken into account if the *Multigraph 1* metadata is present at the beginning of the GML file. Some networks missed this information, while still being multigraphs, so going forward, all networks are considered multigraphs, and the associated metadata is always present in the corresponding section.

The nodes in any given network all possessed the internal property, which was either 1 or 0. Internal nodes, those marked by 1, all represented some form of "real" node within the network, representing points of interest such as cities or junctions in the connections. Non-internal nodes, represented by a 0, usually had very little information associated with them. They represent meta information, something outside of the structure of the network, hence the labeling, non-internal. The latter nodes were discarded. Some networks contained isolated nodes, which were also removed.

Missing coordinates Real nodes in these topology networks represent places of interest, usually cities. There is another node type, labeled as *hyperedge*, that will be discussed shortly. Focusing only on the real places, we observed that a fully detailed node contained location information, given as global coordinates (longitude and latitude). Some real nodes missed this information, making them difficult to pinpoint, both for illustration purposes and for many algorithms that rely on the geometric information of the networks.

Initially, the nodes missing location data seemed limited in number, and a manual approach was selected, but as more networks were being analyzed, the problem seemed more extensive, and an automatic approach was taken. We used the OpenStreetMap[38] API to automatically locate these nodes. In the final unified networks, all node coordinates are given, since in the serialization process, we exported these newly acquired positions.

Example: In the case of the Bestel network taken from the Topology Zoo, San Miguel was a city with missing coordinates, somewhere in Mexico. When the location of San Miguel was queried, our algorithm faced a challenge: the most relevant San Miguel was already represented in the network, and our mystery San Miguel needed to be a separate city. Unfortunately, when looking at its approximate position calculated by the relative position of its neighbors, there was no San Miguel in sight. Thus, to make the best possible use of the data inherited from the Topology Zoo, we bred the city of San Miguel de Bestel, a fictional city in Mexico with an approximate position that does not correspond to a real city, but it fits the coordinates.

Node types labeled as *hyperedge* and *internal* In the Topology Zoo networks, one major challenge was the presence of two special node types, labeled as *hyperedge* and as *internal*. Note that the *hyperedge* label on some **nodes** used in the Topology Zoo can be a bit misleading. We clarify below.

Theoretically, a hyperedge in a hypergraph is a representation of a connection between (possibly more than two) nodes. However, the Topology Zoo graph instances are simple graphs where every edge contains exactly two nodes. To be precise, in the following, we define the use of nodes labeled *hyperedge* in Topology Zoo mathematically. In each Topology Zoo network instance $G = (V, E)$, each node u labeled as *hyperedge* is a true network node that represents a hypothetical hyperedge $S \subseteq V$, in the following way: $\{u, v\} \in E$, for all $v \in S$. Put it differently, if there would be a cluster of interconnected nodes S, Topology Zoo introduces an extra node u and connects it to each node in S via a simple network edge. Examples of hyperedge nodes are the ones denoted by yellow of Fig. 1c, discussed in more detail in the upcoming *Example* subsection.

In Topology Zoo, hyperedges mainly served the purpose of representing non-marked connections in the network, by introducing a new node and splitting the connecting edges into smaller edges that route through this new node, which in turn gained the "hyperedge" property, as described in [25].

While the solution is ingenious, it results in new unanswered questions relating to the nature of these nodes and their correct representation in the dataset. In Topology Zoo, nodes labeled as *hyperedge* often miss crucial information, such as positional data or other properties. Note that for enabling the drawing the network topologies, Topology Zoo had to calculate positions for the *hyperedge* type nodes; still, these coordinates are missing from its GML/GraphML files.

In the original dataset, hyperedge nodes were not assigned explicit spatial positions, although they were connected to multiple real nodes or to other hyperedge nodes. Consequently, the coordinates of the already placed neighboring nodes can be used to approximate the location of a given hyperedge node. A reasonable estimation of the position of these fictive nodes is obtained by computing the average of the coordinates of their connected nodes.

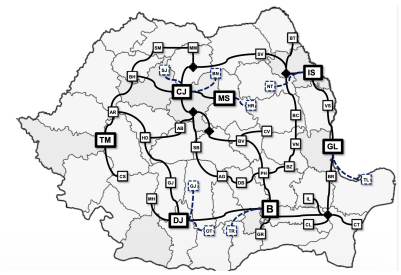
However, this approach has a limitation: some hyperedge nodes are connected to other hyperedge nodes that initially lack spatial coordinates. In such cases, particularly when multiple hyperedge nodes are adjacent, the resulting approximations may introduce positional inaccuracies, leading to mislocated fictive nodes. A second round of the position calculation step is introduced that slightly repositions already placed hyperedges, in order to more correctly approximate their "true" position.

A secondary question is the representation of hyperedge-type nodes in our final data. We propose a compromise that would keep the unique nature of these type of nodes, while still giving our network a uniform structure, including a defined geometric location for each *hyperedge* node. As such, nodes marked with the hyperedge property retain this detail in the final network format, but they also gain the positional properties, that were calculated by us. While the position is approximated, it is still better than no position at all, meaning that if someone would intend to use one of the networks where *hyperedge* nodes are present, they could ignore the hyperedge property and still effortlessly calculate network data using a full set of node properties, or in the inverse scenario, someone could use the hyperedge property to its fullest, intended extent, using it to have a more detailed or conditional calculation.

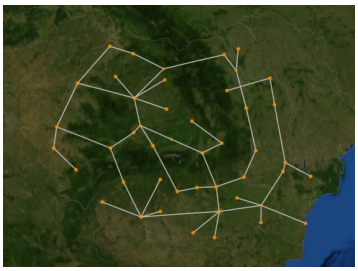
A tangential problem to the hyperedge dilemma was nodes that were marked as "internal" nodes but lacked many other details of true internal nodes, such as a label or coordinates. The missing properties meant that these nodes should be disregarded, but the internal property meant that they are important for network structure. For example, most of them were connected to multiple real nodes. As a compromise solution, these nodes were marked as hyperedge-type nodes in our final dataset and were given the same approximation treatment as the original hyperedge nodes.

Example: To illustrate the need for a careful curation process as described above, we chose to display different versions of Roedunet in Fig. 1. An advantage of this network is that its origins can be easily traced back beyond Topology Zoo, as it is described, e.g., in [39]. More concretely, Fig. 1a depicts Roedunet as planned in 2008. As can be seen in Fig. 1b, Topology Zoo did a fine job in visually re-creating the network. However, their data contains hyperedge nodes (although, based on Fig. 1a, the use of hyperedges is not fully justified) without coordinates and implicitly encodes multiple edges, which makes it challenging to preprocess the file if someone wanted to use the network for some simulations. Fig. 1c shows the representation of Roedunet based on our data. In our visualizations in T-Garden, one may locate the nodes that were hyperedges in Topology Zoo, encoded with yellow (which differs from the baseline red). More importantly, as we turned each hyperedge into a "real" node, no further preprocessing is required to use the T-Garden graph versions for simulations.

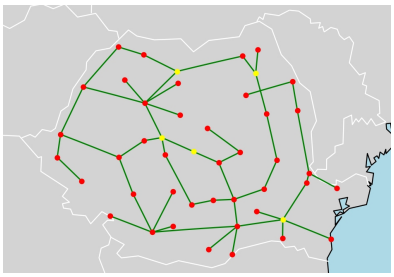
To complement the example of Roedunet, in Fig. 1d, we further displayed the topology as an IP network as displayed in Topology Zoo. Adding the visualization of the Roedunet network in TopoHub,



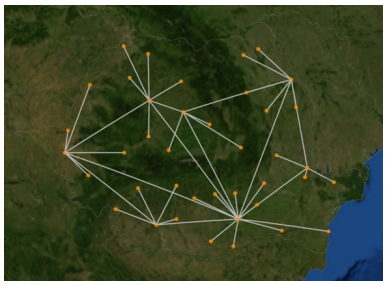
(a) RoEduNet planned architecture[39]



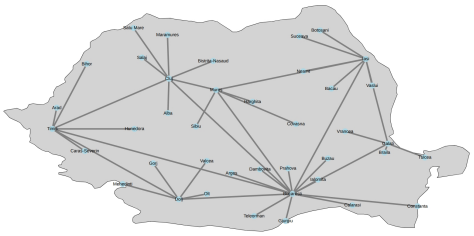
(b) As TopologyZoo fibre network



(c) In T-Garden



(d) As TopologyZoo IP network



(e) In TopoHub

Figure 1: Representation of RoEduNet in different data sets

displayed in Fig. 1e, one can see that the TopoHub version of this network is clearly not encoding the physical layer, thus it is fundamentally different from the networks collected in T-Garden.

SNDLIB

SNDlib[26,27] is a library of data for **S**urvivable fixed telecommunication **N**etwork **D**esign. Its purpose is to provide data about realistic telecommunication networks (including traffic) available to the research community.

SNDlib files are available in two file formats: their own text file format and an XML version of the same files. While the XML version is more universal in nature, our choice fell on the home-brew text version, since this was the obvious intended network format, provided by the SNDlib team. All text files were complete and easily readable, both for humans and from our code with a custom parser. All networks were converted to GML format in our resulting network collection. The following sections will detail the source networks and our resulting network library.

Contrary to the Topology Zoo team, the SNDlib site does not explicitly classify networks, it only presents them in a table, along with some basic network information. While the lack of classification hints at uniform network structures, the reality is that filtering is also necessary for the SNDlib networks. There are two network types on SNDlib: 1) real-world physical networks, where nodes represent real cities with global coordinates as properties; and 2) "logical" networks, where nodes are represented by abstract data and where nodes contain pixel positions on a plane. While on a theoretical level, the second group of networks could also be used in various algorithms, we would like to focus on more real-world oriented problems, and as such, we only adapt the first group of networks from SNDlib, resulting in a total of 10 new networks that can be used. While these networks did not contain the same node variation problems or the missing properties as their Topology Zoo counterparts, they did pose unique challenges.

Data mismatch The parser was relatively straightforward, thanks to the easily readable file format. The data was complete and correct, every node was connected and placed, and there were no repeating labels or edges. Still, some data mismatch was present relative to the precedent set by the Topology Zoo networks.

The network META data was incomplete. The properties that we lacked were geographical extent or location, both of which were filled in manually due to the limited number of networks. Another problem was the missing ID property of nodes. In the SNDlib files, nodes are represented using a label instead of a traditional ID. This means that, when serializing an SNDlib file, the resulting GML would not contain an ID property at every node, resulting in discrepancies compared to the other networks. We solve this problem by simply duplicating the label data into the ID property, resulting in better compatibility.

To make T-Garden a uniform dataset, we decided to omit the data referring to edge capacities for networks taken from SNDlib.

Miscellaneous

Apart from Topology Zoo and SNDlib, there are sporadic sources of backbone networks that can be included in the curated data collected in T-Garden. For now, we have included the Interroute network topology (see Fig. 2) with 25 nodes and 35 links as conserved in paper [40], based on the publicly available network data. A feature of this network topology is its relatively accurate mapping of the network edges to their actual physical routes.

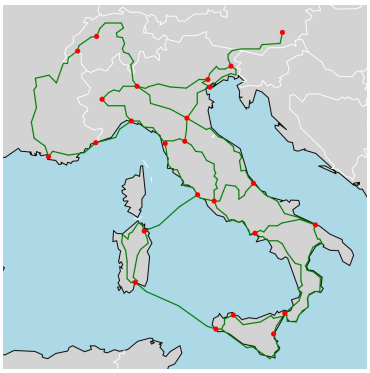


Figure 2: Interroute network

Synthetic networks

Besides the large number of existing backbone networks that were selected from the two main sources, another aim of this research was to provide a varied set of synthetic topologies, to provide further datasets for researchers to use. These networks can be viewed as potential topologies that may emerge in the upcoming decades.

With the synthetic dataset, the main aim was to provide realistic network topologies. In order to provide a good semi-automated generation mechanism, that would excel at approximating real-world properties, we first defined the necessary parameters, and created a new (automated) method, that will be discussed in the following and is summed up in Algorithm 1. To forge truly realistic networks, some of the results were manually post-processed.

Node selection The higher the population in an area, the higher the chance that a node of a backbone topology is located there. Thus, we decided to use GeoNames' [41] `cities1000` database as our city data source of choice, which provided various properties for all settlements in the world, with a population that exceeded 1000 inhabitants at the time of the database creation. Besides city names, we also made use of the concrete numbers of inhabitants, the country, and the region information. At the start of the generation process, a set of regions is defined that will be used as selection parameters. The concrete selection process is split into two possible methods.

One approach is to select the n largest cities from the given regions, which may provide the most meaningful data in terms of major metropolitan regions. A major drawback, however, is the possibility of not selecting sufficiently varied sub-regions. For example, the Benelux sub-region contains a large cluster of cities, and while the importance of these separate cities is undoubted, the relevance of such a small sub-region in a European topological context is less significant.

Another approach for selection was to randomly select n cities from all the cities in the given regions. While a simple, uniformly random selection would have been sufficient to provide a valid alternative, we conceptualized a "smart" random selection, with the main objective being the realistic selection of cities for at least subcontinent-sized regions. First, a closeness limit threshold value was set at 100 `km`, with the generation process being set to try out $2n$ candidates before ignoring this hard limit. The distance of 100 `km` was chosen as a good threshold for sub-continental and larger networks. Secondly, with a large variety of settlements according to population number, a base random selection was unrealistic with the chosen city list mainly containing small towns and villages instead of large cities. To counteract this problem,

Algorithm 1 Network Topology Generation

Input: `region_list` - list of regions; n - city count; `m_percent` - edge count in relation to city count; `mst_type` - type of used mst (`partial` or `complete`); `randomize` - type of city selection used (`random` or `top n`)
Output: $G = (V, E_{\text{result}})$

$V :=$ list of n cities from `region_list`, selection type based on `randomize`
$E_{\text{complete}} :=$ edges of the complete graph on V
$E_{\text{init}} := MST(V)$
if `mst_type` = partial **then**
  Remove 25% of E_{init}
end if
$m :=$ `m_percent` $\cdot\, n$
$E_{\text{result}} := E_{\text{init}}$
Remove edges from E_{complete} that are in, or intersect edges from E_{result}
while $m > |E_{\text{result}}|$ **do**
  $e :=$ next lowest cost edge, still in E_{complete}, with a variable refusal rate (between 10% and 20%)
  Remove e and intersecting edges from E_{complete}
  Add e to E_{result}
end while
Remove unnecessary parallel and spanning edges
Snap unintended intersections to the closest node
return $G = (V, E_{\text{result}})$

but still provide a semi-random selection that would differ from the basic sorted selection, we introduced a weighted random selection process. More concretely, more populated cities are more likely to serve as communication hubs; thus we set the weight of the i^{th} most populated city to $w_i = \frac{1}{1+i}$, lowering the chances of smaller settlements being selected.

Edge selection Edges represent connections between cities. These connections do not follow real-world routes, but approximate them, providing ample opportunity for studying the generated networks. While the possibility of providing a complete graph for a given node list may be enticing at first, connecting every city with every other city would result in a completely unrealistic and dissimilar network to already existing real-world networks. We have to find a subset of every possible edge in the network that both provides good connectivity while maintaining a realistic outlook. Four main properties were identified as necessary in order to provide realistic networks. These go as follows.

1. The network should be (mostly) connected,
2. The network should be (nearly) planar,
3. Shorter connections should have higher probability of being in the network,
4. An appropriate edge number to node number ratio should be chosen.

Trivially, 1) if a network is not connected, it should be treated as multiple separate networks. As of 2), the number of edge crossings in a backbone network is very low (most often zero), as typically a network node is also installed on each link crossing. Thus, for simplicity, we will generate planar networks. Regarding 3), realistically, closer cities would be directly connected, while farther cities would have a route between them that used other cities as intermediaries. Finally, for 4), an edge number to node number ratio equaling 1 results basically to a spanning tree, while this ratio cannot be bigger than 3 (since not even a completely triangulated planar graph has so many edges); a ratio around or slightly above 2 seemed a fine compromise.

The edge generation process starts with generating the complete graph, with the idea behind this process being that selecting from an existing set of edges would be preferential to on-the-fly generation. An intuitive idea was the use of the minimum spanning tree (MST) of the complete graph as a base for our edge list, ensuring that the final generated network would conform to the properties of connectivity and of minimal edges having priority. Two variations of the MST selection were present in the final network list:

1. *complete MST*, where the full MST was included in the network, and
2. *partial MST*, where initially 75% of the MST edges was used, and the rest of the MST was considered only as potential edges.

After the initial edge setup, we remove all selected edges from the potential edge list, along with all edges that would intersect these selected edges. The edges are assigned a value, that correlates with their original length, but also includes two factors that usually impact infrastructure creation in the real world.

- The cost of each edge is set to be its length, then
- underwater edges are given a 30% reduction in cost, and
- edges within the same country have a 50% reduced cost.

The cost reduction of underwater edges simulates the fact that laying an underwater cable would be significantly cheaper than on land connections, where land acquisition and terrain all factor in the higher cost. Secondly, the cost reductions within the same country are chosen since the political cost of international cables is inefficient compared to the lower bureaucracy of national infrastructure.

After the initial setup, the remaining edges are selected from the remaining edge list, going from the lowest cost to the highest cost. In this phase, edge selection is subject to a refusal rate, meaning that some edges do not get selected and are left out; this step ensures that the generation provides varied results. The refusal rate is set to 20% for the initial edge selection. If multiple edges are refused in succession, the refusal rate is lowered to 10% until a selection is finalized, after which the rate returns to the original value. When a selection is successful, all other edges from the list that would intersect the recently chosen edge are removed. This process continues until the predefined desired number of edges is selected, or there are no longer viable edges to choose from. Note that all selected MST edges should also contribute to the final edge count.

In the case of partial MST use, we could not guarantee connectedness from the start; therefore, a corrective action was taken after the edge generation process. If the graph was not connected at the end of the generation, we iteratively added the lowest-cost edges that connected separate components, thereby unifying them. While this process resulted in a connected network, we then need to remove edges to fit the original edge count requirement. This was achieved by removing the same number of edges as the newly added edges, with the condition that the removed edges would not be bridges.

One of the finishing automated steps in the generation process aims to remove unnecessary "parallel" and "spanning" edges, both of which are formed when three edges form an 'almost-degenerate' triangle, by connecting three nodes between themselves. If the three nodes are spaced in a way, that two nodes are positioned relatively closely, while the third node is positioned relatively far away, the two long edges form a parallel structure, which introduces unnecessary long connections, in which case, the longest edge was chosen to be removed. Spanning edges are formed, when the three nodes are placed almost on a line, in which case, the long edge of the structure is yet again redundant and unnecessary. In both cases, the long edge is removed, if its length is at least 90% of the sum of the two shorter edges. Note that after these deletions, the remaining topologies are guaranteed to remain connected, and are liberated from a possibly large number of redundant network elements. This will reduce the number of edges from the designated edge count, but will ensure a much more realistic network.

The final stage of the automated generation process is dedicated to enforcing planarity. The underlying principle of the generator algorithm is to prevent edge intersections that do not fall on existing nodes. However, to improve computational efficiency, the edge intersection analyzer uses a rough approximation, permitting intersections when the computed intersection point lies extremely close to an endpoint of one of the involved edges. To address the inaccuracies introduced by this approximation, a postprocessing step is applied. In this stage, each edge $\{u, v\}$ that intersects another edge $\{w, z\}$ near node w (without being topologically connected to it) is modified through a snapping procedure. Specifically, $\{u, v\}$ is subdivided at the intersection location, and the resulting edges $\{u, w\}$, $\{w, v\}$ are reconnected to the nearby endpoint; if parallel edges were created, they are merged to keep the graph simple. Formally, $E := (E \setminus \{u, v\}) \cup \{u, w\} \cup \{w, v\}$. This transformation eliminates these problematic intersections and moves the graph ever closer to a completely planar form.

Finally, we manually adjusted the generated networks using a homebrew editing tool, with the main objective being the minimization of the number of leaf nodes to ensure a cohesive network structure. Another side objective was the addition of approximately realistic direct intercontinental connections, since their large uninterrupted nature would not fit the generation algorithm, which looks at smaller connections and disregards longer edges. The necessity of these connections was identified even in the early iterations of the generator algorithm, first some consideration went into a possible intercontinental and/or subaquatic cable map integration, but in the end we felt that having hard-coded paths would not be in the spirit of our generation process, with the added complication of hard to come by information regarding the true extent of these connections.

Another thing we considered, was introducing elevation data in our generation cost calculation, but with the lack of good, small-sized, and complete global data, this factor was instead introduced in the other cost-lowering factors, by simply using an average "on land" cost, which remained the original length.

Parameters of the generated synthetic networks We have generated a network for each of the 120 settings that can be chosen from among the following.

- Region: Continental US, US-Canada super-region, Europe, East-Asia super-region, US-Canada + Europe combination, and Global
- Node count: 50, 100, 200, 500, and 1000
- Minimum Spanning Tree: fully used, or each of its edges independently with 75% chance
- Node selection: biggest cities or random

In addition to the settings above, a global network with approximately 2000 nodes was created to provide a larger-scale network for those who require such an option (e.g., for scalability tests).

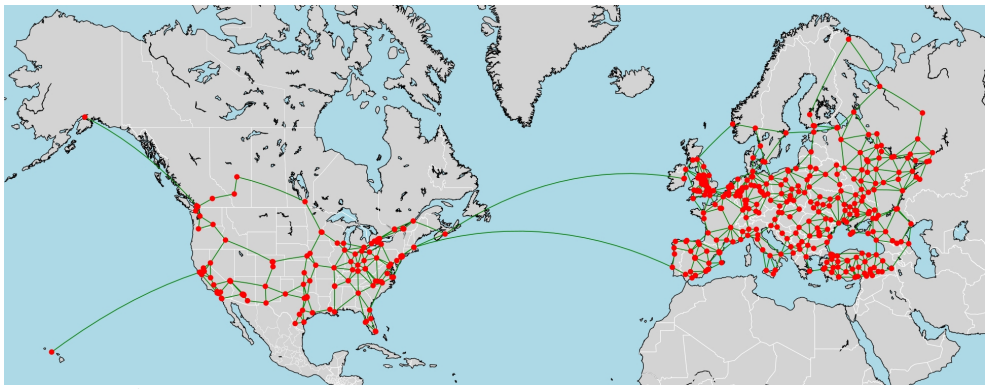
Regions are selected based on an arbitrary selection process, with the possibility of expansion into other regions over time. The edge numbers of the networks are approximately 2.5× the respective node count; apart from manual post-processing, deviations from this multiplier may also arise during the automatic generation phase. Isolated nodes (and other possible small components) were deleted; thus, final node counts could be slightly less than the starting node count. We chose the respective node numbers to fill the void left by the relative absence of larger sourced networks (with more than 100 or even 50 nodes). The deterministic and random node selection process, and the deterministic full, or probabilistic partial use of the edges of a Minimum Spanning Tree are described in more detail earlier in this section, respectively.

Example: In Figure 3, we displayed three synthetic networks from different regions with different sizes. Figure 3a presents an intercontinental network, linking the US-Canada super-region with Europe. This network contains approximately 500 nodes and 1250 links, which is a relatively high number of network elements compared to the real-world sourced networks. Two main intercontinental cables are visible connecting the two regions, approximating the position of a northern and a southern route (matching the partially made-up routes of the Submarine Cable Map of[13]). Nodes are chosen in descending population order of the settlements, explaining the high density node clusters. Edges are built using a partial MST approach.

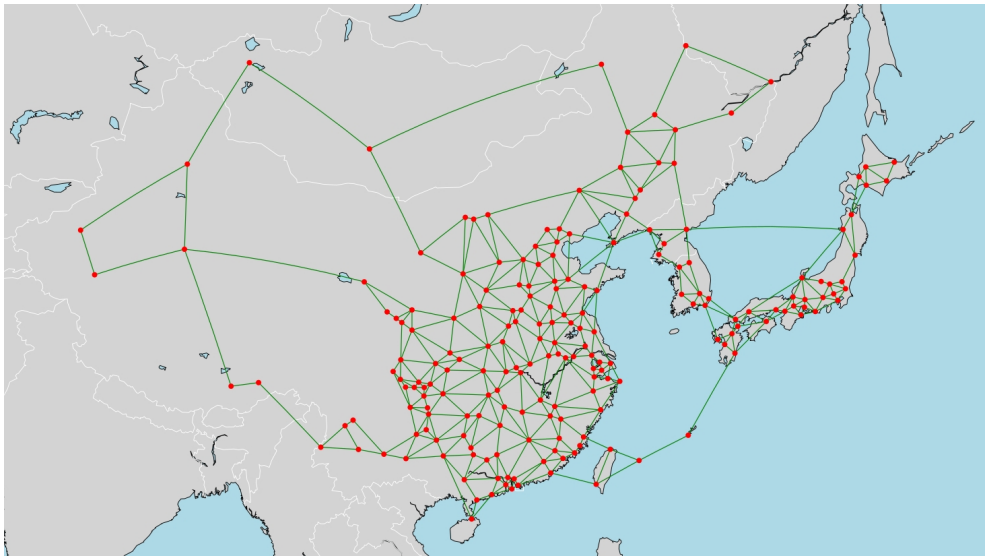
Figure 3b presents a network with randomly chosen nodes, from the East-Asia super-region. The network contains approximately 200 nodes and 500 edges. While the node selection is random, the larger-node preferential approach still results areas with varied densities. The edge set contains the complete MST.

Figure 3c contains the plot of one of our global networks. This instance stands of an approximate number of 100 nodes and 250 edges, which is a relatively small number for a global scale network. Nevertheless, we intended to include as realistic details as possible in this sparse network configuration. A relatively large number of intercontinental connections are present, making this world connection approximation fairly accurate, in terms of interconnectedness.

Input data sources This paragraph summarizes the input data sources used to construct the T-Garden dataset. The real-world part of T-Garden was derived from three public sources. First, we obtained Topology Zoo[25] GML files from archived versions of the Topology Zoo website, accessed through the Internet Archive search query "topology zoo" (`https://archive.org/search?query=topology+zoo`), and selected networks labeled as Fiber or Simplified Fiber. Second, we obtained SNDlib instances from the SNDlib[26,27] repository (`https://sndlib.put.poznan.pl/home.action`) and selected the real-world physical networks with geographic node coordinates, using the original SNDlib text format as input. Third, we included the Interroute topology from the public repository cited in[40] (`https://github.com/jtapolcai/regional-srlg/blob/master/earthquake/network/interroute_j.gml`). For synthetic networks, city candidates were obtained from the GeoNames[41] (`https://www.geonames.org/`) cities1000 database, and missing node coordinates in inherited real-world networks were queried using the OpenStreetMap[38] (`https://www.openstreetmap.org`) API.



(a) Joint North-American – European network on the 500 largest settlements as nodes, with $\sim$ 1250 links, partially containing the MST (minimum spanning tree)



(b) East Asian network on the 200 settlements as nodes (larger cities chosen with bigger probability), with $\sim$ 500 links, containing the MST

(c) Global network on the 100 largest settlements as nodes, with $\sim$ 250 links, containing the MST

Figure 3: Example synthetic networks in T-Garden

Data Records

The T-Garden dataset documented in this Data Descriptor has been deposited as a static, versioned archive on Zenodo[42]. The current dataset consists of a number of **166** networks, out of which **45** are

existing real-world topologies, and **121** are synthetic ones. The dataset can be downloaded in GML format. GML stands for Graph Modeling Language, and it is a hierarchical ASCII-based file format for describing graphs. Many graph handling libraries support GML, such as the NetworkX library in Python, which is the main reason we chose this format as the basis of our graph representation.

Listing 1 presents a basic graph using GML format, with our properties added. Most of the custom properties came from the Topology Zoo network formatting.

The extra information compared to a standard GML file are the following:

- For **nodes**, the *id* property is always stored as a string, in order to accommodate non-integer ids, the optional *label* node is always present. The *Internal* property has a constant value of 1 and it is a remnant of the Topology Zoo property family alongside *Longitude* and *Latitude*, which both represent a node's coordinate information.
- For **edges**, the *source* and *target* properties are also always present as strings, in order to accommodate the string ids. Additionally, all edges contain an *id* field of their own, which represents an edge's name and is also of string type.
- The **meta** information at the beginning of each file contains the *multigraph* property representing the possibility of repeating edges, the network name, the network source and source format, alongside the network's geographical extent and main location.

Listing 1: Example GML File

```
graph [
// Meta information
  node [
    id "1"
    label "Vancouver"
    Internal 1
    Longitude -123.1
    Latitude 49.22
  ]
  node [
    id "2"
    label "Prague"
    hyperedge 1
    Internal 1
    Longitude 14.65748
    Latitude 50.13964
  ]
  edge [
    source "1"
    target "2"
    id "e1"
  ]
]
```

The data is available publicly on Zenodo[42], GitHub (`https://github.com/kepestamas/T-Garden-Networks`), along which we provide a webpage (`tgarden.net`) that presents our data in a collected manner. On the page, each row contains important network information and a link to download the network in GML format or a network plot in SVG format, as plotted by us. Alternatively, each network can be viewed separately by selecting its respective table row, opening a dedicated page that displays the plot along with the relevant information for that network. On this page, a previous and next button ensure navigability between networks, without the need to return to the main network list.

Network collections are also available; conserved networks are grouped by their source, each group having a separate collective download possibility for all networks and all plots from that origin. Generated networks are grouped on the basis of the generation regions. Each region is presented in a separate table, sorted based on network properties, such as node and edge count. The totality of the generated networks and their plots can be downloaded with a single click.

Technical Validation

In order to prove the verisimilitude of the synthetic networks, we made a detailed network analysis of each network. Table 1 describes the network measures for the existing sourced networks, while Table 2 presents network measures of a small selection of medium-sized generated networks. The reported measures are the number of nodes (V), number of edges (E), average value of geometric stretch values ($\texttt{Geom}_{\texttt{avg}}$), average

value of closeness centralities ($C_{\texttt{avg}}$), modularity (Mod), and average value of betweenness centralities ($B_{\texttt{avg}}$).

Along with the above general network measures, the tables also present two important measures that relate to the planarity of each network. $N_{\texttt{cross}}$ presents the number of edge crossings in the geometric representation of each network while $A_{\texttt{cross}}$ represents the average number of crossings per edge.

These metrics capture different insights into the networks. Betweenness centrality measures the importance of a node based on the shortest paths, while closeness centrality measures how close a node is to other nodes in the network. Modularity is a community-related metric: it quantifies how easily a network can be divided into smaller groups. The former two metrics are averaged on nodes. The geometric stretch, on the other hand, has to be averaged over network node pairs s and t. This average of the geometric stretch values ($\texttt{Geom}_{\texttt{avg}}$) as a topology network metric can be useful in determining whether shortest st-paths in a network are effective at approximating the great circle distance between any two nodes s and t of the network on the globe.

Since the generation pipeline was very similar, the synthetic networks have similar values among themselves in every metric category. Contrary to this, these measures vary significantly across the

Network name	$\|V\|$	$\|E\|$	$\texttt{Geom}_{\texttt{avg}}$	$C_{\texttt{avg}}$	Mod	$B_{\texttt{avg}}$	$N_{\texttt{cross}}$	$A_{\texttt{cross}}$
Bandcon	21	28	1.284	0.34	0.354	40.285	3	0.107
Bestel	84	101	1.428	0.098	0.747	480.928	1	0.009
Darkstrand	28	31	1.286	0.209	0.548	79.142	0	0
Dial Telecom	138	151	1.52	0.08	0.77	941.007	2	0.013
FUNET	24	28	1.428	0.237	0.503	60.666	0	0
INS IXC Services	30	38	1.312	0.26	0.526	71.566	0	0
ION	124	149	1.684	0.101	0.738	684.379	8	0.053
ITC Deltacom	113	183	1.258	0.147	0.708	456.973	17	0.092
Intellifiber	73	97	1.343	0.164	0.625	263.753	4	0.041
Interroute	105	153	1.316	0.136	0.688	444.19	44	0.287
Kentucky Datalink	754	899	1.463	0.045	0.875	8933.043	22	0.024
Lambdanet	42	46	1.486	0.161	0.681	149.595	4	0.086
Missouri Network Alliance	64	80	1.462	0.169	0.657	221.906	2	0.025
NTELOS	47	61	1.554	0.169	0.536	164.085	0	0
Network USA	35	39	1.485	0.198	0.479	104.514	2	0.051
Nextgen	17	20	1.234	0.262	0.436	39.647	0	0
OPTOSUNET	26	49	1.615	0.228	0.47	69	6	0.122
OTEGlobe	88	104	1.769	0.134	0.687	inf	31	0.298
Oxford	20	26	1.482	0.313	0.523	40.85	6	0.23
PIONIER	28	32	1.421	0.236	0.56	71.857	0	0
PalmettoNet	45	70	1.331	0.215	0.61	126.888	1	0.014
RoEduNet	46	50	1.781	0.202	0.601	137.065	1	0.02
SWITCH	60	78	1.47	0.185	0.556	193.583	7	0.089
Sago	18	17	1.143	0.195	0.519	54.333	0	0
Shentel	28	35	1.478	0.223	0.501	77.464	0	0
SpiraLight	15	16	1.392	0.3	0.397	30.866	0	0
Syringa Networks	68	68	1.987	0.085	0.75	437.279	0	0
US Carrier	158	189	1.689	0.086	0.789	1027.588	9	0.047
US Signal	61	79	1.363	0.17	0.658	210.77	1	0.012
ValleyNet	39	53	1.186	0.166	0.606	138.435	1	0.018
Viatel	92	96	1.447	0.078	0.776	640.282	0	0
Viatel1	88	92	1.441	0.079	0.744	604.795	0	0
Vision Net	22	21	1.895	0.199	0.616	65.727	1	0.047
abilene	12	15	1.148	0.41	0.425	19.25	0	0
cost266	37	57	1.239	0.272	0.553	85.297	0	0
euNetworks	14	19	1.186	0.366	0.4	24.571	0	0
geant	22	36	1.3	0.405	0.335	37.09	22	0.611
germany50	50	88	1.185	0.251	0.606	123.68	3	0.034
italy	25	35	1.23	0.277	0.482	56.519	0	0
janos-us	26	42	1.118	0.311	0.471	53.846	0	0
janos-us-ca	39	61	1.145	0.244	0.563	98.897	0	0
nobel-eu	28	41	1.267	0.287	0.575	61.571	0	0
nobel-germany	17	26	1.109	0.383	0.225	29.588	0	0
nobel-us	14	21	1.265	0.468	0.164	20.428	6	0.285
polska	12	18	1.195	0.474	0.322	17.25	0	0

Table 1: Conserved real-world networks

Network name	$\|V\|$	$\|E\|$	$\texttt{Geom}_{\texttt{avg}}$	$C_{\texttt{avg}}$	Mod	$B_{\texttt{avg}}$	$N_{\texttt{cross}}$	$A_{\texttt{cross}}$
East Asia_200_500_mst	197	439	1.163	0.141	0.657	813.862	0	0
East Asia_200_500_mst_rand	199	449	1.159	0.137	0.682	831.442	0	0
East Asia_200_500_pmst	197	436	1.167	0.141	0.663	803.563	0	0
East Asia_200_500_pmst_rand	200	449	1.181	0.134	0.695	852.749	0	0
Europe_200_500_mst	200	417	1.149	0.124	0.724	918.5	1	0.002
Europe_200_500_mst_rand	200	430	1.143	0.135	0.752	851.12	0	0
Europe_200_500_pmst	200	419	1.15	0.121	0.728	945.6	1	0.002
Europe_200_500_pmst_rand	200	440	1.141	0.132	0.704	866.76	0	0
Global_2000_5000_mst_rand	1977	4324	1.223	0.04	0.868	inf	0	0
Global_200_500_mst	194	393	1.166	0.13	0.648	848.494	0	0
Global_200_500_mst_rand	199	503	1.15	0.148	0.7	773.447	0	0
Global_200_500_pmst	194	396	1.179	0.138	0.664	807.381	0	0
Global_200_500_pmst_rand	200	497	1.195	0.147	0.652	781.204	1	0.002
North America+Europe_200_500_mst	200	408	1.13	0.108	0.643	1040.68	0	0
North America+Europe_200_500_mst_rand	200	425	1.122	0.106	0.68	1074.025	0	0
North America+Europe_200_500_pmst	200	408	1.128	0.107	0.662	1059.004	0	0
North America+Europe_200_500_pmst_rand	199	425	1.124	0.117	0.638	974.959	0	0
North America_200_500_mst	194	381	1.124	0.124	0.661	892.283	1	0.002
North America_200_500_mst_rand	199	428	1.137	0.137	0.718	830.974	0	0
North America_200_500_pmst	194	386	1.12	0.133	0.659	833.958	1	0.002
North America_200_500_pmst_rand	196	425	1.12	0.139	0.712	809.244	0	0
US_200_500_mst	191	378	1.147	0.115	0.592	940.225	0	0
US_200_500_mst_rand	192	424	1.116	0.131	0.682	846.25	0	0
US_200_500_pmst	191	374	1.152	0.121	0.617	900.434	0	0
US_200_500_pmst_rand	190	431	1.107	0.14	0.637	790.215	0	0

Table 2: Medium sized ($\sim$ 200 nodes) synthetic networks

sourced networks, especially based on the size and density of the networks in question. Additionally, large differences can be observed between networks sourced from different sources.

An observed fact is that the generated networks, mainly the non-random selection variants, all form dense communities of nodes where some cities form large suburbia, while many of their real-world counterparts approximate large city clusters with singular nodes. Another observation is that the generated networks all fall on the larger side, with multi-region networks containing hundreds of nodes being the norm, whereas the sourced networks usually fall on the smaller side, with the norm being less than **50** nodes. This difference is intentional, since there is a clear lack of medium and large-sized network topologies.

Regarding planarity, it can be observed that most networks in both the sourced and the generated datasets satisfy the criterion of "near-planarity." Three categories of networks can be distinguished. The first category consists of completely planar networks, which are both mathematically planar (with respect to graph structure) and geometrically planar (i.e., they exhibit no edge crossings when node positions are fixed and edges are approximated as great-circle arcs on the Earth's spheroid). The second category includes networks that are mathematically planar but only geometrically near-planar. The third category comprises networks that are mathematically non-planar yet geometrically near-planar.

To quantify the results of the network analysis, among the 45 sourced networks, 21 are completely planar, 18 are mathematically planar only, and 6 are quasi-planar. In the case of the generated networks, out of 121 total networks, 88 are completely planar, 2 are mathematically planar only, and 31 are quasi-planar.

A notable similarity between the two datasets is observed in the average number of edge crossings per network, which is 4.53 for the sourced networks and 1.71 for the generated networks. However, this comparison is biased, as the generated networks are, on average, larger. Also, some of the sourced networks may contain some links that, in fact, are suspected logical links, which can introduce numerous link crossings if treated like a physical link embedded as a geodesic – such an example is the London-Dubai edge of the Interroute European network. When instead considering the average number of crossings per edge, the values are 0.058 for the real-world networks and 0.001 for the generated networks, indicating an order-of-magnitude difference. This result provides strong validation for the generator algorithm, as it produces networks that are significantly closer to planarity, and consequently, they closely resemble physical backbones. The small number of non-planar networks with the minimal number of intersections may be the result of the after-the-fact manual modifications.

Beyond the intentional and emerging differences in the sourced and generated networks, we argue

that the generated networks capture and reflect the essence of what *being a backbone network* is. While, in some sense, 'backbone-networkness' is hard to formally capture, we did our best so that, among the generated backbone topologies in T-Garden, anyone could find networks that suit their personal convictions. Turning to concrete metrics, the geometric stretch is a good example to showcase the similarity of the generated networks to the sourced ones. According to the $\texttt{Geom}_{\texttt{avg}}$, most SNDlib networks produce a value around 1.1 to 1.2, which is the exact range into which the synthetic networks fall.

Data availability

The version of the T-Garden dataset described in this article is available from Zenodo (`https://doi.org/10.5281/zenodo.21116252`). We also made the generated and the adapted data publicly available on GitHub (`https://github.com/kepestamas/T-Garden-Networks`).

Code Availability

As the primary contribution of this work is the dataset itself rather than an implementation, only a research grade variant of the most important snippets of the generation code is publicly released (`https://github.com/kepestamas/T-Garden-Code`).

References

[1] Critical Infrastructure Sectors, Cybersecurity and Infrastructure Security Agency (CISA). `https://www.cisa.gov/critical-infrastructure-sectors`. Last accessed on: 16/12/2025.

[2] Review of Critical Infrastructure Domains in Europe, SPEAR Project. `https://spear2020.eu/News/Details?id=120`. Last accessed on: 16/12/2025.

[3] Mohamed El-Sayed and Jeffrey Jaffe. A view of telecommunications network evolution. *IEEE Communications Magazine*, 40(12):74–81, 2002.

[4] Eugenio Iannone. *Telecommunication networks*. CRC Press, 2017.

[5] T Gomes, J Tapolcai, C Esposito, D Hutchison, F Kuipers, J Rak, A de Sousa, A Iossifides, R Travanca, J Andre, L Jorge, L Martins, P Ortiz Ugalde, A Pasic, D Pezaros, S Jouet, S Secci, and M Tornatore. A survey of strategies for communication networks to protect against large-scale natural disasters. In *Int. Workshop on Reliable Networks Design and Modeling (RNDM)*, 2016.

[6] George A Barnett. Recent developments in the global telecommunication network. In *2012 45th Hawaii International Conference on System Sciences*, pages 4435–4444. IEEE, 2012.

[7] Jeremiah F Hayes and Thimma VJ Ganesh Babu. *Modeling and analysis of telecommunications networks*. John Wiley & Sons, 2004.

[8] Rasmus L Olsen, Kartheepan Balachandran, Sara Hald, Jose Gutierrez Lopez, Jens Myrup Pedersen, and Matija Stevanovic. Telecommunication networks. In *Intelligent Monitoring, Control, and Security of Critical Infrastructure Systems*, pages 67–100. Springer, 2014.

[9] Fernando A. Kuipers, Freek Dijkstra, and Piet Van Mieghem. Graph theory and complex networks: An introduction. *Computer Networks*, 145:1–14, 2018. `doi:10.1016/j.comnet.2018.07.003`.

[10] Massimo Tornatore, Achille Pattavina, and Antonio Pescapè. Optical network design and modeling: State of the art and future directions. *IEEE Journal on Selected Areas in Communications*, 37(5):1042–1054, 2019. `doi:10.1109/JSAC.2019.2906783`.

[11] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies with Rocketfuel. *IEEE/ACM Transactions on Networking*, 12(1):2–16, 2004.

[12] Steven Gay, Pierre Schaus, and Stefano Vissicchio. Repetita: Repeatable experiments for performance evaluation of traffic-engineering algorithms. *arXiv preprint arXiv:1710.08665*, 2017.

[13] Telegeography. `www2.telegeography.com/en/our-research`. Last accessed on: 16/12/2025.

[14] Alexander Schrijver. Finding k disjoint paths in a directed planar graph. *SIAM Journal on Computing*, 23(4):780–788, 1994. doi:10.1137/S0097539792224061.

[15] Sebastian Neumayer, Alon Efrat, and Eytan Modiano. Geographic max-flow and min-cut under a circular disk failure model. In *IEEE INFOCOM*, pages 2736–2740, 2012.

[16] Y. Kobayashi and K. Otsuki. Max-flow min-cut theorem and faster algorithms in a circular disk failure model. In *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, pages 1635–1643, April 2014. doi:10.1109/INFOCOM.2014.6848100.

[17] Balázs Vass, Erika Bérczi-Kovács, Ábel Barabás, Zsombor László Hajdú, and János Tapolcai. Polynomial-time algorithm for the regional SRLG-disjoint paths problem. In *Proc. IEEE INFOCOM*, London, United Kingdom, May 2022.

[18] Erika Bérczi-Kovács, Péter Gyimesi, Balázs Vass, and János Tapolcai. Efficient Algorithm for Region-Disjoint Survivable Routing in Backbone Networks. In *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications (INFOCOM 2024)*, Vancouver, Canada, May 2024.

[19] Balázs Vass, Levente Birszki, Erika Bérczi-Kovács, Péter Babarczi, Péter Gyimesi, and János Tapolcai. Availability-Aware Routing in Presence of Geographically Correlated Failures. In *IEEE INFOCOM 2026 - IEEE Conference on Computer Communications (INFOCOM 2026)*, Tokyo, Japan, May 2026.

[20] Elias Dahlhaus, David S. Johnson, Christos H. Papadimitriou, Paul D. Seymour, and Mihalis Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994. doi:10.1137/S0097539792225297.

[21] Éric Colin de Verdière. Multicuts in planar and bounded-genus graphs with bounded number of terminals. In *Algorithms – ESA 2015*, volume 9294 of *Lecture Notes in Computer Science*, pages 373–385. Springer, 2015. doi:10.1007/978-3-662-48350-3_32.

[22] Steven Fortune, John Hopcroft, and James Wyllie. The directed subgraph homeomorphism problem. *Theoretical Computer Science*, 10(2):111–121, 1980. doi:10.1016/0304-3975(80)90009-2.

[23] Jeff Erickson, Clyde Monma, and Arthur Veinott. Send-and-split method for minimum-cost network flow. *Mathematical Programming*, 77(2):157–176, 1997.

[24] András Frank and Tibor Jordán. Minimal edge-coverings of pairs of sets. *Journal of Combinatorial Theory, Series B*, 57(1):17–31, 1993.

[25] Simon Knight, Hung X Nguyen, Nickolas Falkner, Rhys Bowden, and Matthew Roughan. The Internet Topology Zoo. *IEEE Journal on Selected Areas in Communications*, 29(9):1765–1775, 2011. URL: http://www.topology-zoo.org/dataset.html.

[26] S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly. SNDlib 1.0–Survivable Network Design Library. In *Proc. of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium*, April.

[27] Sebastian Orlowski, Roland Wessäly, Michal Pióro, and Artur Tomaszewski. SNDlib 1.0: survivable network design library. *Networks*, 55(3):276–286, 2010. URL: https://sndlib.put.poznan.pl/home.action.

[28] Piotr Jurkiewicz. TopoHub: A repository of reference Gabriel graph and real-world topologies for networking research. *SoftwareX*, 24:101540, 2023.

[29] Jérôme Kunegis. KONECT: the Koblenz network collection. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13 Companion, page 1343–1350, New York, NY, USA, 2013. Association for Computing Machinery.

[30] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 4292–4293, 2015. URL: https://networkrepository.com.

[31] Christopher L. DuBois. UCI network data repository, 2008. URL: http://networkdata.ics.uci.edu.

[32] Jure Leskovec and Rok Sosič. SNAP: A general-purpose network analysis and graph-mining library. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(1):1, 2016. URL: `https://snap.stanford.edu/data/`.

[33] Timothy A. Davis and Yifan Hu. The University of Florida sparse matrix collection. *ACM Transactions on Mathematical Software*, 38(1):1–25, 2011. URL: `https://sparse.tamu.edu/`.

[34] CAIDA. The CAIDA anonymized Internet datasets. `https://www.caida.org/data/`, 2024. Last accessed on: 16/12/2025.

[35] RIPE NCC. RIPE Atlas: Internet Measurements. `https://atlas.ripe.net/`. Last accessed on: 16/12/2025.

[36] RIPE NCC. RIPE Routing Information Service (RIS). `https://ris.ripe.net/`. Last accessed on: 16/12/2025.

[37] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. In Gaël Varoquaux, Travis Vaught, and Jarrod Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

[38] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.

[39] Octavian Rusu. Photonic network for research and education (roedunet). `https://niham.nipne.ro/aliceworkshop08/download/Photonic%20RoEduNet.pdf`, 2008. Last accessed on: 16/12/2025.

[40] A. Valentini, B. Vass, J. Oostenbrink, L. Csák, F. Kuipers, B. Pace, D. Hay, and J. Tapolcai. Network resiliency against earthquakes. In *2019 11th International Workshop on Resilient Networks Design and Modeling (RNDM)*, pages 1–7, Oct 2019. URL: `https://github.com/jtapolcai/regional-srlg/blob/master/earthquake/network/interroute_j.gml`

[41] Unxos GmbH. GeoNames. `https://www.geonames.org/`. Last accessed on: 16/12/2025.

[42] Tamás-Zsolt Képes, Noémi Gaskó, János Tapolcai, and Balázs Vass. A curated dataset of real and synthetic quasi-planar Internet backbone network topologies. Dataset, Zenodo, Version v1, 2026. `https://doi.org/10.5281/zenodo.21116252`.

Author Contributions

B. Vass and J. Tapolcai initially identified the gap in quasi-planar backbone data availability. T.-Zs. Képes curated the existing and forged the new data, with the help of the rest of the team members. T.-Zs. Képes created the related webpage. All authors contributed to the data validation. B. Vass and T.-Zs. Képes led the writing with contributions from all the other authors. Corresponding author: B. Vass (email: balazs.vass@ubbcluj.ro).

Competing Interests

The authors declare no competing interests.

Funding

This work was supported by a grant of the Romanian Ministry of Research, Innovation and Digitization, CNCS-UEFISCDI, project number PN-IV-P2-2.1-TE-2023-1977.