Seventh FRAMEWORK PROGRAMME
FP7-ICT-2009-5 - ICT-2009-1.6
Future Internet experimentally-driven research

SPECIFIC TARGETED RESEARCH OR INNOVATION PROJECT

Deliverable D3.3

"Graph analysis / mining"

Project description	
Project acronym:	**EULER**
Project full title:	**Experimental UpdateLess Evolutive Routing**
Grant Agreement no.:	**258307**
Document properties	
Number:	**FP7-ICT-2009-5-1.6-258307-D3.1**
Title:	**Graph Analysis and Mining**
Responsible:	**Institut National de Recherche en Informatique et Automatique (INRIA)**
Contributor(s):	**All partners**
Dissemination level:	**Public (PU)**
Date of preparation:	**September 2012**
Version:	**1.0**

Affiliation	Authors
CTI	Ioannis Caragiannis, Panagiotis Kanellopoulos
INRIA	David Coudert, Christian Glacet, Nicolas Hanusse, Aurélien Lancin, Nicolas Nisse
UPMC	Fabien Tarissan
ALB	Dimitri Papadimitriou

List of authors

Contents

Chapter 1

Context and document summary

As pointed out in the deliverable D3.1, the holly grail of Task 3.1 in EULER is the development of a new graph model for the Internet that will encompass the advantages of the different models that have been proposed so far. Clearly, such an understanding of the Internet structure will have important benefits for routing.

In this deliverable, we consider two different levels of Internet topology referring to the *network of the interconnections of routers* and/or to the *network of Autonomous Systems* (AS). Such topologies are modeled as undirected graphs. The interconnection network between the routers represents the Internet at the physical level. At this low level of description, nodes of the graph represent physical entities (such as routers and points of interconnection) and the edges of the graph correspond to the physical links (or bundles of physical links) interconnecting them. Usually, the physical entities (*the routers*) are classified as *access, edge, aggregation entities, and interconnection points (the so-called IXP's)* that mainly differ in their number or in- and out- physical interfaces and by their switching or forwarding capabilities. The AS network corresponds to a higher level of description of the Internet. At this level of abstraction, ASes are logical nodes that correspond to a collection of connected Internet Protocol (IP) prefixes. Interconnections between ASes define the relationships between them. These interconnections can be classified as "customer-provider", "peering" (peering can either be public/shared or private), and "mutual-transit". The corresponding network is considered as the topology on which the routing function is being applied when performing inter-AS routing. In this deliverable, we rely on the maps of the advised ASes collected by CAIDA [CAI].

The driving idea of the EULER project is to make use of the structural and statistical properties of the Internet topology in order to specialize the design of efficient distributed routing schemes under dynamic network and policy conditions. For this purpose, Tasks T3.1 and T3.2 of Work Package 3 of the EULER project are dedicated to the study of the actual structural properties of the Internet and of their evolution under dynamic scenarii. Note that we mainly focus on properties related to distances and paths. The considered paths follow two levels of description. First, we consider *topological paths*, i.e., sequence of nodes such that two consecutive nodes are interconnected by a (logical) link. For instance, we observe the distribution of the length of the shortest paths both in the router-level network and in the AS network. Second, we consider the *forwarding (or routing) paths* that are actually followed by packets toward destination, i.e., the paths that are computed by a routing function using the information stored in the routing tables.

These objectives are very ambitious since they require to overcome several challenging difficulties. First, determining the actual properties of the Internet require accurate measurement of its topology. Second, because of the huge size of the considered networks, efficient algorithms are needed to compute the desired properties. Indeed, since we are interested by the short term and long term evolutions of various structural parameters, this requires to be able to repeat many times their computation. Last but not least, the treatment of such a bunch of data is itself a challenging issue.

During the first year of EULER, WP3 has started to tackle the above issue. We first have focused on a careful investigation of the related models proposed in the literature and an as extensive as possible selection of graph properties that are more likely to affect routing performance. These first results have been reported in D3.1. While several well known properties of the Internet are well documented in the literature (diameter, degree-distribution, clustering coefficient), it appears that other properties that are well known to be useful for routing have not received such an attention (hyperbolicity, chordality, centrality betweenness, etc.). Moreover, very few studies of the evolution of structural properties of the Internet or of the corresponding models have been done. This is mainly due to the complexity of their computation and the lack of algorithms that can actually be executed for networks with an order of 10^4 nodes. It is also worth to note that, while many models have been proposed to represent the Internet topology (such as GLP), no model allows to overlap all known properties of the Internet. An important working effort of WP3 has been done to implement several graph models and algorithms for property testing. Preliminary results from this implementation and experiments in testbed environments have been performed on topologies with at most 1 000 nodes and were also reported in D3.1.

During the second year of EULER, our activities has been heavily based on the results and implementations obtained during the first period. We have first continue the understanding of the Internet topological structure using graph-theoretical notions, models, and techniques. This led us to the design of new efficient algorithms for computing non trivial properties of graphs. In particular, the proposed algorithms have allowed to provide new statistical results regarding huge networks such as the CAIDA maps and classical or new models (Erdös-Rényi, GLP, bipartite, etc.).

The main focus of the second part of T3.1 concerns the evolution of the structural properties of the Internet or of models under dynamicity conditions. Some graphs problems have been widely studied in dynamic graphs. However, these studies only deal with classical problems such as the maintenance of a spanning tree (or forest) in a dynamic graph [DKK07, Ita08b, Ita08a]. Similarly, in the large scale network context, only the evolution of "simple" properties such as diameter, degree or distance distribution have been considered. However, when aiming at efficient routing schemes, such studies are not sufficient. Indeed, the fact that the diameter or even the distance distribution of an evolving network does not increase does not mean that the distance between two given nodes remains the same. The main issue of T3.1 is not only the evolution of the structural properties of the Internet but the impact of these evolution to routing. In particular, the evolution of some properties may allow us to discover some other hidden properties: the fact that the distribution of the hyperbolicity remains unchanged when removing until 10% of the edges of a GLP instance or of a CAIDA map reveals the redundancy of the shortest paths between nodes.

The rest of this document is structured as follows. First two chapters are a direct continuation of previous work of task T3.1 of WP3. In Chapter 3, we provide new efficient algorithms

to compute interesting structural properties that have been pointed out in deliverable D3.1. These algorithms are proved to be very useful for the results described in the second part of this deliverable. Then, in Chapter 4, we propose a new model of the Internet topology as a bipartite graph characterizing the connections between AS routers and Ethernet switches. This model takes advantage of new measurements approaches provided in deliverable D3.2. Chapter 5 is devoted to a deep statistical analysis of the evolution of the properties of the CAIDA maps during the last decade. This is done in parallel with the study of the evolution (under the same kind of dynamic) of the properties of various models. In Chapter 6, we focus on the evolution of the properties under dynamic conditions. Due to the time-complexity of measuring the properties, we focus on simple scenarii consisting on random removal or addition of some edges/nodes. First, we provide first theoretical investigations on the evolution of the distances in such case and its impact on several routing schemes. In Section 6.1, we provide a theoretical analysis of the number of out-dated routing tables after the removal of some edges. In Section 6.2, we study the evolution of the performance of greedy routing, based on some tree-embedding of a graph, when fails make some edges fall.

Chapter 2

Notations

In this section, we define the standard notations from graph theory used in this deliverable.

Table 2.1: Standard notations used in this deliverable.

Notation	**Definition**
V	Vertex set
E	Edge set
$G = (V, E)$	Undirected graph with vertex set V and edge set E
n	Order or number of vertices of the graph, $n = \|V\|$
m	Size or number of edges of the graph, $m = \|E\|$
ρ	Density of the graph: $\rho = \frac{2m}{n(n-1)}$
$\text{dist}(u, v)$	Shortest path distance between vertices u and v
D	Diameter
$\Gamma(u)$	Set of neighbors of vertex u in the graph.
$\Gamma(X)$	Set of neighbors of the vertices $u \in X$. $\Gamma(X) = \cup_{u \in X} \Gamma(u)$.
$\deg(u)$	Degree of vertex u, $\deg(u) = \|\Gamma(u)\|$
Δ	Maximum degree
$\Gamma[u]$	Strict neighborhood of u, that is $\Gamma[u] = \Gamma(u) \setminus \{u\}$
$\Gamma[X]$	Strict neighborhood of the vertices $u \in X$, that is $\Gamma[X] = \Gamma(X) \setminus X$
δ	Hyperbolicity of the graph [Gro87]. For convenience, we note $\delta^* = 2\delta$.
tw	Treewidth of G
tl	Tree-length of G

Chapter 3

New algorithms for property testing

In Section 2 of D3.1, we have reviewed several structural properties of the Internet which are proven effective either for the design of compact routing schemes, or for distance decreasing routing schemes (greedy routing). While several well known properties of the Internet are well documented in the literature (diameter, degree-distribution, clustering coefficient), it appears that other properties that are well known to be useful for routing have not received such an attention (hyperbolicity, chordality, centrality betweenness, etc.). Indeed, the latter properties or parameters reflect the closeness of the structure or the metric of the Internet with the ones of a tree. This is clearly a good news for routing since efficient (labelled) compact routing schemes exist in tree topologies. The fact that few studies deal with properties such as hyperbolicity or chordality is mainly due to their computational complexity.

In this chapter, we continue this study of some structural properties. In particular, we present efficient algorithms for computing the hyperbolicity of graphs and some nice tree-decomposition of graphs with small chordality. The difficulties met to compute these properties are distinct. The hyperbolicity is known to be computable in time $O(n^4)$ in n-node graphs which is prohibitive for large graphs. On the other hand, chordality is known to be NP-complete even in planar graphs. In this chapter, we propose the first exact algorithm for computing hyperbolicity that is scalable for Internet-like graphs. We also design a greedy algorithm providing good bounds on chordality and treewidth.

In Chapter 5 of these deliverable, we describe experimentations using these algorithms that we have performed to study the evolution of these parameters on the Internet topology.

3.1 Hyperbolicity

This section extends the study on hyperbolicity initiated during the first year of EULER and reported in Section 2.15 of deliverable D3.1. Some preliminary experimental results were also presented in Section 3.2 of D3.1 (e.g., Table 3.8). The main contribution of this section is the design of a new algorithm that allowed us to conduct an extensive study of the hyperbolicity of large-scale graphs such as CAIDA maps.

3.1.1 Objectives and Motivation

The (Gromov) *hyperbolicity* of a graph reflects how the metric (distances) of the graph is close to the metric of a tree (see Section 2.15 of Deliverable D3.1). Gromov [Gro87] defines

the notion of δ-hyperbolic metric spaces using the notion of δ-thin triangles. Given any three points x, y, and z of a hyperbolic metric space, the triangle (x, y, z) is δ-thin if any point of the geodesic joining x and y is at distance at most δ of one of the geodesics joining x to z or y to z. A δ-hyperbolic space is a geodesic metric space in which every geodesic triangle is δ-thin. In other words, a graph has hyperbolicity $\leq \delta$ if, for any $u, v, w \in V(G)$ and for any shortest paths P_{uv}, P_{vw}, P_{uw} between these three vertices, any vertex in P_{uv} is at distance at most δ from $P_{vw} \cup P_{uw}$ [Gro87]. Intuitively, in a graph with small hyperbolicity, any two shortest paths between the same pair of vertices are close to each other. For instance, a graph has hyperbolicity 0 if and only if it is acyclic.

An alternative definition given by Gromov [Gro87], and called the 4-*points condition*, is the following. A metric space is δ-hyperbolic if for any four points u, v, w, x the two larger of the distance sums $\mathrm{dist}(u, v) + \mathrm{dist}(w, x)$, $\mathrm{dist}(u, w) + \mathrm{dist}(v, x)$, $\mathrm{dist}(u, x) + \mathrm{dist}(v, w)$ differ by at most $\delta^* = 2\delta$. These notions extend to connected graphs, and we say that a connected graph $G = (V, E)$ equipped with its standard graph metric dist_G (shortest path distance) is δ-hyperbolic if the metric space (V, dist_G) is δ-hyperbolic. In other words, a connected graph $G = (V, E)$ is δ-hyperbolic if it satisfies the 4-points condition. The hyperbolicity of a graph measures its tree-likeness. The less the value of δ is, the more the graph looks like a tree.

Determining the hyperbolicity δ of a graph of order n can therefore be done in time $O(n^4)$. In fact, since all 4-tuples have to be checked, the time complexity is in $O(\binom{n}{4})$. An implementation of the naive algorithm for determining the hyperbolicity of a graph as been included into the distory (Distance Between Phylogenetic Histories) package [dis] of the CRAN (The Comprehensive R Archive Network) project [CRA]. This package is devoted to the study of geodesic distance between phylogenetic trees and associated functions. The implementation uses the "revolving doors Gray code" principle [Knu05] for visiting all 4-tuples of the input graph. Unfortunately, the proposed implementation is impracticle for large and dense graphs (size of the largest biconnected component). For instance, for a graph with a biconnected component of order 10 000, the number of 4-tuples to consider is higher than $4 \cdot 10^{14}$, which requires approximately 100 hours of computation assuming that 10^9 4-tuples are evaluated per second. A heuristic algorithm for determining the hyperbolicity of CAIDA AS maps has been used in [dMSV11]. Other heuristics based on sampling were used in [Sha11, NST12] for random Erdös-Renyi GNP graphs generated using probability $p = c/n$, where c is a small constant (1.5 or 2), and so the resulting graphs are almost trees having very small biconnected components. In addition, a 2-approximation algorithm, with running time in $O(\binom{n}{3})$, is obtained fixing one vertex and evaluating all possible 4-tuples containing that vertex [CDE+08].

The need for more efficient algorithms has been expressed by many authors, for instance it has recently been mentionned in the conclusion of [CFHM12] that "*exact computation of δ by its definition takes $O(n^4)$ time, which is not scalable to large graphs, and thus the design of more efficient exact or approximation algorithms would be of interest*".

The algorithm proposed below is the first exact algorithm scalable for large graphs.

Outline of the section. We present a new exact algorithm for computing the hyperbolicity of a graph. This algorithm has worst case time complexity in $O(n^4)$, but in practice the optimal value is returned much faster. In particular, this algorithm has been used to compute the hyperbolicity of the last-years CAIDA maps which allowed us to understand the evolution of the hyperbolicity of the Internet. These results are presented in Section 5.2.3 page 59.

3.1.2 Exact algorithm for computing the hyperbolicity: Design and theoretical analysis

In this section, we formally describe a new exact algorithm for computing the hyperbolicity of graphs. We then give some hints on its time-complexity. Last but not least, we explain how to turn this algorithm into an approximation algorithm.

Our new algorithm is based on the following lemma, in which $\text{dist}(x, y)$ denotes the graph distance between nodes x and y, and $\delta^*(a, b, c, d)$ is the computed value of the hyperbolicity of the 4-tuple (a, b, c, d). Recall that $\delta^* = 2\delta$ and so that $\delta^*(a, b, c, d) = 2\delta(a, b, c, d)$.

Lemma 1. *Let $G = (V, E)$ be a connected graph, let $a, b, c, d \in V$, let $S_1 = \text{dist}(a, b) + \text{dist}(c, d)$, $S_2 = \text{dist}(a, c) + \text{dist}(b, d)$, and $S_3 = \text{dist}(a, d) + \text{dist}(b, c)$, and assume w.l.o.g. that $S_1 \geq \max\{S_2,\ S_3\}$. We have $\delta^*(a, b, c, d) \leq \min\{\text{dist}(a, b),\ \text{dist}(c, d)\}$.*

Proof. We have $S_2+S_3 = \text{dist}(a, c)+\text{dist}(b, d)+\text{dist}(a, d)+\text{dist}(b, c) = (\text{dist}(a, c) + \text{dist}(b, c))+ (\text{dist}(a, d) + \text{dist}(b, d))$. Using the triangular inequality, we deduce $S_2+S_3 \leq 2\cdot\text{dist}(a, b)$. Since S_1 is the largest sum, we have $\delta^*(a, b, c, d) = S_1 - \max\{S_2,\ S_3\} \leq S_1 - \max\{S_2,\ S_3\}/2 = S_1 - \text{dist}(a, b) = \text{dist}(c, d)$.

We obtain similarly that $\delta^*(a, b, c, d) \leq \text{dist}(a, b)$. □

To make use of Lemma 1, we construct in Algorithm 1 the list SLL of 3-tuples $(S_1 = \ell_1 + \ell_2, \ell_1, \ell_2)$, for $1 \leq \ell_2 \leq \ell_1 \leq D$, sorted in decreasing lexicographic order (lines 3-4). The value of ℓ_i is a shortest path distance in the graph. For instance, if $D = 4$, the 3-tuples are sorted in the following order:

$$(8, 4, 4),\ (7, 4, 3),\ (6, 4, 2),\ (6, 3, 3),\ (5, 4, 1),\ (5, 3, 2),\ (4, 3, 1),\ (4, 2, 2),\ (3, 2, 1),\ (2, 1, 1).$$

We then compute for each 3-tuple $(S_1 = \ell_1 + \ell_2, \ell_1, \ell_2)$ the values $\text{h}(a, b, c, d) = S_1 - \max\{\text{dist}(a, c)+\text{dist}(b, d),\ \text{dist}(a, d)+\text{dist}(b, c)\}$ for all 4-tuple (a, b, c, d) such that $\text{dist}(a, b) = \ell_1$ and $\text{dist}(c, d) = \ell_2$. Thus, the algorithm considers the 4-tuples (a, b, c, d) in such a way that either S_1 is the largest sum, or, in case S_2 (resp. S_3) is larger than S_1 we know that the 4-tuple has previously been considered with S_2 (resp. S_3) as maximum value, and so we have $\text{h}(a, b, c, d) \leq 0$. Then, thanks to Lemma 1 we know that at any step of the algorithm the value of ℓ_2 is an upper bound on the value of $\text{h}(a, b, c, d)$. Also, if the current lower bound is h^*, none of the 4-tuples such that $\text{dist}(c, d) \leq \text{h}^*$ can be used to improve the lower bound. We can thus cut exploration (lines 6-8).

Since we have $\binom{n}{2}$ pairs, and that Algorithm 1 considers pairs of pairs, the worst case time complexity of the algorithm is in $O\left(\binom{n}{2} \cdot \left(\binom{n}{2} - 1\right)/2\right)$. However, with a more carefull analysis we observe that we can parameterized the time complexity with the optimal value of the hyperbolicity and the distribution of the path lengths. That is,

Proposition 2. *Given a δ-hyperbolic graph G of diameter D and the sets $P[\ell]$ of pairs of vertices at distance ℓ from each other, the time complexity of lines 6-9 of Algorithm 1 is in*

$$O\left(\sum_{\ell_1=2\delta}^{D} |P[\ell_1]| \left(\frac{|P[\ell_1]| - 1}{2} + \sum_{\ell_2=\max\{1,\ 4\delta-\ell_1\}}^{\ell_1-1} |P[\ell_2]|\right)\right)$$

Proof. Let $P[\ell]$ be the list of pairs $(a, b) \in V \times V$ with $a < b$ such that $\text{dist}(a, b) = \ell$. For a δ-hyperbolic graph, Algorithm 1 will consider in the worst case all the triples $(\ell_1 + \ell_2, \ell_1, \ell_2)$

Algorithm 1 Hyperbolicity

Require: $G = (V, E)$ is a 2-connected graph.
Ensure: δ, the hyperbolicity of G (observe that $\delta = \mathrm{h}^* /2$).

```
1: Let P[ℓ] be the list of pairs (a, b) ∈ V × V with a < b such that dist(a, b) == ℓ
2: Sort P[ℓ] in increasing lexicographic order
3: Let SLL be the list of triples (ℓ1 + ℓ2, ℓ1, ℓ2) for 1 ≤ ℓ2 ≤ ℓ1 ≤ D
4: Sort SLL by decreasing lexicographic order
5: Let h* := 0
6: for all (S1, ℓ1, ℓ2) in SLL such that ℓ2 > h* do
7:    for all (a, b) ∈ P[ℓ1], if ℓ2 > h* do
8:       for all (c, d) ∈ P[ℓ2], if ℓ2 > h* do {When ℓ1 == ℓ2, we ensure that (a, b) < (c, d) in
         P[ℓ1] }
9:          h* := max{h*, S1 − max{dist(a, c) + dist(b, d), dist(a, d) + dist(b, c)}}
10:      end for
11:   end for
12: end for
13: return h*/2
```

such that $\ell_1 + \ell_2 \geq 4\delta$ and $D \geq \ell_1 \geq \ell_2 \geq 1$. Furthermore, when $\ell_1 = \ell_2$, line 8 ensures that pair $(a, b) < (c, d)$ in $P[\ell_1]$. Since we are not interested in the computation time of lines 1-4 (which is clearly dominated by the computation of the distances beween each pairs of vertices), the result follows. □

For instance, if the input graph is a $n \times n$ grid, with diameter $2n-2$ and hyperbolicity $\delta = n-1$ (so $\delta^* = 2n-2$), the value of the hyperbolicity will be obtained with the first considered 4-tuple. Lines 6-9 of Algorithm 1 will thus be executed in constant time. On the other hand, if the input graph is a $n \times 2$ grid, with diameter n and hyperbolicity $\delta = 1$ (so $\delta^* = 2$), almost all 4-tuples will be considered and so the running time will be in $O\left(\binom{n}{2} \cdot \left(\binom{n}{2} - 1\right)/2\right)$. Chordal graphs, which have hyperbolicity at most 1 [BKM01], are worst case instances for this algorithm since its running time increases with the gap between the diameter and the hyperbolicity.

Algorithm 1 requires to compute the distances bewteen all pairs of vertices (line 1), to sort pairs of vertices at equal distances in lexicographic order (line 2), and also some negligeable operations (lines 3-4). Therefore we have

Corollary 3. *Given a δ-hyperbolic graph G of diameter D and the sets $P[\ell]$ of pairs of vertices at distance ℓ from each other, the time complexity of Algorithm 1 is in*

$$O\left(\max\left\{n(n+m),\ n^2 \log n,\ \sum_{\ell_1=2\delta}^{D} |P[\ell_1]| \left(\frac{|P[\ell_1]|-1}{2} + \sum_{\ell_2=\max\{1,\ 4\delta-\ell_1\}}^{\ell_1-1} |P[\ell_2]|\right)\right\}\right)$$

Since the computational complexity of Algorithm 1 depends on the distance distribution of the graph and of the computed value of the hyperbolicity, we have not been able yet to obtain closed formula for its time complexity on the particular graph classes of interest for the EULER project. However, we will present in Section 3.1.3 some experimental results.

Observe that at any step of the algorithm, the value of ℓ_1 is an upper bound for the hyperbolicity of G. Therefore, for large graphs inducing an excessive running time (some instances may require weeks of computations), Algorithm 1 can be turned into an approximation algorithm with approximation ratio $\frac{\ell_1}{\mathrm{h}^*}$. It suffices for instance to stop the execution of the algorithm after a given time (for instance one hour) and to return the values h^* and ℓ_1. More precisely, we can insert one of the following test between line 6 and line 7 of the algorithm:

- "If computation time is larger than allowed computation time, then stop computations and return h^* and ℓ_1". We get $\frac{\mathrm{h}^*}{2} \leq \delta \leq \frac{\ell_1}{2}$;
- "If $\frac{\ell_1}{\mathrm{h}^*} \leq apx$, then stop computations and return $\frac{\mathrm{h}^*}{2}$." We get an approximation of the value δ of the hyperbolicity with proven approximation factor apx (i.e.., $\frac{\mathrm{h}^*}{2} \leq \delta \leq apx \cdot \frac{\mathrm{h}^*}{2}$);
- "If $\frac{\ell_1}{2} - \frac{\mathrm{h}^*}{2} \leq apx$, then stop computations and return $\frac{\mathrm{h}^*}{2}$." We get an approximation of the value δ of the hyperbolicity with proven additive approximation constant apx (i.e., $\frac{\mathrm{h}^*}{2} \leq \delta \leq \frac{\mathrm{h}^*}{2} + apx$).

As we show in the next section, the main part of the running time of the algorithm consists in closing the small gap between lower and upper bounds that are generally found very quickly. Therefore, depending on the expected result, it may be appropriate to use the above rules that may allow to save a lot of time while preserving a sufficient precision.

3.1.3 Experimental performances

In this section, we evaluate the performances of Algorithm 1 described above. This evaluation is done by comparing the running time of Algorithm 1 with the time of the naive algorithm $O\binom{n}{4}$. Our objective is to determine the topologies in which our algorithm is better than the naive one. Indeed, Algorithm 1 appears to be worse than the naive one for some pathological topologies such as grids. The good news are that its running-time outperforms the one of known algorithms in the case of Internet-like graphs. In particular, Algorithm 1 allows us to compute the hyperbolicity of the last CAIDA AS maps ($n = 25\,815$ for the largest biconnected component of the CAIDA map of September 2012) in two weeks.

We have reported in Table 3.1 the evolution of the computation time of Algorithm 1 on $n \times m$ grids such that $n * m = 2^2 * 3^2 * 4^2 = 576$ and $n * m = 2^2 * 3^2 * 5^2 = 900$. The running times are averages over 10 executions of the algorithm. The hyperbolicity of a $n \times m$ grid is $\delta_{n\times m} = \min\{n, m\} - 1$. As expected, the computation time on square grids is way smaller than for grids with sides of very different sizes. In Section 3.1.2, we said that rectangular grids are some of the worst case instances for Algorithm 1. To verify this claim, we have plotted in Figure 3.1 the relative number of visited 4-tuples with Algorithm 1 compared to the total number of 4-tuples in the graph. Recall that a 4-tuple may be visited up to three times by the algorithm. In these plots, we observe that as soon as the sides of the grids differ by a factor at least 6, then the computation time of Algorithm 1 is larger than the naive algorithm, but it is much faster when the sides of the grid differ by a factor less than 6.

Next, we have reported in Figure 3.2 the running time of Algorithm 1 on Barabasi-Albert (BA) graphs. We have generated graphs with number of nodes in the range $[1\,000..10\,000]$ and with different values of the degree k of newly added nodes. We have generated 100

n	m	δ	time
24	24	23	0.08
18	32	17	0.16
16	36	15	1.13
12	48	11	15.01
9	64	8	40.33
8	72	7	50.79
6	96	5	71.80
4	144	3	90.60
3	192	2	97.08
2	288	1	101.82

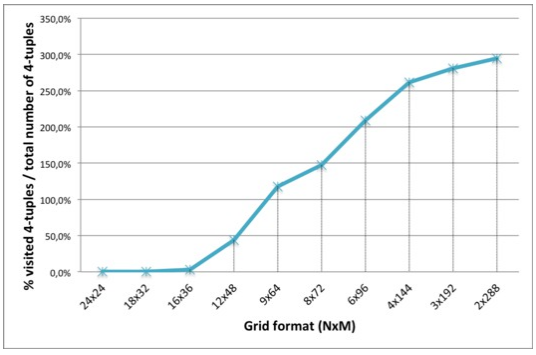
(a) $n * m = 576$

n	m	δ	time
30	30	29	0.16
25	36	24	0.18
20	45	19	5.98
18	50	17	23.10
15	60	14	86.94
12	75	11	201.71
10	90	9	297.54
9	100	8	349.51
6	150	5	496.64
5	180	4	537.42
4	225	3	574.37
3	300	2	599.19
2	450	1	622.15

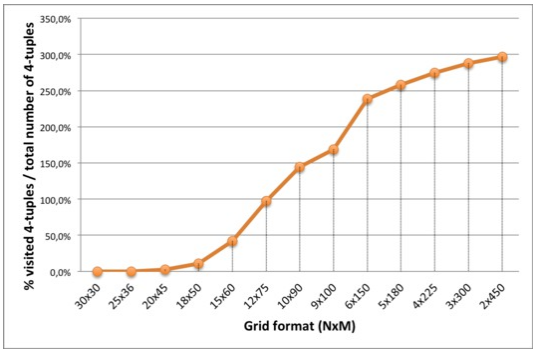
(b) $n * m = 900$

Table 3.1: Computation time in secondes of the hyperbolicity of $n \times m$ grids such that $n * m = 2^2 * 3^2 * 4^2 = 576$ (Table 3.1a) and $n * m = 2^2 * 3^2 * 5^2 = 900$ (Table 3.1b).

graphs for each couple number of nodes and parameter k, and reported the average values. Figure 3.2a reports the average values of the hyperbolicity of these graphs. We observe that the hyperbolicity of BA graphs decreases with the degree of newly added nodes. Figure 3.2b represents the average computation time of Algorithm 1 on these graphs. We observe a slow increase of the computation time with the increase of the parameter k. This slow increase was expected since the time complexity of both the all-pairs shortest path algorithm and the decomposition into bi-connected components algorithm depend on the number of edges of the graph and so on the parameter k. We also observe in Figure 3.2b some pics in the computation times for particular combinations of the number of nodes and the parameter k. However, at the time of writing this deliverable we have not been able to explain this surprizing behavior. Last, we have reported in Figure 3.2c the ratio of the average number of 4-tuples



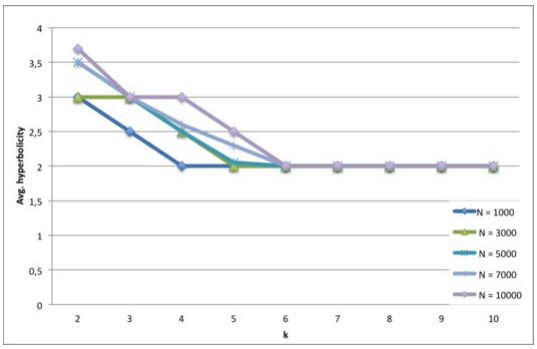
(a) $n * m = 576$



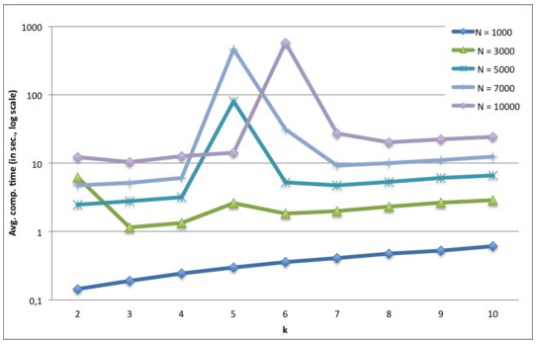
(b) $n * m = 900$

Figure 3.1: Relative performance of Algorithm 1 *vs.* the naive algorithm on $n \times m$ grids.

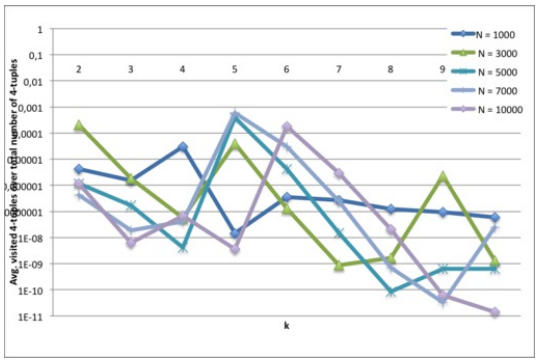
visited by Algorithm 1 over the total number of 4-tuples of the graphs. This highlights the drastic running time improvement of Algorithm 1 over the naive algorithm to compute the hyperbolicity of BA graphs. More precisely, Algorithm 1 is between 10^3 and 10^{11} times faster than the naive algorithm.



(a) Hyperbolicity of BA graphs



(b) Computation time



(c) Ratio visited/total number of 4-tuples

Figure 3.2: Computation time of the hyperbolicity of Barabasi-Albert graphs.

Last, we have reported in Table 3.2 the size of the largest bi-connected component (BCC), the hyperbolicity, the running time and the number of visited 4-tuples by Algorithm 1 when computing the hyperbolicity of some CAIDA maps. We have selected CAIDA maps with different values of the hyperbolicity to highlight the running time improvement of Algorithm 1. As expected with the time complexity expressed in Proposition 2, the running time improvement is correlated to the hyperbolicity of the CAIDA maps. Indeed, Algorithm 1 is more than 100 times faster than the naive algorithm for these maps. Nevertheless, the running times remains large. However, as explained in Section 3.1.2, at each step of the execution of Algorithm 1 we get proven lower and upper bounds for the hyperbolicity of the considered graph. Also, we have plotted in Figure 3.3 the time at which new lower and upper bounds are obtained when computing the hyperbolicity of the maps of Table 3.2. We observe that a few minutes of computations is sufficient for the lower bound to reach the optimal value but that the time to decrease the upper bounds to the optimal value could be very long. In fact, for maps with low hyperbolicity, almost all computation time is spent to prove the optimality of the lower bound.

AS map name	Largest BCC	Hyperbolicity	Time	Visited 4-tuples	Tot. 4-tuples
2004/01/05	10 424	2.5	336s	$3.8 \cdot 10^{10}$	$\simeq 5 \cdot 10^{14}$
2004/06/07	11 100	2.0	18h50m	$8.8 \cdot 10^{12}$	$\simeq 6.3 \cdot 10^{14}$
2005/09/05	12 957	3.0	58s	$1.2 \cdot 10^{8}$	$\simeq 1.2 \cdot 10^{15}$
2012/06/01	25 815	2.0	30d	$1.5 \cdot 10^{15}$	$\simeq 1.8 \cdot 10^{16}$

Table 3.2: Computation time of the hyperbolicity of some CAIDA maps.

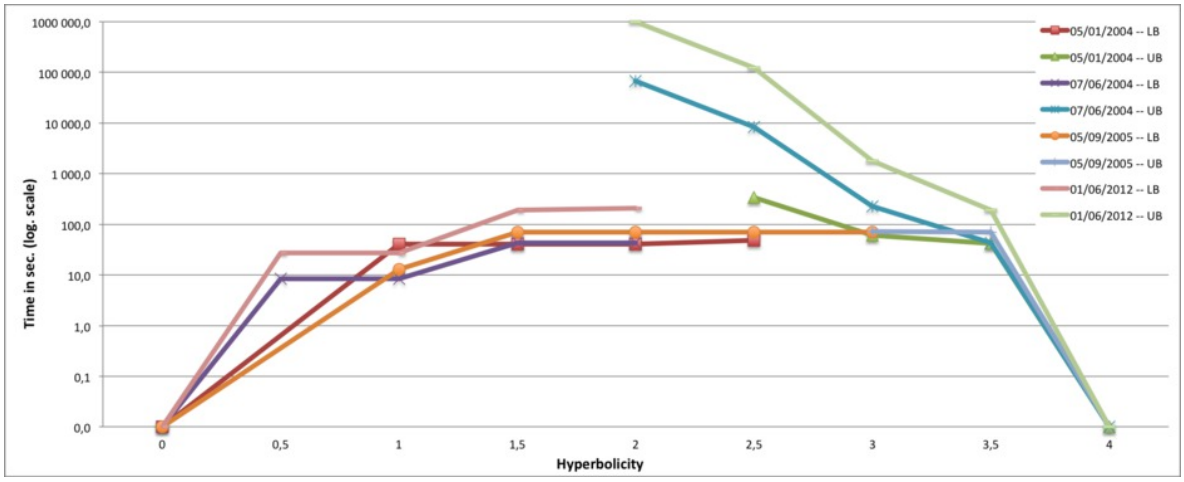


Figure 3.3: Time to reach lower and upper bounds of the hyperbolicity of CAIDA maps since 2004. Computation times for lower bounds are plotted from left to right, and from right to left for upper bounds.

3.2 Chordality and treewidth

This section extends the study on chordality and treewidth initiated during the first year of EULER and reported in Section 2.8 and 2.13 of deliverable D3.1. The main contribution of this section is the design of a new greedy algorithm that allows to decide whether a graph admits a large induced cycle or compute a particular tree-decomposition. In particular, this tree-decomposition can be used for the design of efficient compact routing schemes.

Even if the algorithm does not guaranty to compute the largest induced cycle of a graph (which is an NP-complete problem), it has been used in Chapter 5 to have first results on the chordality and the structure of large cycles of Internet-like graphs.

3.2.1 Objectives and Motivation

A parameter related to the hyperbolicity is the *chordality* of a graph. The *chordality* of a graph is the length of its longest induced (i.e., chordless) cycle. A graph with no induced cycle larger than k is called a k-chordal graph. By definition of the thin triangles, it is easy to see that bounded chordality implies bounded hyperbolicity. Some papers consider relations between both parameters [BC03, WZ11]. Several recent works on compact routing take advantage of such structural properties of large-scale networks for algorithm design (e.g., routing [KPBV09, CSTW09]). Indeed, Internet-like networks have a so-called high clustering coefficient (see e.g. [WS98, OP09]), leading to the existence of very few long chordless cycles, whereas their low (logarithmic) diameter implies a small hyperbolicity [dMSV11]. Unfortunately, contrary to the problem of computing the hyperbolicity of a graph, the problem of computing the chordality of a graph G is NP-complete since it may be related to computing a longest cycle in the graph obtained from G after subdividing all edges once. Finding the longest induced path is $W[2]$-complete [CF07] and the problem is Fixed Parameter Tractable in planar graphs [KK09]. It is coNP-hard to decide whether an n-node graph G is k-chordal for $k = \Theta(n)$ [Ueh99].

Another way to study tree-likeness of graphs is by *tree-decompositions*. Introduced by Robertson and Seymour [RS84], such decompositions play an important role in design of efficient algorithms. Roughly speaking, a tree-decomposition maps each vertex of a graph to a subtree of the *decomposition tree* in a way that the subtrees assigned to adjacent vertices intersect [RS84, Bod98]. The nodes of the decomposition tree are called *bags*, and the size of a bag is the number of vertices assigned to it (assigned subtrees intersect the bag). The *width* of a tree-decomposition is the maximum size over its bags, and the *treewidth* of a graph is the smallest width over its tree-decompositions. By using dynamic programming based on a tree-decomposition, many NP-hard problems have been shown to be linear time solvable for graph with bounded treewidth [CM93]. In particular, there are linear-time algorithms to compute an optimal tree-decomposition of a graph with bounded treewidth [Bod93, BK96]. However, from the practical point of view, this approach has several drawbacks. First, all above-mentioned algorithms are linear in the size of the graph but (at least) exponential in the treewidth. Moreover, due to the high clustering coefficient of large-scale networks, their treewidth is expected to be large [dMSV11]. Hence, to face these problems, it is important to focus on the structure of the bags of the tree-decomposition, instead of trying to minimize their size. For instance, several works study the diameter of the bags [DG07, Lok10].

It is NP-complete to decide whether the treewidth of a graph G is at most k [ACP87]. For chordal graphs, cographs [BM93], circular arc graphs [SSR94], chordal bipartite graphs [KK95]

and etc., the treewidth problem is polynomially solvable. Bodlaender and Thilikos proved that the treewidth of a k-chordal graph with maximum degree Δ is at most $\Delta(\Delta-1)^{k-3}$ which implies that treewidth is polynomially computable in the class of graphs with chordality and maximum degree bounded by constants [BT97]. They also proved that the treewidth problem is NP-complete for graphs with small maximum degree [BT97].

In Section 3.2.2, we describe a quadratic algorithm that, given a n-node graph G and an integer $k \geq 3$, either returns an induced cycle of length at least $k+1$ in G or computes a tree-decomposition of G with each bag having a dominating path of order $\leq k-1$. More precisely, each bag of our tree-decomposition contains a chordless path with at most $k-1$ vertices, such that any vertex in the bag is either in the path or adjacent to some vertex of the path. In the case when G admits such a decomposition, this ensures that G has treewidth at most $(k-1)(\Delta-1)+2$ (where Δ is the maximum degree), tree-length at most k and hyperbolicity at most $\lfloor 3k/2 \rfloor$. In particular, this shows that the treewidth of any k-chordal graph is upper-bounded by $O(k \cdot \Delta)$, improving the exponential bound of [BT97].

3.2.2 Structured Tree-decomposition

In this section, we present an algorithm that, given a n-node graph G and an integer $k \geq 3$, either returns an induced cycle of length at least $k+1$ in G or computes a tree-decomposition of G with interesting structural properties. These results have been presented in [KLNS12a, KLNS12b]. First, we need some definitions.

A *tree-decomposition* of a graph $G = (V, E)$ is a pair $(\{X_i | i \in I\}, T = (I, M))$, where T is a tree and $\{X_i | i \in I\}$ is a family of subsets, called bags, of vertices of G such that (1) $V = \cup_{i \in I} X_i$; (2) $\forall \{uv\} \in E$ there is $i \in I$ such that $u, v \in X_i$; and (3) $\forall v \in V$, $\{i \in I | v \in X_i\}$ induces a (connected) subtree of T. The *width* of a tree-decomposition is the size (minus 1) of its largest bag and its *ℓ-width* is the maximum diameter of the subgraphs induced by the bags. The *treewidth* denoted by $\mathrm{tw}(G)$, resp., *tree-length* denoted by $\mathrm{tl}(G)$, of a graph G is the minimum width, resp., ℓ-width, over all possible tree-decompositions of G [RS84, DG07].

Let $k \geq 2$. A k-caterpillar is a graph that has a dominating set, called *backbone*, which induces a chordless path of order at most $k-1$. That is, any vertex of a k-caterpillar either belongs to the backbone or is adjacent to a vertex of the backbone. A tree-decomposition is said to be *k-good* if each of its bags induces a k-caterpillar.

Theorem 4. *There is a $O(m^2)$-algorithm that takes a m-edge graph G and an integer $k \geq 3$ as inputs and:*

- *either returns an induced cycle of length at least $k+1$;*
- *or returns a k-good tree-decomposition of G;*

Proof. The proof is by induction on $|V(G)| = n$. We prove that either we find an induced cycle larger than k, or for any chordless path $P = \{v_1, \ldots, v_i\}$ with $i \leq k-1$, there is a k-good tree-decomposition for G with one bag containing $\Gamma_G[P]$. Obviously, it is true if $|V(G)| = 1$. Now we assume that it is true for any graph G with n' nodes, $1 \leq n' < n$, and we show it remains true for n-node graphs.

Let G be a connected n-node graph, $n > 1$. Let $P = \{v_1, \ldots, v_i\}$ be any chordless path with $i \leq k-1$ and let $N = \Gamma_G[P]$, $\Gamma_j = \Gamma_G[v_j]$ for $j = 1, \ldots, i$ and $G' = G \setminus N$. There are three cases to be considered:

Case 1. $G' = \emptyset$. In this case, we have $G = N$. The desired tree-decomposition consists of one node, corresponding to the bag N.

Case 2. G' is disconnected. Let $C_1, \ldots, C_r$, $r \geq 2$, be the connected components of G' For any $j \leq r$, let G_j be the graph induced by $C_j \cup N$. Note that any induced cycle in G_j, $j \leq r$, is an induced cycle in G. By the induction hypothesis, either there is an induced cycle $\mathcal{C}$ larger than k in G_j, then $\mathcal{C}$ is also an induced cycle larger than k in G, or our algorithm computes a k-good tree-decomposition TD_j of G_j with one bag X_j containing N. To obtain the k-good tree-decomposition of G, we combine the TD_j's, $j \leq r$, by adding a bag $X = N$ adjacent to all the bags X_j for $j = 1, \ldots, r$. It is easy to see that this tree-decomposition satisfies our requirements.

Case 3. G' is connected. We consider the order of the path $P = \{v_1, \ldots, v_i\}$. In the following proof, first we prove that if the order of path P, $i = k - 1$, then we can find either an induced cycle larger than k or the required tree-decomposition for G. Subsequently, we prove it is also true for path with length $i < k - 1$ by reversed induction on i. More precisely, if $i < k - 1$, either we find directly the desired cycle or tree-decomposition, or we show that there exists a vertex v_{i+1} such that $P' = P \cup \{v_{i+1}\}$ is a chordless path with order $i + 1$. By reverse induction on i we can find either an induced cycle larger than k or a k-good tree-decomposition of G with one bag containing $\Gamma_G[P'] \supseteq \Gamma_G[P]$.

1. If $i = k - 1$, then we consider the following two cases.
   - Assume first that there is $u \in \Gamma_G(P) \cup \{v_1, v_i\}$ (in particular, $u \notin P \setminus \{v_1, v_i\}$) such that $\Gamma_G(u) \subseteq \Gamma_G[P \setminus \{u\}]$. Let $\tilde{G} = G \setminus u$. Then $\tilde{G}$ is a graph with $n' = n - 1$ vertices. By the induction hypothesis on $n' < n$, the algorithm either finds an induced cycle larger than k in $\tilde{G}$, then it is also the one in G; Otherwise our algorithm computes a k-good tree-decomposition $\widetilde{TD}$ of $\tilde{G}$ with one bag $\tilde{X}$ containing $\Gamma_{\tilde{G}}[P \setminus \{u\}]$. To obtain the required tree-decomposition of G, we just add vertex u into the bag $\tilde{X}$. The tree-decomposition is still k-good.
   - Otherwise, there exist two distinct vertices $v_0 \in \Gamma_G(v_1)$ and $v_{i+1} \in \Gamma_G(v_i)$ and there are vertices $u_1, u_2 \in V(G')$ (possibly $u_1 = u_2$) such that $\{v_0, u_1\} \in E(G)$ and $\{v_{i+1}, u_2\} \in E(G)$. If $\{v_0, v_{i+1}\} \in E(G)$, $P \cup \{v_0, v_{i+1}\}$ is an induced cycle with $k + 1$ vertices. Otherwise, let Q be a shortest path between u_1 and u_2 in G' (Q exists since G' is connected). So $P \cup \{v_{i+1}, u_2\} \cup Q \cup \{u_1, v_0\}$ is an induced cycle with at least $k + 1$ vertices in G.
2. If $i < k - 1$, we proceed by reverse induction on i. Namely, assume that, for any chordless path Q with $i + 1$ vertices, our algorithm either finds an induced cycle larger than k in G or computes a k-good tree-decomposition of G with one bag containing $\Gamma[Q]$. Note that the initialization of the induction holds for $i = k - 1$ as described in case (b). We show it still holds for a chordless path with i vertices. We consider the following two cases.
   - Either there is $u \in \Gamma_G(P) \cup \{v_1, v_i\}$ (in particular, $u \notin P \setminus \{v_1, v_i\}$) such that $\Gamma_G(u) \subseteq \Gamma_G[P \setminus \{u\}]$. That is, we are in the same case as the first item of (a). We proceed as above and the result holds by induction on n.
   - Or there is $w \in \Gamma_G(v_1) \cup \Gamma_G(v_i) \setminus P$ such that $P \cup \{w\}$ is chordless (i.e., w is a neighbor of v_1 or v_i but not both). Therefore, we apply the induction

hypothesis (on i) on $P' = P \cup \{w\}$. By the assumption on i, either our algorithm returns an induced cycle larger than k or it computes a k-good tree-decomposition of G with one bag containing $\Gamma_G[P'] \supseteq \Gamma_G[P]$.

To conclude, we describe the algorithm and study its complexity. Let G be a m-edge n-node graph with maximum degree Δ. Roughly, the algorithm proceeds by steps. At each step, one vertex is considered and the step takes $O(m)$ time. We prove that at each step (but the initial step), at least one edge will be *considered* and that all edges are considered at most once. This implies a time-complexity of $O(m^2)$ for the algorithm.

The algorithm starts from an arbitrary vertex $v \in V(G)$ and computes the connected components $C_1, \cdots, C_j$ of $G \setminus \Gamma[v]$ ($j \geq 1$) in time $O(m)$. We start with the k-good tree-decomposition for the induced graph of $\Gamma[v]$ in G that consists of a bag $B = \Gamma[v]$ adjacent to, for any $i \leq j$, each bag $B_i = \{v\} \cup \{w \in \Gamma(v) \ : \ \Gamma(w) \cap C_i \neq \emptyset\}$. This takes time $O(m)$.

Now, at some step of the strategy, assume that we have built a k-good tree-decomposition $(T, \mathcal{X})$ of a connected subgraph G_0 of G. Let $C_1, \cdots, C_j$ ($j \leq 1$) be the connected components of $G \setminus G_0$, and, for any $i \leq j$, let S_i be the set of the vertices of G_0 that are adjacent to some vertex of C_i. Assume finally that, for any $i \leq j$, there is a leaf bag $B_i \supset S_i$ of $(T, \mathcal{X})$ where $P_i = B_i \setminus S_i$ is a chordless path dominating B_i.

For any $e \in E(G)$, we say that $e = \{x, y\}$ is *alive* if there is $i \leq j$ such that $x \in S_i \cup C_i$ and $y \in C_i$. Note that, if an edge is alive, such an integer i is unique. An edge that is not alive is said *dead*. Note also that, after the initial step, all edges in the bag B are dead and other edges are alive.

The next step consists of the following. Choose any $i \leq j$ and let w be any vertex of S_i such that $Q = P_i \cup \{w\}$ is a chodless path. Note that by definition of S_i, there is at least one edge from w to C_i and that such an edge is alive before this step. We add the bag $B' = Q \cup B_i \cup (\Gamma(w) \cap C_i)$ adjacent to B_i. If Q is larger than k, by the above proof, the algorithm finds a large cycle. Otherwise, the connected components $C'_1, \cdots, C'_r$ of $C_i \cup B_i \setminus B'$ are computed in time $O(m)$. Let S'_h, $h \leq r$, be the subset of the vertices of S_i that are adjacent to some vertex in C'_h, and let Q_h be the smallest subpath of Q dominating S'_h. Computing the sets $S'_1, \cdots, S'_r$ only requires a time $O(m)$ since we have only to check the edges in B'. For any $h \leq r$, add a bag $B'_h = Q_h \cup S'_h$ adjacent to B'.

It is easy to check that this algorithm follows the above proof and that it eventually computes the desired tree-decomposition or returns a large cycle.

To conclude, it is easy to check that the set of edges alive after one step is contained in the set of edges alive before this step, and that, at each step at least one edge (the one(s) from w to S_i) become dead. Therefore, at each step, the number of alive edges strictly decreases and the algorithm terminates when there are no more. Since each step takes time $O(m)$ and there are at most m steps, the result follows. □

From the above theorem, it is easy to get the following corollaries.

Theorem 5. *Let G be a graph that admits a k-good tree-decomposition. Then* $\mathrm{tw}(G) \leq (k-1)(\Delta-1)+2$ *where Δ is its maximum degree, and* $\mathrm{tl}(G) \leq k$.

Proof. It directly follows the fact that, in a k-good tree-decomposition, each bag has a dominating path with $< k$ vertices. □

Theorem 6. *Any graph G that admits a k-good tree-decomposition has hyperbolicity at most* $\lfloor 3k/2 \rfloor$.

Proof. Let $G = (V, E)$ be a graph that admits a k-good tree-decomposition $(\{X_i | i \in I\}, T = (I, M))$. Let T be rooted at bag X_0, $0 \in I$. For any $u, v \in V$, let us denote the distance between u and v by $\text{dist}(u, v)$ in G. By definition of a k-good decomposition, for any $i \in I$ and for any $u, v \in X_i$, $\text{dist}(u, v) \leq k$.

Let $x, y, z \in V$ and let P_1, P_2, P_3 be any three shortest paths in G between x and y, y and z, x and z respectively. Let $u \in V(P_1)$. To prove the Theorem, we show that there is $v \in V(P_2) \cup V(P_3)$ such that $\text{dist}(u, v) \leq \lfloor 3k/2 \rfloor$.

First, let us assume that there is $i \in I$ such that $u \in X_i$ and there is $v \in (V(P_2) \cup V(P_3)) \cap X_i \neq \emptyset$. In that case, $\text{dist}(u, v) \leq k$ and the result holds.

Otherwise, let T_u be the subtree of T induced by $\{i \in I \ : \ u \in X_i\}$. Similarly, let T_x be the subtree of T induced by $\{i \in I \ : \ x \in X_i\}$ and T_y be the subtree of T induced by $\{i \in I \ : \ y \in X_i\}$. Let P be the path in T between T_x and T_y. Note that P may be empty if $V(T_x) \cap V(T_y) \neq \emptyset$. Let $j \in V(T_x) \cup V(T_y) \cup V(P)$ that is closest to T_u in T. Note that $j \notin V(T_u)$ because otherwise we would be in the first case. Note also that either X_j is a separator between x and u or $x \in X_j$, and either X_j is a separator between y and u or $y \in X_j$. Let P_{xu} and P_{uy} be the subpaths of P_1 from x to u and from u to y respectively. By remark above, there exist vertices $w \in V(P_{xu}) \cap X_j$ and $t \in V(P_{uy}) \cap X_j$. Possibly, $w = t$. Then $\text{dist}(w, u) + \text{dist}(u, t) = \text{dist}(w, t)$ because P_1 is a shortest path, therefore, $\text{dist}(w, u) + \text{dist}(u, t) = \text{dist}(w, t) \leq k$. So there is $\ell \in X_j$ with $\text{dist}(u, \ell) \leq \lfloor k/2 \rfloor$.

Finally, let us show that there is $h \in V(P_2 \cup P_3) \cap X_j$. If $x \in X_j$ or $y \in X_j$, it is obvious. Otherwise, this is because X_j separates x and y in G, and therefore, z cannot be in both the component of $G \setminus X_j$ containing x and of the one containing y, therefore, one of the paths P_2 or P_3 should path trough X_j.

To conclude, $\text{dist}(u, h) \leq \text{dist}(u, \ell) + \text{dist}(\ell, h) \leq \lfloor k/2 \rfloor + k \leq \lfloor 3k/2 \rfloor$. □

Corollary 7. *Any k-chordal graph G with maximum degree Δ has treewidth at most $(k-1)(\Delta-1)+2$, tree-length at most k and hyperbolicity at most $\lfloor 3k/2 \rfloor$.*

Proof. By definition of k-chordal graph and Theorem 4, any k-chordal graph admits a k-good tree-decomposition. The result follows Theorems 5 and 6. □

Corollary 8. *There is an algorithm that, given a m-edge graph G and $k \geq 3$, states that either G has chordality at least $k+1$ or G has hyperbolicity at most $\lfloor 3k/2 \rfloor$, in time $O(m^2)$.*

We propose a compact routing scheme for any n-node graph G that admit a k-good tree-decomposition (this includes k-chordal graphs). Δ denotes the maximum degree of G and, for any $v \in V(G)$, $\deg_v$ is its degree.

Theorem 9. *For any n-node m-edge graph G with maximum degree Δ and admitting a k-good tree-decomposition, there is a labelled routing scheme $\mathcal{R}$ with the following properties. $\mathcal{R}$ uses addresses of size $O(\log n)$ bits, port-numbers of size $O(\log \Delta)$ bits and routing tables of size $O(\max\{k \cdot \log \Delta, \log n\})$ bits. The routing tables, addresses and port-numbers can be computed in time $O(m^2)$. Except the address of the destination (not modifiable), the header of a message contains $O(k \cdot \log \Delta)$ modifiable bits. The header and next hop is computed in time $O(1)$ at each step of the routing. Finally, the additive stretch is $\leq k(2\lceil \log \Delta \rceil + 5) - 3$.*

Chapter 4

Towards a Bipartite Graph Modeling of the Internet Topology

4.1 Objectives and Motivation

Modeling the properties of the Internet topology aims at generating large scale artificial IP networks that mimic properties of real ones for simulation purposes. As shown in Chapter 1 of D3.1, current models typically consider the Internet as a simple graph where edges are point-to-point connections between routers. This approach does not take into account point-to-multipoint connections that exist at lower layers in the network, e.g. layer-2 clouds, such as Ethernet switches or Token Ring. Instead, such physical point-to-multipoint connections are modeled as several logical IP level point-to-point connections.

In this work, we rely on recent development in topology discovery based on IGMP probing that allows for revealing part of the network's layer-2 structure. We take advantage of this additional knowledge for proposing an Internet model based on bipartite graphs considering both point-to-point and point-to-multipoint connections. The model remains simple: it only takes as input the node degree sequence for both layer-2 and layer-3 nodes, randomly generates a bipartite graph respecting those distributions, and then derives the corresponding layer-3 topology. We show that, despite the simplicity of our model, realistic network properties, such as high local density, emerge naturally. This is in contrast with the now common belief that such properties can only appear with more intricate models or if explicitly injected in random models. Besides, we also provide evidences of how the analysis performed at the bipartite level might shed light on important properties of the real network structure.

4.2 Related Work

Improving our understanding of the Internet topology structure is extremely important. It has much impact on the ability to provision and manage IP networks and enhance their reliability and efficiency. It also allows for designing effective network protocols matching the specific requirements of a large panel of applications. Assessing the quality of a network or protocol design involves theoretical studies and simulations conducted on artificial graphs obtained from models of the Internet topology. Many efforts have been made in modeling Internet [HIM+08, PSV04], from very simple models [ER59] to more complex ones based on latest developments in Internet topology discovery and modeling [ALWD05, WL10, MDBP10].

However, modeling the Internet remains a challenging task because of its heterogeneity and dynamics [MOVL09, HUM+08].

Usually, the Internet is depicted as a simple graph where vertices represent, depending on the Internet topology view, IP interfaces, routers, or autonomous systems (ASes) and edges stand for direct connections between those vertices. In particular, when considering the router level of the Internet, edges usually represent point-to-point links between routers, i.e., IP hops. However, the Internet is actually made of fundamentally different kinds of nodes at layer-2 (L2), which induce its layer-3 (L3) structure: routers might be connected through L2 devices like Ethernet switches, IXP, sub-networks, ... and a L3 link between two routers goes therefore through a L2 device. Such point-to-multipoint connections, induced from the L2 view of the network, are generally invisible because they are challenging to discover when using common active topology discovery techniques based on traceroute [DF07].

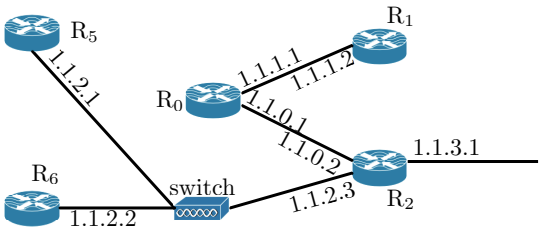
Being able to map Internet topologies exhibiting the two layers of connection would open new perspectives in Internet modeling and topology generation. Indeed, one could then model the Internet topology as a *bipartite graph*, i.e., a graph in which vertices can be divided into two disjoint sets, $\top$ (e.g., Ethernet switches) and $\bot$ (e.g., routers), such that every edge connects a vertex in $\top$ to one in $\bot$. Bipartite graphs are a fundamental object in computer science and, as such, are widely studied [WS98, New01, IMF04]. A key operation over bipartite graphs is the *projection* that transforms the bipartite structure into a simple graph where a link between two routers in $\bot$ exists if they are linked to a same L2 device in $\top$. Interestingly, the projection fits exactly the inference of the Internet L3 topology from its L2 topology. This makes bipartite graphs (and their projection) an appealing approach for Internet topology modeling with a L2/L3 view.

Fortunately, a recent advance in Internet topology discovery through IGMP probing [MVdSD+09] has offered an opportunity to better characterize the nature of IP connections (point-to-point or point-to-multipoint). With a single IGMP probe, one can obtain all local multicast interfaces and neighbors of a multicast router, as well as its multicast connections through L2 multi-access networks. This latter feature provides point-to-multipoint connections between L3 devices. Considering a map resulting from IGMP probing [MDBP10], we are able to construct a bipartite graph where vertices are of the two types.

It is worth noticing that IGMP data allows one to easily discover the *actual* bipartite shape of the Internet induced by the interactions between L2 and L3 devices. Generally, such a bipartite structure is artificially generated to capture some clustering properties in flat network ground measurements [GL06]. This is not our approach here since we stick to the *existing* bipartite structure detected by the measurement tool. Note also that, although the data obtained by mrinfo could be partial and/or biaised, the present work is indenpendant from the quality of the data. The problem of improving the measurement tools is different from the one of identifying relevant properties able to exploit the features observed in the data, which is what we focus on in the present chapter.

4.3 Methodology

In this work, we step into the breach opened by the L2 devices inference and describe the first bipartite model of the Internet topology. Our model has the strong advantage of being "simple",i.e., it is a random-based model that does not require injecting several constraints. As input, we only consider the node degree distribution of both L2 and L3 devices



1.1.0.2 [version 12.4]
- 1.1.0.2 → 1.1.0.1 [1/0/pim/querier]
- **1.1.2.3** → **1.1.2.1** [1/0/pim/querier]
- **1.1.2.3** → **1.1.2.2** [1/0/pim/querier]
- 1.1.3.1 → 0.0.0.0 [1/0/pim/leaf]

Figure 4.1: `mrinfo` example

for generating the random bipartite graph and, then, project this structure into a simple graph. To this regard, our model can be seen as an extension of the standard *configuration model* [ACL00, NWS01, XL07] using two distinct degree sequences.

Note that our model does not aim at identifying network construction mechanisms as would do a structural model [ZCB96, ZCD97], a preferential attachment model [BA99, AB00], or an engineer-oriented model [ALWD05]. If such approaches may bring interesting knowledge on the networks emergence (although they are often criticized, see [WAD09]), they are usually not well suited for formal analysis and they tend to enforce specific properties in the generated graphs. This is why we rather follow the tradition of random models [ER59, ACL00, Wax88].

In order to assess the relevance of our model, we perform two different analyses. First, we demonstrate that the resulting projected graphs have similar behaviors than actual ones, specially regarding metrics that were not injected in the model, such as the local density or the degree correlations for instance. Indeed, it is worth to notice that standard models usually only reproduce properties they focus on but are unable to cope with all other features. On the contrary, our random model is able to reproduce relatively accurately properties that were not intentionally given as input. In order to emphasize this point and better evaluate the improvements brought by our bipartite model, we confront our results with random graphs directly generated with the configuration model from which our model derives. It results that, as expected, the configuration model is unable to cope with other properties than the degree distribution. Second, we evaluate different metrics on the bipartite structure itself and give evidences of the relationship between the observed bipartite properties and the projected ones. Our analyses show that, although not perfectly reproducing the real data, our model succeeds in capturing most of its properties and provides mathematical tools for explaining properties of the L3 structure from analyzes of the bipartite structure.

Analyzing the limitations derived from our first study, we also explore possible extensions of our model. First, we investigate the interest of taking into account the correlation between point-to-point and point-to-multi-point connections of the routers. Second, using statistical metrics defined at the bipartite level, we identify strong redundant patterns and propose a way to cope with such a structural property in the model. Indeed the overlapping between L2-L3 connections is frequent in real IP networks since redundancy is a key feature to increase the reachability between networking devices. Such resiliency patterns cannot be accurately revealed using a L3 view only.

To conclude this section on the methodology adopted in our work, we introduce the required background on `mrinfo`, a tool allowing for silently revealing all multicast IP addresses of a router, as well as its connections towards other routers and L2 devices.

`mrinfo` messages use the Internet Group Management Protocol (IGMP [Dee89]). IGMP

was initially designed to allow hosts to report their active multicast groups to a multicast router on their LAN. However, the Distance Vector Multicast Routing Protocol, DVMRP, has defined two special types of IGMP messages that can be used to monitor routers [Pus03]. Although current IPv4 multicast routers do not use DVMRP anymore, they still support these special IGMP messages. Upon reception of an IGMP `ASK_NEIGHBORS` message, an IPv4 multicast router replies by sending an IGMP `NEIGHBORS_REPLY` message that lists all its multicast enabled adjacencies. Figure 4.1 shows an example of the usage of `mrinfo` to query the router R_2 (`1.1.0.2` is the replying interface of R_2). `mrinfo` reports that this router is directly connected to R_0 (through interface `1.1.0.1`) via a layer-3 (L3) point-to-point link. One can also notice that R_2 is connected to routers R_5 and R_6 through a layer-2 (L2) network (labeled "switch" in Figure 4.1) because interface `1.1.2.3` appears twice in the `mrinfo` reply (see bold text in Figure 4.1). Finally, `mrinfo` reports that interface `1.1.3.1` has no multicast router neighbor (the right IP address is equal to `0.0.0.0`). All this topological information is obtained by sending a single IGMP message. `mrinfo` provides similar information than a `show` command dedicated to the multicast routing plan.

In the analysis provided in this report, the inference of *L2 networks* is critical. In our context, by L2 network, we mean a technology allowing a router to transmit IP packets to several other IP routers through the same interface, i.e., a multi-access network. One often distinguishes between *Non Broadcast Multiple Access* (NBMA) networks (e.g., ATM, Frame Relay, X25), and *broadcast networks* (BN) such as most LAN networks (e.g., Ethernet, token ring, FDDI).

These two kinds of networks behave differently as far as IP multicast is concerned. In particular, when using *Protocol Independent Multicast* (PIM) as a routing protocol in a BN, only one of the PIM IP neighbors is elected as the *querier* [Fen97]. Moreover, in common BN such as L2 Ethernet switches, the IP view around the L2 device should exhibits symmetric properties and reveals that IP interfaces involved in this symmetric point-to-multipoint connection are allocated within a tight subnet prefix. In NBMA networks, IP packets are usually transmitted via circuits that behave as a collection of point-to-point or point-to-multipoint connections. Such properties can be easily revealed within the `mrinfo` range: in this work, we focus on most common BN such as L2 Ethernet switches. These represent the vast majority of multi-access networks in the `mrinfo` dataset that we use.

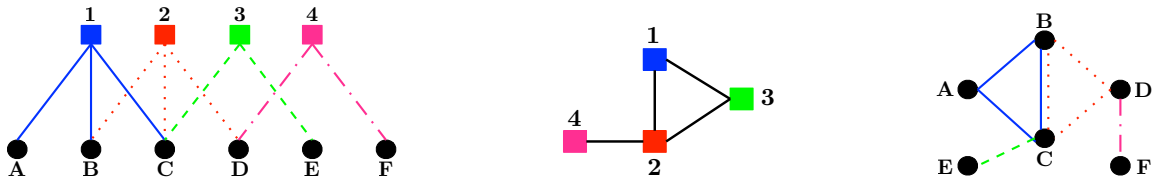
In this work, we consider the publicly available `mrinfo` dataset [Pan]. It is worth to notice that obtaining L2 and L3 topologies is also possible using Gunes and Sarac's subnet inference technique [GS07]. We believe that the framework provided in this document can also be applied on Gunes and Sarac's dataset.

4.4 Bipartite Graphs and our Model

We first give a theoretical presentation of bipartite graphs (Section 4.4.1) that will be used for modeling topology data collected with `mrinfo`. Then, we explain how we model a subset of the router level topology as a bipartite graph (Section 4.4.2).

4.4.1 Bipartite Graphs

A *bipartite graph* is a triplet $G = (\top, \bot, E)$, where $\top$ is the set of *top* nodes, $\bot$ the set of *bottom* nodes, and $E \subseteq \top \times \bot$ the set of links. Compared to standard graphs, nodes in a bipartite graph are in two disjoint sets, and the links are always between a node in one set



(a) bipartite graph (b) $\top$-proj. (c) $\bot$-proj.

Figure 4.2: Example of bipartite graph and its $\{\top, \bot\}$-projections

and a node in the other set. An example of bipartite graph is given in Figure 4.2a, where $\top$ nodes are depicted by squares and $\bot$ nodes by circles.

The $\bot$-projection of G is the graph $G_\bot = (\bot, E_\bot)$ where two nodes (of $\bot$) are linked together if they have at least one neighbor in common (in $\top$) in G: $E_\bot = \{(u, v), \exists x \in \top : (u, x) \in E \text{ and } (v, x) \in E\}$. The $\top$-projection is defined dually. Both projections are illustrated in Figure 4.2b and 4.2c.

Classical Analysis over Projections. In order to analyze this bipartite structure, it is natural to transform a bipartite graph into one of its projection in order to compute standard metrics defined for graphs. Let us recall briefly here those metrics and the usual properties shared by real-world networks [WS98].

Let $G = (V, E)$ be the (projected) graph. We denote by $\Gamma(v)$ the set of neighbors of $v \in V$: $\Gamma(v) = \{u \in V,\ (u, v) \in E\}$ and by $\deg(v)$ its degree: $\deg(v) = |\Gamma(v)|$.

The usual statistics used to characterize such a graph involve its size ($n = |V|$), its number of links ($m = |E|$), its highest degree ($\Delta = \max_v \deg(v)$), and its average degree ($k = \frac{2.m}{n}$). Over those notions, one can also study the density $\rho = \frac{2.m}{n.(n-1)}$ that is usually small as real networks happen to be very sparse. Indeed the probability that a link exists between two randomly selected nodes is generally very small.

On the contrary, two nodes sharing a common neighbor have usually a high probability to be linked. This property is often referred to as the *local density* and is generally captured by the *clustering coefficient* and the *transitivity ratio* [WS98, SW05a, SW05b]. The first one computes, for every node $v \in V$, the probability that two of its neighbors are linked together. This is denoted by $\texttt{cc}(v) = \frac{\Delta(v)}{\vee(v)}$ where $\Delta(v)$ is the number of triangles (sets of three nodes with three links) to which v belongs and $\vee(v) = \frac{\deg(v).(\deg(v)-1)}{2}$ the number of pairs of neighbors of v. The clustering coefficient of the graph is the average value $\texttt{cc} = \frac{\sum_v \texttt{cc}(v)}{n}$.

The second coefficient, the transitivity ratio, provides a more direct computation of the property over the whole graph. Let $\Delta = \sum_v \Delta(v)$ and $\vee = \sum_v \vee(v)$, then $\texttt{tr} = \frac{3.\Delta}{\vee}$ is defined as the transitivity ratio of G.

A classical observation is that those two quantities are high, at least compared to the density. In other words, if one selects a random pair of links with an extremity in common (transitivity ratio) or a random node and two of its neighbors (clustering coefficient), then the probability that the third possible link exists is high.

Specific Metrics for Bipartite Graphs. The metrics defined in previous paragraph have the advantage to be well understood and allow for immediate analysis of the flat topology.

On the other hand, the required projection leads to a loss of information. It is thus necessary to define extensions of those metrics on the bipartite structure itself.

From a bipartite graph $G = (\top, \bot, E)$ and for each top node $v \in \top$, we denote by $\Gamma_\top(v)$ the set of bottom neighbors of v: $\Gamma_\top(v) = \{u \in \bot,\ (u, v) \in E\}$ and by $\Gamma_\bot \Gamma_\top(v)$ the set of top neighbors of v: $\Gamma_\bot \Gamma_\top(v) = \{u \neq v \in \top, \exists x \in \bot : (u, x) \in E \text{ and } (v, x) \in E\}$. We use similar notations for the bottom nodes $\Gamma_\bot(v)$ and $\Gamma_\top \Gamma_\bot(v)$. For instance, on Figure 4.2a, $\Gamma_\top(1) = \{A, B, C\}$ and $\Gamma_\bot \Gamma_\top(1) = \{2, 3\}$. Similarly, $\Gamma_\bot(C) = \{1, 2, 3\}$ and $\Gamma_\top \Gamma_\bot(C) = \{A, B, D, E\}$.

Let $n_\top$ (respectively $n_\bot$) be the number of $\top$ (respectively $\bot$) nodes and $m_{\texttt{bip}}$ be the number of bipartite links. We denote by $k_\top$ (respectively $k_\bot$) the average degree of $\top$ (respectively $\bot$) nodes and $\rho_{\texttt{bip}} = \frac{m_{\texttt{bip}}}{n_\top . n_\bot}$ the density of the bipartite graph. On Figure 4.2a, $n_\top = 4$, $n_\bot = 6$, $k_\top = 2.5$, $k_\bot = 1.6$, $m_{\texttt{bip}} = 10$, and $\rho_{\texttt{bip}} = 0.42$.

Those statistics are natural extensions of graph metrics. However, for the *local density*, there is no standard variant since, by definition, there is no triangle in a bipartite graph. As suggested by Latapy et al. [LMDV08], we will rely on the following coefficient that tends to capture the overlapping between the neighborhood of two nodes of $\top$

$$\texttt{cc}_\top(u, v) = \frac{|\Gamma_\top(u) \cap \Gamma_\top(v)|}{|\Gamma_\top(u) \cup \Gamma_\top(v)|}. \tag{4.1}$$

This coefficient is interesting as it captures the relative overlap between neighborhoods of top nodes, i.e., $\texttt{cc}_\top(u, v)$ is equal to 1 if the neighborhood of u and v intersects exactly, to 0 if they do not share any neighbor. If we apply the overlapping coefficient on nodes 1 and 2 in Figure 4.2a, we have $\texttt{cc}_\top(1, 2) = \frac{|\{A,B,C\} \cap \{B,C,D\}|}{|\{A,B,C\} \cup \{B,C,D\}|} = 0.5$.

From this coefficient, it becomes natural to define the clustering coefficient related to a specific $\top$ node v. This is given by

$$\texttt{cc}_\top(v) = \frac{\sum_{u \in \Gamma_\bot \Gamma_\top(v)} \texttt{cc}_\top(u, v)}{|\Gamma_\bot \Gamma_\top(v)|}. \tag{4.2}$$

Applied on node 1 of Figure 4.2a, it gives $\texttt{cc}_\top(1) = 0.375$. This coefficient enables to study the distribution of this property over the top nodes as well as its correlation with the degree or other properties. Then, one can naturally compute the *bipartite top clustering coefficient* $\texttt{cc}_\top$ of G as the average value of $\texttt{cc}_\top(v)$ over all the nodes v of $\top$. More formally

$$\texttt{cc}_\top(G) = \frac{1}{|\top|} \sum_{v \in \top} \texttt{cc}_\top(v). \tag{4.3}$$

Following those definitions, we can derive the dual $\texttt{cc}_\bot(G)$ bottom clustering coefficient of G which finally leads to the global *clustering coefficient* of G defined by:

$$\texttt{cc}_{\texttt{bip}}(G) = \frac{\texttt{n}_\top \texttt{cc}_\top(G) + \texttt{n}_\bot \texttt{cc}_\bot(G)}{\texttt{n}_\top + \texttt{n}_\bot}. \tag{4.4}$$

4.4.2 Model

Our methodology is sketched in Figure 4.3 and Figure 4.4. We first start by removing loops[1] in our dataset ("cleaning step" on Figure 4.4). We then have a dataset that contains a set of

[1]It affects less than 0.3% of the links on average.

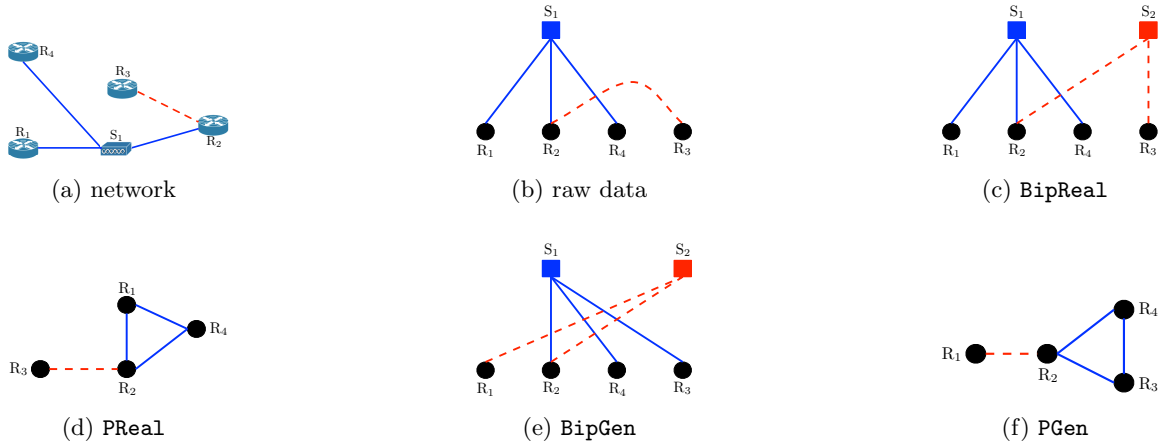


Figure 4.3: Example of transformation between network, raw data, `BipReal`, `PReal`, `BipGen`, and `PGen`

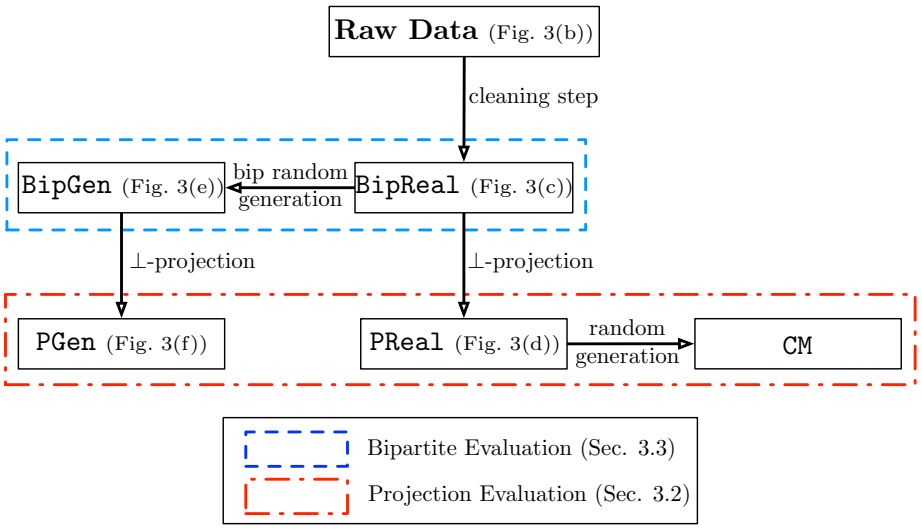


Figure 4.4: Model setup

L3 devices (routers) and L2 devices (typically switches) with links between them and among routers (see Figure 4.3b). Because of these point-to-point links between routers (between R_2 and R_3 in this example), this is not a pure bipartite graph, as defined in Section 4.4.1. However, there is no difference between a point-to-point link and a pair of routers connected through a L2 device[2] that itself is connected only to these two routers. On Figure 4.3b, the direct link between R_2 and R_3 can be replaced by a L2 device of degree 2 linking only R_2 and R_3 (see S_2 in Figure 4.3c) without any loss or addition of relevant information. Indeed, if we $\perp$-project Figure 4.3c, we get back the direct link between R_2 and R_3. In addition, there

[2]The point-to-point link may actually be seen as a L2 device as it can be the case using OSPF.

is no such L2 devices with degree 2 in the raw data as IGMP probing can only detect L2 devices connecting, at least, three routers. Consequently, we replace each point-to-point link between two routers by a new L2 device linking them without any loss of information. This results in a bipartite graph that encodes exactly the same information as the raw data, and that we call `BipReal`. Although this step has no impact regarding the projection, it concerns an important fraction of links in the real data since point-to-point connections represent 58% of all the links on average.

The classical modeling approach consists in computing the $\perp$-projection `PReal` of `BipReal` (see Figure 4.3d), and, then, in modeling it with a random graph `CM` obtained with the *Configuration Model* [ACL00, NWS01, XL07]. This model produces a random graph with the node degree sequence given in input of the model, the one of `PReal` here. We claim that this approach is not satisfying as it does not allow to capture other properties than the one being part of the model. For instance, if one wants to also capture local density properties, one has to look for another model such as the one introduced by Newman [New09]. However, this would lead to the same observation, i.e., no other properties than the ones injected will be captured and one has to look for another model if additional properties are desired.

Instead, we propose a model that relies directly on the real bipartite structure in order to generate graphs that will reproduce several aspects of the actual data. Our model consists in using the degree sequences of L2 and L3 devices in the bipartite graph `BipReal` in order to generate a random bipartite graph `BipGen` (see Figure 4.3e) and project it into a standard graph `PGen` (see Figure 4.3f). Following the tradition of random models, the `BipGen` graph is obtained by shuffling the links between L2 and L3 nodes while maintaining the node degree distribution at both levels. Our expectation is that this bipartite representation of the data will produce a graph `PGen` close to the actual one `PReal`, closer than the `CM` one, in particular regarding other metrics than node degree distribution (e.g., the local density, degree correlations, ...).

For the evaluation, we use the dataset provided by `mrinfo`, as described above and [MVdSD+09]. From the four year daily dataset, we arbitrarily select, each month, the largest output file, leading thus to 56 global topologies (generally they exhibit a large connected component having more than $7,000$ nodes). From this subset, we more specifically focus on the largest topology, corresponding to the data collected by `mrinfo` on 2006/09/07. We infer the presence of L2 devices following methodology discussed by Mérindol et al. [MDBP10, Section 2.3].

The rest of the section is devoted to the comparison of core statistics in order to assess the quality of the models. Section 4.5.1 focuses on statistics on the projection while Section 4.5.2 studies the statistics related to the bipartite level. Our purpose is to check whether this simple process provides good results, in particular regarding metrics that were not injected in the model. Recall that, during the whole transformation process, we only relied on the L2 and L3 node degree distributions. Connections between the two layer devices are then simply randomized without injecting any other structural relationship.

4.5 Model evaluation

In this section, we evaluate the model proposed in previous section.

	Raw Data			Ratios	
	PReal	PGen	CM	2006	Avg case
n	9,740	9,749	9,740	1.00	1.00
m	35,567	48,877	35,470	1.37	1.32
ρ	7.5	10.3	7.5	1.37	1.32
k	7.3	10.0	7.3	1.37	1.32
Δ	58	234	58	4.03	2.93
tr	0.88	0.53	0.01	0.60	0.72
cc	0.58	0.42	0.00	0.72	0.73

Table 4.1: Global statistics for projection evaluation

4.5.1 Projection Evaluation

Here, we evaluate the projection by considering general statistics before going into details.

General Statistics The first statistics we focus on concern some basic properties observed in most real-world networks [WS98], formally presented in Section 4.4.1. For each metric, the right part of Table 4.1 (labeled as "Ratios") positions the data used in this report (the column labeled as "2006") with respect to the set of 56 IGMP topologies (the column labeled as "Avg case"), each BipGen topology being generated 10 times, thus leading to 10 corresponding PGen projections. The left part of Table 4.1 (labeled as "Raw Data") provides absolute values for the PGen and PReal graphs according to the largest topology used over the document.

From Table 4.1, one can see that the number of links, m, is significantly higher for PGen than in the actual graphs (around 37%). It follows naturally that the density ($\times 10^{-4}$ in Table 4.1) and the average degree are also higher for PGen. As explained later in this document, it comes mainly from the fact that there exists overlaps and significant correlations between the two levels of nodes that are not necessarily preserved during the randomization process. On the other hand, the CM graph is particularly close to actual values regarding the same properties. It is not surprising since this model focuses precisely and only on the degree sequence of the projection. Looking at the transitivity ratio and the clustering coefficient, Table 4.1 reveals that the CM model is unable to take account of the local density captured by those coefficients. The PGen model seems, on the contrary, able to capture it (although the values are quite different) in the sense that the local density is relatively very high compared to the global density, the key point for this property.

Finally, it is worth to notice that, for several properties, Table 4.1 reveals that the selected topology positions itself in a worst case scenario compared to the averaged results over the 56 topologies. This is particularly obvious for the highest degree. This indicates that the conclusions drawn from the analysis of this particular case would also be relevant for the other dataset.

A Deeper Analysis In order to refine the general statistics provided above, Figure 4.5 presents the distribution of the degrees for the real data, PReal, and the random graphs generated by the two methods, PGen and CM. The horizontal axis, in log-scale, is the degree of the nodes, while the vertical axis, also in log-scale, presents the inverse cumulative mass.

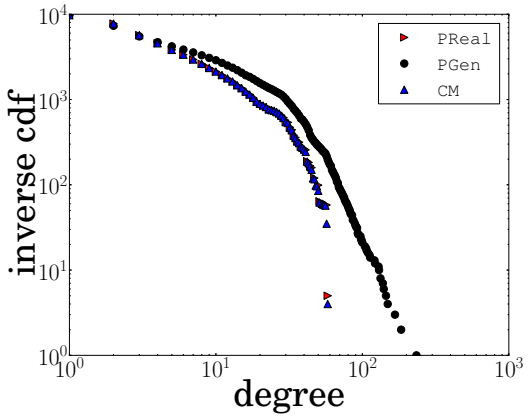


Figure 4.5: Inverse cumulative degree distribution

As expected, the `CM` model is very efficient (it is superimposed on `PReal` on Figure 4.5) as its process is precisely to mimic the degree sequence given in input, i.e., the one of `PReal`.

The slight differences observed stem from the cleaning steps (removing multiple-links, loops, etc) made during the generation. Regarding the `PGen` method, one can see that it is less efficient but it shows a similar distribution. One might notice that the main differences are located in the higher degrees. This is also corroborated by Table 4.1. The highest degree is significantly higher for the `PGen` graph than the real one (234 instead of 53). It partially comes from the fact that, although the generated bipartite graph respects the degree distribution of the routers and the L2 devices, it does not ensure that the overlapping of the L2 devices is preserved, thus increasing the degrees of L3 nodes in the projection. This potential overlapping and other possible correlations between the two layers of node will be investigated more precisely in Section 4.5.2 and 4.6.2.

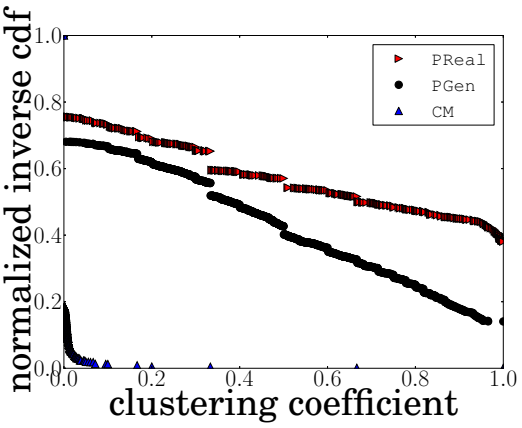


Figure 4.6: Clustering coefficient inverse cumulative distribution

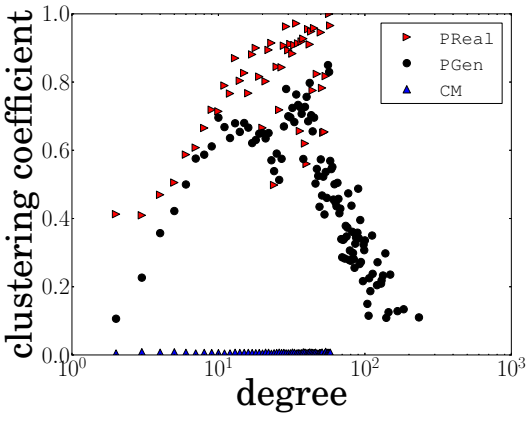


Figure 4.7: Average clustering coefficient associated to a given degree

Figure 4.6 presents the inverse cumulative distribution of the clustering coefficient for the real data, `PReal`, and graphs generated by the two methods, `PGen` and `CM`. Note that the plots are normalized over the number of nodes with degree ≥ 2 in order to avoid side

	BipReal	BipGen
n_{L_2}	10,224	10,224
n_{L_3}	9,758	9,758
$m_{\texttt{bip}}$	25,422	25,415
k_{L_2}	2.5	2.5
k_{L_3}	2.6	2.6
$\rho_{\texttt{bip}}$	0.00025	0.00025
$\texttt{cc}_{\texttt{bip}}$	0.37	0.27

Table 4.2: Global statistics for bipartite evaluation

effects from the nodes of degree 1, for which the notion of clustering coefficient is inadequate. Figure 4.6 clearly shows that the `CM` model is unable to provide a correct representation of such a distribution. This is corroborated by Table 4.1 as the clustering coefficient as well as the transitivity ratio are close to 0. This is due to the fact that the model does not consider the local density and that the number of triangles is very low (only 1 299 triangles while the actual graph has over 203 608 ones). On the other hand, the `PGen` graph provides a similar progression, but with a significant shift of the values.

Figure 4.7 shows the correlation between the node degree and the average clustering coefficient, i.e., a (x, y) dot means that the average clustering coefficient for the nodes having degree x is y. Figure 4.7 confirms the analysis made above. Whatever the degree of a node in the `CM` model, its clustering coefficient remains close to zero. The `PGen` graph, on the other hand, is able to present a similar scatter plot shape, although the values are significantly different. More interestingly, one can see that high clustering coefficients are related to nodes having a similar degree on both figures.

The main difference concerns small degree nodes. For instance, nodes with degree 2 in `PGen` graphs have an average clustering coefficient of 0.1 while actual ones are close to 0.4. This indicates an interesting characteristic of the two bipartite structures. Whereas in the actual bipartite, it seems that, when a router is connected to two others routers, they tend to share L2 devices. This is absolutely not the case for the `PGen` graph. This particular difference concerning degree-2 nodes can be explained by the L2 devices added during the first step of the `PGen` generation (see Section 4.4.2). A deeper study of the degree correlations in the bipartite structure will confirm this statement (see Section 4.6.1).

This first analysis made on the projected graphs confirms the relevance of using bipartite structure to model the data as it succeeds in reproducing *globally* the characteristics of the real network. In particular, it is able capture metrics that are not part of the model. This is a significant improvement in itself since the usual way to obtain properties is to encode them directly in the generation process, which we claim is not satisfying in a long term perspective.

4.5.2 Bipartite Evaluation

This section intends to better characterize the differences observed between real (i.e., `BipReal`) and L2L3 (i.e., `BipGen`) projections from the point of view of the bipartite structure. Following notations presented in Section 4.4.1, we compare standard properties of bipartite graphs.

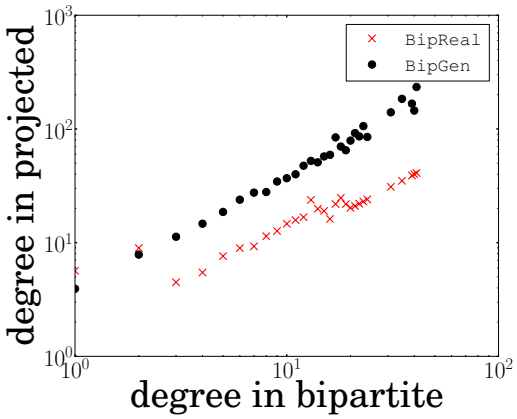


Figure 4.8: Correlation between degrees in the bipartite and in the projection

General Statistics Table 4.2 gathers the statistics presented in Section 4.4.1 for the real and the random bipartite graphs, where $\top$ refers to L2 nodes (L_2) and $\bot$ to L3 nodes (L_3). It shows that all the simple properties are respected by the random bipartite graph, except for the bipartite clustering coefficient for which a slight shift is observed. Note that we do not present the ratios given in Table 4.1 here since they are all equal to 1 (either for this specific case or the average ones), except for the bipartite clustering coefficient for which our case ratio (0.73) is slightly worst than the average value (0.78).

Those observations show that our model succeeds in preserving the global characteristics of the real bipartite structure but do not provide insight on why the projections differ. This is why we turn now to a more refined analysis over those notions.

A Deeper Analysis First, Figure 4.8 presents the correlation between the degree of $L3$ nodes in the bipartite graph and their average degree in the projection, i.e., a (x, y) dot means that the nodes having degree x in the bipartite structure have an average degree y in the projection.

Figure 4.8 shows that the behavior is similar in both cases. In particular, they both follow a straight line in the log-log scale for x values ≥ 3 with a similar slope. But two important differences are noticed. First, the values are significantly lower for the actual bipartite. This indicates some redundancies in the bipartite structure, meaning that many neighbors of nodes in the projection share actually several common L2 devices in the bipartite. This overlapping pattern induces the lowering of their degree in the projection.

From the `BipGen` points in Figure 4.8, one can conclude that this redundancy over the L2 nodes is seemingly lost when shuffling the links in the bipartite. Note that this is true in particular for high degree nodes, suggesting that the difference observed for the highest degree in the projection might be due to this redundancy. Another difference can be pointed out for low degree nodes for which the remark stated above does not stand. Degree-1 nodes in particular present the opposite situation: in real bipartite, the single L2 device to which they are connected happens to have a relatively high degree (close to 6 on average). This differs both from the tendency observed for nodes with degree ≥ 3 and from the random case for which the correlation is consistent for all degrees.

Figure 4.9 and 4.10 focus on the bipartite clustering coefficient as defined in Section 4.4.1.

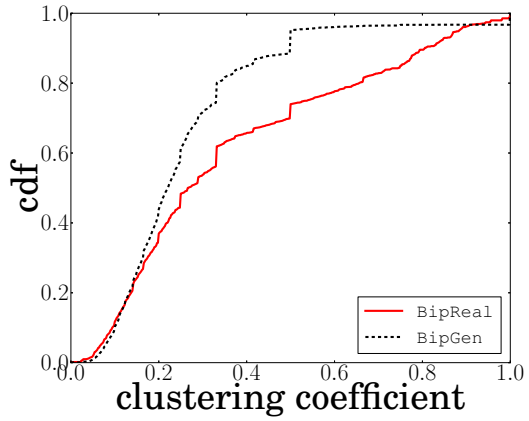


Figure 4.9: Clustering coefficient distribution

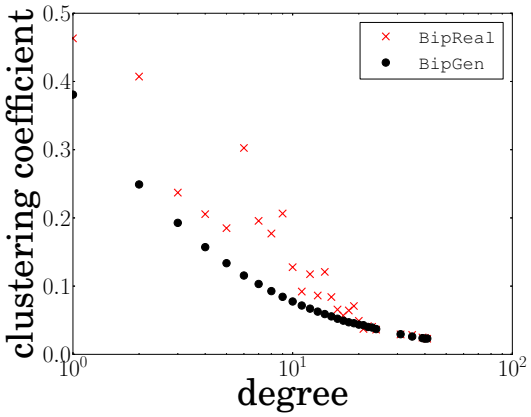


Figure 4.10: Degree correlation

Figure 4.9 presents the cumulative bipartite clustering coefficient of L2 nodes for real and random bipartite graphs, while Figure 4.10 shows the correlation between degree of L2 nodes and their average bipartite clustering coefficient (i.e., a (x, y) dot means that the average bipartite clustering coefficient for L2 nodes having degree x is y).

Both figures show that the two bipartite graphs have a similar behavior regarding this coefficient although a non negligible fraction of nodes in the real bipartite has a higher clustering coefficient than in our model. This is particularly true for low degree nodes (Figure 4.10). This means that low degree L2 nodes tend to share their neighbors with other L2 nodes. This phenomenon explains the gap observed in Figure 4.9 for high clustering coefficients and corroborates the difference observed for the global $\mathtt{cc}_{\mathtt{bip}}$ statistics in Table 4.2. It strengthens also our former remark on the redundancy that seems to be more important in the real bipartite topology than in the random one and that explains the differences observed on high degree nodes in the projection.

In order to test this hypothesis, we compute two more refined properties. The bipartite clustering coefficient, although dealing with overlapping of L2 nodes, is defined for pairs of nodes. We might want to use a more direct notion defined for a single L2 node. One possible solution is to use the *redundancy coefficient* [LMDV08] defined for all L2 nodes v as the fraction of pairs (u, w) of L3 neighbors of v that are connected to a common L2 node other than v. When such a case occurs, then (u, w) is linked in the projection whether v exists or not. Thus, we might consider v as redundant as far as u and w are concerned. Our analysis shows that the two topologies behave very differently regarding this coefficient: on average, 27% of L3 nodes pairs connected to a L2 device in a real case would not be affected by the removing of this device in term of their link in the projection. This proportion drops to 0.3% in the random case.

The notion above focuses on the L2 nodes that are redundant for the projection. One might similarly define a notion of redundancy over the links, i.e., the links that would not modify the projection if they were removed from the bipartite graph. Let us call *internal link* such a link [ATML12]. Our analysis shows that 13.7% of links in the real bipartite graph are internal links, while this proportion is only 0.2% for the random case.

These two last properties are clearly in relation with the notion of degree in the projection

(a) before randomization

(b) after randomization

Figure 4.11: Effects of bipartite random generation (`BipReal` to `BipGen`) on L3 degree

and, as such, explain partially the differences observed in Section 4.5.1. A deeper analysis is left for further works but Section 4.6.3 already provides interesting directions to improve our model.

All the properties explored in Section 4.4.2 show the benefit one can gain from modeling such L2-L3 data with bipartite graphs. While it offers support for generating flat graphs that are able to reproduce *qualitatively* several and independent properties of the original data (see Section 4.5.1), it also proposes new mathematical tools to analyze its structure from the point of view of the bipartite graph itself. In particular, it allows for identifying which aspects of the real network might stem from random processes and which ones are due to strong designed patterns.

4.6 Discussion

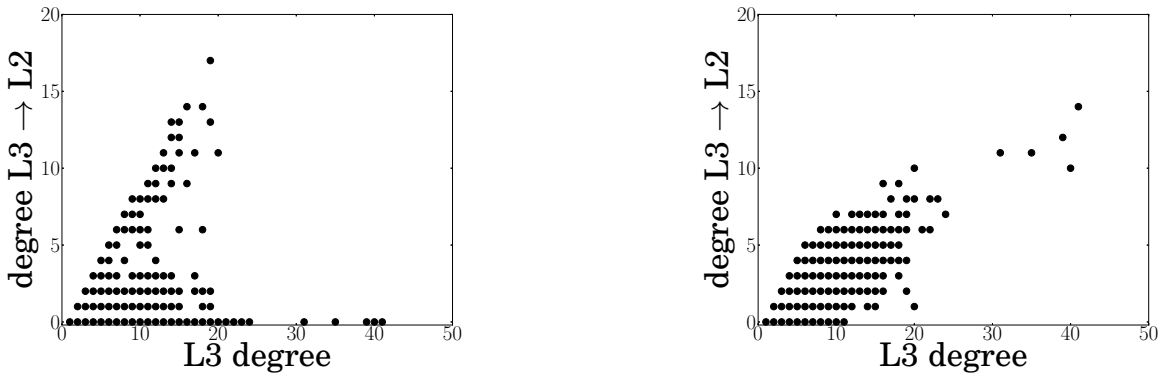
In order to better understand the limitations of our model illustrated in Section 4.5.1 and 4.5.2, we investigate here two interesting properties: (i) we evaluate the effects of the bipartite random generation on L3 degrees and, (ii), we study the redundancy between L2 devices (i.e., we analyze the cases in which removing a L2 node or a L2-L3 link would affect or not the L3 projection).

While the first property (i), detailed in Section 4.6.1, allows us to emphasize an interesting correlation between the L3 degree and the L2-L3 degree, the second property (ii), discussed in Section 4.6.2, allows us to exhibit strong patterns and to explain how point-to-multipoint connections "behave" in real networks. Finally, we envision two possible extensions to improve our model in Section 4.6.3.

4.6.1 Correlation Analysis

Although our model respects the degree distribution of both L2 and L3 devices, there remains one important difference between the raw data and the proposed bipartite structure. As explained in Section 4.4.2, due to the definition of bipartite graphs, we first replace any point-to-point connections between routers by a virtual L2 device connecting them. Although this modeling is strictly equivalent to the point-to-point connection for the projection perspective, it might have an impact on the structure during the randomization process.

Indeed, as shown in Figure 4.11, a simple rewriting in the bipartite graph may induce an important modification for the degree of the nodes in the projected graph. This is the case



(a) Real (b) Random

Figure 4.12: Degree versus point-to-multipoint degree

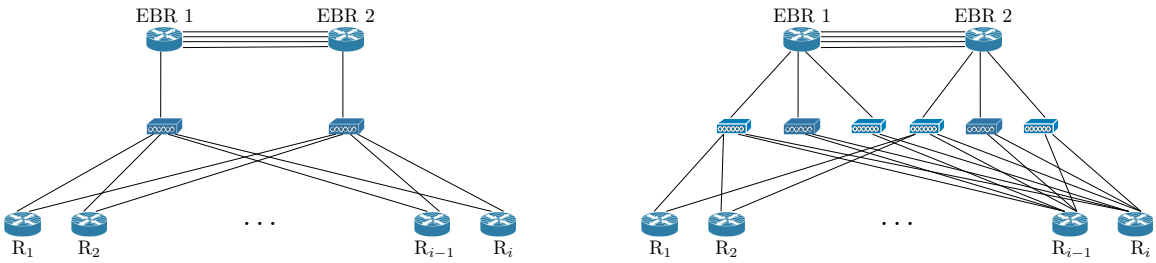
for node B in this virtual example. Before the randomization, it is connected to a unique L3 node (both via node 1 and 2, see Figure 4.11a), thus having degree one in the projection. But simply switching the extremities of links $(2, B)$ and $(3, C)$ leads to a new bipartite graph (see Figure 4.11b) in which B is now connected to every nodes (to A via 1 and to all others via 3), thus increasing drastically its degree in the projection. Obviously, this example is an extreme case but it illustrates how the randomization process at the bipartite level may affect the degree properties of the projections.

In order to study how such a randomization may impact the generated graphs, we investigate how routers are connected to L2 and L3 devices in the raw data and in random graphs (considering here only "actual" L2 devices). Figure 4.12 shows the correlation between L3 degrees and the number of point-to-multipoint connections both for real data and random bipartite graphs. On each plot, a (x, y) dot stands for a router having x links in the bipartite graph, y of them being with a L2 device with degree strictly higher than 2 (recall that a L2 nodes with degree 2 stands precisely for point-to-point connections).

Figure 4.12a shows a striking fact: routers with degree higher than 20 have no connection to real L2 devices but only to point-to-point connections. As explained above in the example, our randomization process does not verify such a strong characteristic, as it can be noticed on Figure 4.12b. This means that, in our model, it is likely that routers having a high L3 degree (i.e., routers having only point-to-point links) will be connected to actual L2 devices (whose degree is strictly greater than 2), thus increasing, and potentially significantly, their degree in the projection.

This observation on real data leads us to explore a more constrained model able to preserve the correlation between the number of point-to-point and point-to-multipoint connections of routers collected in the ground data. It relies on splitting the L3 degree into two disjoint values: one for the number of point-to-point links and one for the number of point-to-multipoint ones. Once such a couple of degrees for each L3 node has been defined, it is easy to adapt our former model to cope with this distinction.

Surprisingly, this approach does not improve significantly the properties observed in the generated projected graphs (regarding the ones observed in the real projections). Some characteristics of the random projections are slightly better than with our model, such as the



(a) 2011 (b) 2007

Figure 4.13: PoP configuration in London for Level3

highest degree that is a bit lower (on average) but still higher than the actual one. Further, the overall degree distribution and correlations studied in Section 4.4.1 fail to be better reproduced. Indeed, the low number of large degree routers in the raw data set limits their impact on global properties.

4.6.2 Redundant Networking Patterns

At the end of Section 4.5.2, we identified an interesting property using the redundancy coefficient and internal links. Indeed, the data considered in this work exhibits many redundant Point-of-Presence (PoP) patterns.

Figure 4.13 illustrates such redundant patterns between L2 devices observed in the raw data. In both Figure 4.13a and 4.13b (both figures come from the Level3 London PoP observed in 2011 and 2007), one can guess that the redundancy coefficient and the number of internal links are high. Such network structures, generally required for physical/logical redundancy and/or load balancing, are not random. Thus, these structures, favoring the network robustness, imply that the degree in the projected graph will be lower for the projection of the real network than for the projection of the random bipartite graph.

On Figure 4.13a, we can observe that the two L2 devices generate two cliques of $i+1$ routers that only differ on EBR1 and EBR2 (while these two routers are connected through multiple parallel point-to-point links). As a result of the projection, i links will disappear in the projection of the real graph while it is likely that, in the random bipartite graph, those links will be distributed over all the network: the projection will then have a higher average and maximal degree.

The example given on Figure 4.13b exacerbates this observation: here, while two of the six L2 devices[3] interconnects the i routers, $i-k$ other routers are connected again through the four others L2 devices. This kind of configuration is not that rare and can, at least, partially explain our random model limitations on metrics previously highlighted.

[3]Note that it is possible that such a symmetry involves some VLAN configurations leading so to two physical L2 devices having three VLANs. Furthermore, such an evolution between 2007 and 2011 suggests an improvement in the architecture capacity.

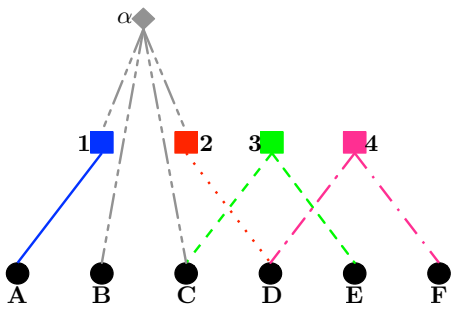


Figure 4.14: A Tripartite view of IP networks

4.6.3 Next Steps

Observations made in Section 4.6.1 and Section 4.6.2 open the way to the design of improved models, potentially fixing lacks of the former one. The two improvements we envision belong to two distinct families of models: *random* and *structural*.

On the one hand, we could rely on the strong redundancy patterns highlighted in Section 4.6.2. Based on such observations, it becomes natural to attempt to capture the redundant PoP patterns illustrated in Figure 4.13. One possible way would be to encode the overlapping among L2 devices in the model itself. To do so, one can extend the bipartite structure into a *tripartite* one using a third level to integrate such a redundancy.

In practice, one can encode any overlapping among L2 devices by the addition of a new node at a third level (L1), connecting both the L2 and L3 devices they are covering (see [LPCN10]). Applying this procedure on the bipartite graph of Figure 4.2a, for example, would result in the tripartite structure presented in Figure 4.14. Indeed, the nodes 1 and 2 are both connected to nodes B and C in the bipartite graph. This overlapping is then encoded in the tripartite graph by the addition of a new third-level node α connecting A, B, 1, and 2.

Once such a tripartite structure has been defined, one can easily apply similar randomization processes than the one proposed in this work. This process would shuffle independently L1-L3 links and L1-L2 links but preserve the structure defined by the new third level. This would result in generating a new tripartite graph presenting the same redundancy patterns than the original ones that could eventually be projected into a bipartite graph.

On the other hand, we could follow the path opened by the observations in Section 4.6.1: there are evidences of a significant correlation between a router degree and the degrees of L2 devices connected to it. We observed, for instance, that very small degree routers are connected to high degree L2 devices (on average). More generally, backbone routers (in an AS core) have a large degree and mostly L3-L3 links, while access routers (providing Internet services to clients) exhibit a lower degree and, generally, mainly L3-L2 connections. However, although this kind of design features can bring improvements in the ability of the model to reproduce this specific property, it also comes with a loss of generality in generated graph properties. As mentioned in the introduction, we believe that random models, such as our tripartite proposal, are more suited for formal analysis.

The two directions suggested here may take several forms. They would require to investigate different mapping as well as to define new extensions of metrics proposed for the bipartite structure analysis. We leave these promising directions for future work.

Chapter 5

Evolution of structural properties of Internet-like networks

5.1 Evolution of structural properties

5.1.1 Objectives and Motivation

In this chapter, we extend the preliminary results presented in D3.1 on the measurement and analysis of the structural properties of the Internet topology (AS/router topology). More precisely, in D3.1, we have measured various properties (that we recall below) on 11 CAIDA maps (from 2004 to 2011). We have also initiated the study of the same set of properties for several models of Internet-like topology (GNP, GNM, GLP). In order to test our simulation tools, we performed the latter study only on networks with small size (around 1000 nodes). Since, we have extended our campaign of measurement. This chapter describes the obtained results and their analysis.

During the second year of EULER, we have considered the evolution of the CAIDA Autonomous Systems (AS) map between 2004 (10k AS) and nowadays (the CAIDA map of June 2012 has around 45k advertised AS). Moreover, for each of the 9 CAIDA maps (one per year) studied during this period, we have generated one GLP graph with the same profile (number of nodes, degree distribution, etc.) and have done the same measurements on these generated models. Note that, due to the huge size of the considered networks, this study has required to improve our tools both of measurement and of simulation. In particular, the new efficient algorithms that we have designed (see Section 3) have allowed us to measure new properties such as hyperbolicity that we were unable to measure before in large-scale graphs.

All these measurements allowed us to analyze the long-term topology evolution of the Internet properties between 2004 and 2012. This is interesting since during this period, the scale of its macroscopic properties has changed durably (e.g., network size increase).

In addition to the new measures we obtained, in this chapter, we provide an analysis of the evolution of the properties of Internet-like networks in short-term dynamic conditions (while D3.1 mainly focussed on the study of these properties in stationary conditions). By short-term dynamic conditions, we mean for instance the failures of links. This is classically modeled by the removal of one (or a "small" set of) link(s) selected either by means of a known distribution or randomly. In our experiments, we have randomly selected in advance the sets of links to remove, thus generating a set of 80 000 graphs on which we have measured

various properties (around 2 months of computation). Our approach has the advantage to be reproducible since all graphs have been stored.

We start by carefully describing our experimental protocol, then we list the obtained results and analyze them.

5.1.2 Experimental Protocol

CAIDA maps and generated graphs. As mentioned above, the experiments that we have conducted consider 9 CAIDA maps (see Table 5.1) [CAI]. For each selected CAIDA map (around one per year), we have derived a corresponding GLP profile.

Recall that, the generation of GLP topology depends on five parameters n, β, m, m_0, p (see page 12 of D3.1 for more details). $\beta \in (-\infty, 1)$ is a tunable parameter that governs the GLP process and indicates the preference for a new node (edge) connected to more popular nodes. The generation of the graph starts with a random tree with m_0 nodes. Then, the following process goes on until the graph reaches n nodes.

- With probability p, $m \leq m_0$ new links are added. A node with degree k_i is chosen to be an endpoint of such a new link with probability $\Pi(k_i) = \frac{k_i - \beta}{\sum_{j=1}^{N}(k_j - \beta)}$ where N is the current number of nodes.
- With probability $1 - p$ a new node with degree m is added. Again, a node with degree k_i is selected to be a neighbor of this new node with probability $\Pi(k_i)$.

For each considered CAIDA map with n nodes, we generated a set of GLP graphs with n nodes. Parameter m is estimated to 1.13 [CCG+02] and $m_0 = 6$ [BT02]. Then, parameters p and β are evaluated according to the formula in [BT02] depending on the degree distribution, the number of nodes, m and m_0. For each CAIDA map, we have generated 100 GLP graphs with same profile n, m, m_0, p and β (depending on the considered CAIDA map).

Table 5.1 resumes the selected CAIDA maps and GLP profiles properties.

CAIDA maps			**GLP parameters**					
Name	Size	Largest BCC	Profile	m_0	m	β	p	Required Size
2004/01/05	16 301	10 424	1	6	1.13	0.8109	0.4410	23 469
2004/12/06	18 501	11 910	2	6	1.13	0.7952	0.4536	27 520
2005/12/05	20 889	13 319	3	6	1.13	0.7830	0.4355	31 075
2006/12/25	23 918	14 949	4	6	1.13	0.7583	0.4494	35 740
2007/12/03	26 690	16 348	5	6	1.13	0.7362	0.4345	40 868
2008/12/01	30 356	19 149	6	6	1.13	0.7042	0.4986	44 640
2010/01/20	33 508	20 940	7	6	1.13	0.6745	0.4951	49 750
2011/01/16	36 878	23 214	8	6	1.13	0.55	0.5973	65 270
2012/06/01	41 203	25 815	9	6	1.13	0.5880	0.6161	68 850

Table 5.1: GLP profiles derived from CAIDA maps.

About the dynamic model. In the sequel, we have studied the short-term evolution of the properties in presence of failures of links. For this purpose, we have considered random failures, i.e., removal of random links, at two levels. First, we have considered the removal of small sets of links (5 links) looking at the impact of the removal of each link. Second, we have considered the impact of the simultaneous removal of larger set of links (100 links).

More precisely, for any graph described in previous paragraph (both the CAIDA maps and the GLP models), we have performed the following process. Starting from an initial graph G_0 with largest bi-connected component C, we have created a sequence $(G_0, G_1, \cdots, G_r)$ of graphs where G_i is obtained from G_{i-1} by removing a set of random links in C. The number of links removed at each time follows the following sequence $(1; 1; 1; 1; 1; 100; 1; 1; 1; 1; 1; 100; \cdots)$, that is, we start by removing one edge, which is done five time, and then we remove 100 edges simultaneously, then we remove one edge five times, and so on. The process is done until C becomes disconnected.

Note that, for each of the 900 generated graphs described in previous paragraph, we have performed this process and stored all the obtained graphs.

This procedure has been also applied on a bipartite graph and its associated 9 random instances as detailed in Section 4 page 25.

About measured properties. We have considered most of the properties documented in D3.1. Each of the following properties has been computed for all CAIDA maps and graphs generated (including the graphs obtained by the dynamic model described above) and their largest bi-connected component (LBCC).

We briefly recall the definitions of these properties (see D3.1 for more details).

- number of nodes n, number of edges m, density $\rho = \frac{2m}{n(n-1)}$;
- maximum- minimum- average- node degree, degree distribution, rank distribution;
- number of (bi)-connected components, distribution of connected components size;
- diameter (length of a longest path), radius (minimum eccentricity);
- distance distribution, average shortest path length $\langle \text{dist} \rangle$ for some node, characteristic path length (median of $\langle \text{dist} \rangle$) clustering coefficient over all nodes);
- hopplot distribution (probability that a random pair of vertices are at distance at most dist);
- clustering coefficient distribution (the clustering coefficient of a node u with degree $\deg(u)$ is the number of edges between neighbors of u over $\deg(u)(\deg(u)-1)/2$), global clustering coefficient (average local clustering coefficient), number of triples (sum of the local clustering coefficient);
- assortativity (probability that two nodes with similar degree are connected).

In addition, we have considered the *robustness* of the considered graph topologies to nodes or edges removals. That is, we have measured the average number of necessary edges (resp. nodes) removals to disconnect the graph. This notion refers to the *all-terminals reliability* as defined by Colbourn [Col87] which is the probability that all-to-all communications can be

established given the probability p of failures of an edge (resp. node). In other words, the higher the number of edges (resp. nodes) to remove from the graph before it gets disconnected, the more reliable it is.

We also are interested in obtaining bounds on other properties that are more difficult to compute such as chordality and treewidth (NP-complete, see Section 3). For these last properties, due to time-limitation, we only obtained some results in stationary conditions (without including the dynamic).

Storage of generated graphs and measures. The generated graphs are stored in separate files (one file per graph). The filename convention is as follows:

`model-name_profile-number_instance-number.edgelist`

When considering graphs with removed links:

`model-name-number_instance-number_edge-number-of-removed-edges.edgelist`

For example, `glp_1_5.edgelist` is the fifth generated instance of the first profile of the GLP model when `as_20040607.edgelist` is the CAIDA map provided the 7 of june 2004. For graphs with removed edges, `glp_1_5_edge-200.edgelist` will be the fifth generated instance of the first profile of the GLP model with the edge removal procedure applied until to have 200 edges removed.

Each evaluated property of a graph is stored in a separate file. The file name is the same as the input graph file name for which the property has been determined plus the name of the property both separated by an underscore. Property file extension is "prop". Content format of the computed property files is the same than in D3.1 experimentations.

Due to the huge amount of topologies generated (over 60GB), a new set of tools based on he Grph library has been designed to build and automatize experimentations for the computation of properties over selected sets of graphs. These computations have been distributed over two clusters at Inria.

5.1.3 Numerical results and Analysis

5.1.3.1 Long term evolution of CAIDA maps

In Deliverable D3.1, several properties of some CAIDA AS maps (between 2004 and 2011) have been reported. In this section, we go further in the analysis of the long-term evolution of these properties. First, we report new measurements based on CAIDA maps released on december 2008 (2008/12) and june 2012 (20012/06). In particular, the CAIDA map of 2012 has around 6000 new nodes and 20000 new edges (compared to the CAIDA map of 2011). Second, we describe the evolution of the properties during this 9-years long period. The measures of the properties of the CAIDA maps are reported in Table 5.2. Table 5.3 describes the measures of same properties of the restriction of these CAIDA maps to their largest bi-connected components.

Property	**CAIDA maps**								
	2004/01/05	**2004/12/06**	**2005/12/05**	**2006/12/25**	**2007/12/03**	**2008/12/01**	**2010/01/20**	**2011/01/16**	**2012/06/01**
Number of nodes	16 301	18 501	20 889	23 918	26 690	30 356	33 508	36 878	41 203
Number of edges	32 955	38 265	41 820	49 089	55 336	64 768	75 001	103 485	121 309
Density	0.00024	0.00023	0.00019	0.00017	0.00015	0.00015	0.00013	0.00015	0.00014
Minimum degree	1	1	1	1	1	1	1	1	1
Average degree	4.04	4.14	4	4.10	4.15	4.27	4.48	5.61	5.89
Maximum degree	2 331	2 328	2 379	2 358	2 632	2 283	2 631	2 972	3 537
Diameter	10	10	10	11	10	10	11	11	10
Radius	5	5	6	6	5	5	6	6	5
Avg. shortest path length	3.77	3.79	3.83	3.87	3.87	3.84	3.86	3.81	3.84
Characteristic path length	3.64	3.65	3.71	3.74	3.80	3.79	3.82	3.77	3.78
Number of biconnected comp.	1 378	1 550	1 736	2 022	2 317	2 540	2 891	3 124	3 488
Largest biconnected comp.	10 424	11 910	13 319	14 949	16 348	19 149	20 940	23 214	25 815
Number of connected triples	$8.8 \cdot 10^6$	$10.2 \cdot 10^6$	$11.1 \cdot 10^6$	$13.1 \cdot 10^6$	$14.9 \cdot 10^6$	$19.9 \cdot 10^6$	$23 \cdot 10^6$	$34 \cdot 10^6$	$43 \cdot 10^6$
Global clustering coefficient	0.233	0.243	0.236	0.221	0.203	0.238	0.217	0.240	0.240
Assortativity	0.015	0.017	0.017	0.02	0.017	0.026	0.023	0.04	0.04
Nodes robustness	12	11	11	11	10	11	11	11	11
Edges robustness	5	6	5	6	6	6	6	8	8
LPower law exponent	-1.11	-1.11	-1.12	-1.12	-1.14	-1.13	-1.13	-1.13	-1.11
Caterpillarity	244	321	306	294	390	560	680	396	552
Treewidth caterpillarity	1 703	1 987	1 693	3 352	3 897	3 193	3 599	5 279	4 266

Table 5.2: Values of properties for CAIDA maps.

Table 5.3: Values of properties for the largest biconnected components of the CAIDA maps.

Property	**CAIDA maps - biconnected component**								
	2004/01/05	**2004/12/06**	**2005/12/05**	**2006/12/25**	**2007/12/03**	**2008/12/01**	**2010/01/20**	**2011/01/16**	**2012/06/01**
Number of nodes	10 424	11 910	13 319	14,949	16 348	19,149	20 940	23 214	25 815
Number of edges	27 061	31 666	34 236	40,105	42,972	57 192	62,406	89 783	105 894
Density	0.0005	0.00045	0.00039	0.00036	0.00032	0.00031	0.00028	0.00033	0.00032
Average degree	5.19	5.32	5.14	5.36	5.26	5.97	5.96	7.73	8.20
Average degree	3.00	3.09	2.98	3.12	3.05	3.48	3.49	4.55	8.20
Maximum degree	1 952	1 964	2 019	2 006	2 277	1 955	2 400	2 679	3 047
Diameter	8	8	8	8	8	8	8	8	8
Radius	4	5	4	4	4	4	4	4	4
Avg. shortest path length	3.38	3.40	3.45	3.46	3.46	3.47	3.49	3.45	3.47
Characteristic path length	3.24	3.25	3.33	3.34	3.35	3.38	3.41	3.40	3.44
Number of connected triples	$6.5 \cdot 10^6$	$7.7 \cdot 10^6$	$8.3 \cdot 10^6$	$9.9 \cdot 10^6$	$1.1 \cdot 10^7$	$1.5 \cdot 10^7$	$1.8 \cdot 10^7$	$2.8 \cdot 10^7$	$3.6 \cdot 10^7$
Global clustering coefficient	0.371	0.387	0.378	0.363	0.342	0.388	0.357	0.391	0.394
Assortativity	0.019	0.022	0.021	0.025	0.021	0.033	0.028	0.049	0.05
Nodes robustness	114	134	141	150	166	189	196	215	285
Edges robustness	297	295	326	399	378	444	457	680	701
Power law exponent	-1.08	-1.09	-1.1	-1.1	-1.11	-1.11	-1.12	-1.13	-1.11
Caterpillarity	-	-	-	-	-	-	-	-	-
Treewidth caterpillarity	-	-	-	-	-	-	-	-	-

Largest bi-connected component. The size of the AS network has increased from 16301 nodes and 32955 edges (2004) to 41203 nodes and 121309 edges (2012) while the size of its bi-connected component has increased from 10424 nodes and 27061 edges (2004) to 25815 nodes and 105894 edges (2012). It is interesting to observe that the ratio between the size of the network and its largest bi-connected component has been stable for the nodes around 1.59 and decrease from 1.21 to 1.12 for the edges until 2010 and remains stable around 1.14 for 2011 and 2012. In 2008, the ratio between the number of edges of the network and the number of edges of its largest bi-connected component has dropped down to 1.12 which can be explained by the fact that several smaller bi-connected components that were existing until 2008 have been merged with the main largest bi-connected component.

Global clustering coefficient and Density. Between 2004 and 2012, the global density has oscillated between 0.00024 and 0.00014. Similarly, the local density represented by the clustering coefficient slightly oscillates between 0.203 (in 2007) and 0.240 (Dec. 2004) and is currently equal to 0.243. This indicates that the ratio between the number of new edges over the number of new nodes remains stable (very slightly increases).

Degree distribution. The general trend is the increasing of degree of nodes. Hence, the average degree increased from 4.04 to 5.89 and the maximum degree increased from 2331 to 3537. This clearly confirms that nodes tends to connect to high degree nodes. Also, the degree distributions for each CAIDA map are plotted in Figure 5.2b. They clearly follow a power law distribution whose coefficient remains very stable between -1.11 and -1.14.

In the largest bi-connected component, the average degree increases from 5.19 to 8.20. The increase is particularly important during the last two years. This is due to the fact that the largest bi-connected component has increased a lot (see paragraph above) and that, now, almost all new links belong to it.

The assortativity has be multiplied by 2 (from 0.015 to 0.04). This positive assortativity expresses that many nodes with similar degree are connected with each other.

Shortest path length. The diameter has remained stable equal to 10 (and to 8 in the bi-connected component). The average shortest path length has first increased from 3.77 to 3.83 (from 2004 to 05) and then has oscillated between 3.81 to 3.87. It is currently equal to 3,84. The average shortest-path length in the largest bi-connected component has followed the same evolution from 3.84 to 3.47. It is important to note the small difference between the average shortest-path length in the network and in the largest bi-connected component. This indicates that most (all) of the nodes that do not belong to the largest bi-connected component are very closed to (neighbors of) it.

Edge and Node robustness. To measure the edge- and node- robustness of our networks and their bi-connected components, we have randomly removed nodes (resp., edges) until disconnecting the network. We have repeated this experiment 100 times for each network and we report the average of the obtained value. Clearly, the deviations of our experiments are very high since the number of experiments (100) is quiet small compared with the size of the networks. This is due to time-limitations.

The node-robustness and the edge-robustness of the AS network have not evolved during the studied period since it kept a very small value: from 10 to 12 for the nodes and from 5

to 8 for the edges. This is not surprising since most of the nodes that are not in the largest bi-connected component are nodes with degree one and their proportion has remained the same.

More interestingly, the "average" node-robustness of the largest bi-connected component has increased from 114 to 285 and the "average" edge-robustness has increased from 297 to 701. Surprisingly, this implies that, in 2004, the removal of $1,1\%$ edges chosen randomly allowed to disconnect the largest bi-connected component, while in 2012, removing randomly $0,7\%$ of the edges is sufficient. Maybe this behavior is due to the small number of experiments we did due to time-limitation.

Cycles and Bags We have executed the algorithm detailed in Section 3) on each of the CAIDA maps. Due to its time complexity (quadratic in the number of edges) we have only done one execution per map. However, note that, since it is a greedy algorithm that depends on the ordering of the vertices, it would be interesting to do other executions.

Using this algorithm, we were able to find induced cycles larger than 244 (CAIDA map of 2004) and induced cycles larger than 552 (in 2012). This is a bit surprising since the high clustering coefficient suggests that such large induced cycles should not exist or should at least be rare. It is an interesting question to know whether there are numerous such large induced cycles. Also this polynomial-time algorithm allowed us to show an upper bound of 1703 (in 2004) and of 4266 (in 2012) of the treewidth of the AS network.

In conclusion, these results clearly express the growth of the network (number of nodes, edges, maximum degree, etc.) while its general global/average characteristics remain unchanged (power law coefficient, diameter, proportion between the whole graph and the largest bi-connected component etc.). As an exception, and surprisingly, the robustness of the network has decreased.

5.1.3.2 Long term evolution of GLP and comparison with CAIDA measures

In [BT02], the analyze of the GLP model shows that it fits several properties (eg. degree distribution, average shortest path, etc) of the Internet. However, this study was only performed in comparison with a CAIDA map of 2002. Thus, the next results present the long term evolution of GLP graphs and compares them with the corresponding CAIDA maps: both the whole graphs and their bi-connected component were analyzed. The measures of the properties of the GLP graphs are reported in Table 5.4. Table 5.5 describes the measures of same properties of the restriction of these GLP graphs to their largest bi-connected components.

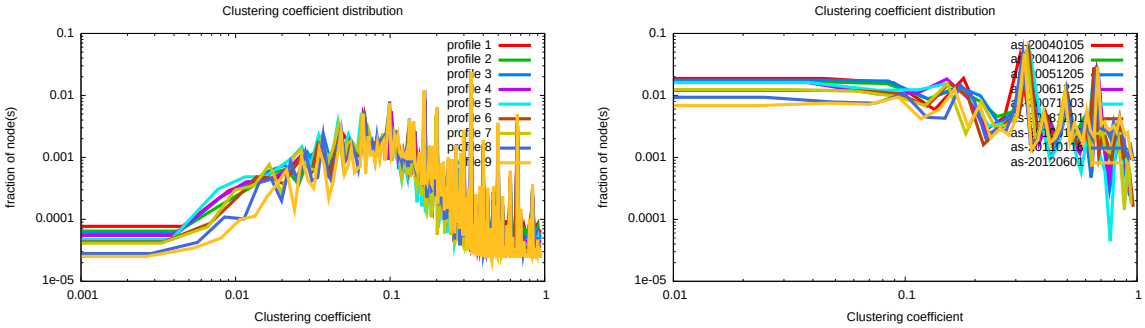
Largest bi-connected component and Density. The size of GLP graphs increased from 16301 nodes to 41203 nodes as GLP profiles have been computed from CAIDA maps. The number of edges increased from 29079 edges (profile 1) to 107 964 edges (profile 9). However, for each profile, the number of edges is smaller compared to those of CAIDA maps. When observing the ratio between the size of the GLP graph with its largest bi-connected component, the ratio decreased from 3.34 to 2.46 for the nodes and 1.64 to 1.29 for the edges. This is significantly different from the evolution of the CAIDA maps bi-connected components which remains stable over the years with a ratio of 1.59 for the nodes and decreased from 1.21 to 1.13 between 2004 and 2008 and remained stable around 1.14 until 2012 for the edges.

Property	GLP profiles with β								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Number of nodes	16 301	18 501	20 889	23 918	26 690	30 356	33 508	36 878	41203
Number of edges	29 079	33 918	37 523	44 322	48 729	61 825	68 425	93 314	107 964
Density	0.00022	0.0002	0.00017	0.00015	0.00014	0.00013	0.00012	0.00014	0.00013
Minimum degree	1	1	1	1	1	1	1	1	1
Average degree	3.57	3.67	3.59	3.7	3.65	4.07	4.08	5.06	5.24
Maximum degree	1 056	1 114	1 209	1 272	1 359	1 352	1 422	1 343	1 429
Diameter	8.9	9.06	9.2	9.4	9.6	9.5	9.67	9.58	9.34
Radius	4.9	5	5	5	5	5	5.1	5.1	5.01
Avg. shortest path length	3.60	3.63	3.67	3.7	3.75	3.72	3.76	3.71	3.68
Characteristic path length	3.52	3.54	3.58	3.61	3.67	3.64	3.68	3.62	3.58
Number of biconnected comp.	1 988	2 304	2 679	3 153	4 781	5 320	4 741	5 363	
Largest biconnected comp.	4 869	5 660	6 419	7 614	8 615	10 485	11 881	15 238	5 698
Number of connected triples	$4.3 \cdot 10^6$	$5.2 \cdot 10^6$	$6 \cdot 10^6$	$7.2 \cdot 10^6$	$8 \cdot 10^6$	$1 \cdot 10^7$	$1.2 \cdot 10^6$	$1.7 \cdot 10^7$	$2.2 \cdot 10^7$
Global clustering coefficient	0.069	0.067	0.062	0.060	0.055	0.056	0.064	0.076	0.07
Assortativity	0.084	0.09	0.082	0.087	0.08	0.108	0.105	0.161	0.17
Nodes robustness	9.12	8.09	7.9	7.3	7.35	7.04	6.9	6.94	7.3
Edges robustness	2.53	2.66	2.61	2.73	2.71	3.09	3.1	4.36	4.4
Power law exponent	-1.13	-1.14	-1.15	-1,16	-1.17	-1.17	-1.18	-1.18	-1.16

Table 5.4: Average values of properties for GLP graphs with β.

Property	GLP profiles with β - biconnected component								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Number of nodes	4 870	5 661	6 420	7 615	8 615	10 486	11 881	15 238	16 716
Number of edges	17 647	21 077	23 053	28 018	30 653	41 954	46 797	71 674	83 456
Density	0.00149	0.00132	0.00112	0.00097	0.00083	0.00076	0.00066	0.00062	0.00060
Minimum degree	2	2	2	2	2	2	2	2	2
Average degree	7.25	7.45	7.18	7.36	7.12	8.00	7.88	9.41	9.99
Maximum degree	667	722	781	850	912	980	1 042	1 098	1 171
Diameter	6.19	6.29	6.48	6.62	6.90	6.86	6.92	6.86	6.74
Radius	3.53	3.56	3.77	3.85	3.99	4.00	4.00	4.00	3.99
Avg. shortest path length	3.09	3.11	3.15	3.17	3.22	3.20	3.24	3.23	3.20
Characteristic path length	3.03	3.05	3.08	3.11	3.15	3.14	3.17	3.15	3.13
Number of connected triples	$1.96 \cdot 10^6$	$2.46 \cdot 10^6$	$2.78 \cdot 10^6$	$3.56 \cdot 10^6$	$3.95 \cdot 10^6$	$6.07 \cdot 10^6$	$6.90 \cdot 10^6$	$1.21 \cdot 10^7$	$1.58 \cdot 10^7$
Global clustering coefficient	0.273	0.259	0.241	0.224	0.202	0.202	0.184	0.176	0.194
Assortativity	0.144	0.148	0.135	0.138	0.128	0.155	0.148	0.200	0.210

Table 5.5: Average values of properties for the largest biconnected component of GLP graphs with β.



(a) GLP graphs

(b) CAIDA maps

Figure 5.1: Clustering coefficient distributions for CAIDA maps and corresponding GLP graphs.

The sizes of the largest bi-connected components in CAIDA maps and their corresponding GLP models are very different: the one of CAIDA is more than twice than the one of GLP model. This is due to the preferential attachment function of the GLP model which makes the new nodes to attach preferentially to a small kernel avoiding to reinforce the connectivity of these new nodes. To better parameterize the GLP model, it could be interesting to slightly modify the preferential attachment function related to the addition of new edges. This phase of addition of new edges could be used to increase the size of the largest bi-connected component.

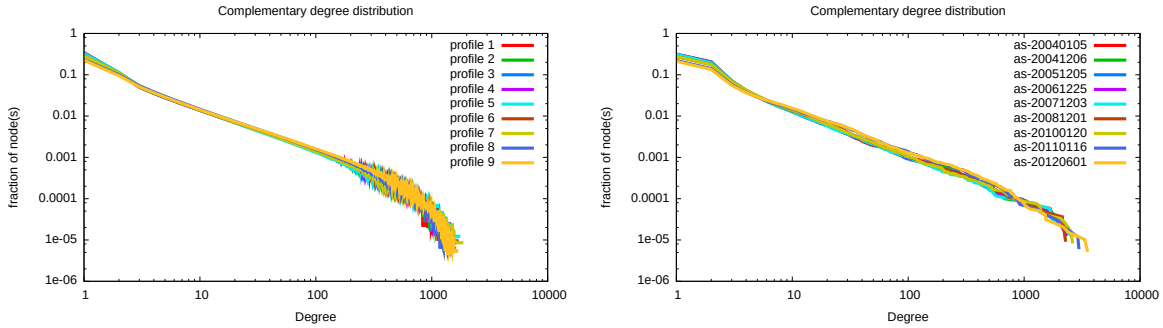
The global density oscillate from 0.00022 to 0.00014 which is close the density evolution of the CAIDA maps.

Global clustering coefficient. The long-term evolution of the coefficient clustering distribution of GLP graphs is depicted in Figures 5.1a. It oscillates between 0.055 (profile 5) and 0.076 (profile 8). There is a clear difference with the measures of this property in CAIDA maps (oscillations between 0.203 in 2007 and 0.240 in Dec. 2004). This is not surprising since it is well known that the property of having a high clustering coefficient is not captured by the GLP model. This is mainly due to the larger number of degree-one nodes in GLP graphs (see paragraph below) and to the fact that, in CAIDA map, a non negligible fraction of nodes have clustering coefficient between 0.2 and 1 while, in GLP, very few nodes have clustering coefficient exceeding 0.2.

Note that in D3.1 we had noticed a difference between our measures (performed on GLP graphs with 1000 nodes) and the measures obtained in [BT02] for similar models. Our new results obtained on larger instances confirm our previous observations.

Degree distribution. First, we notice that the degree-distribution of GLP models does not change and keeps following a power law distribution similar to the one of the corresponding CAIDA maps. This is clearly expected by construction of these graphs.

The long-term evolution of the average and maximum degree of the GLP graphs follows the same trend as the ones of the corresponding CAIDA maps. The average degree increases from 3.57 to 5.24, i.e., both the value and the evolution are comparable to the one of CAIDA maps. On the other hand, the maximum degree of GLP graphs increases from 1056 to 1429.



(a) GLP graphs

(b) CAIDA maps

Figure 5.2: Complementary degree distributions for CAIDA maps and corresponding GLP graphs.

This is much smaller than the maximum degree observed in CAIDA maps (from 2331 to 3537). This difference can be better observed when looking at the degree distribution in Figures 5.2a and 5.2b. Indeed, the degree distributions of CAIDA maps and corresponding GLP models is comparable until degree around 1000. However, the tails of the degree distributions of the CAIDA map speads until nodes with degree more than 3000, while the tails of the degree distributions of the GLP graphs concentrate all large degree node around 1000.

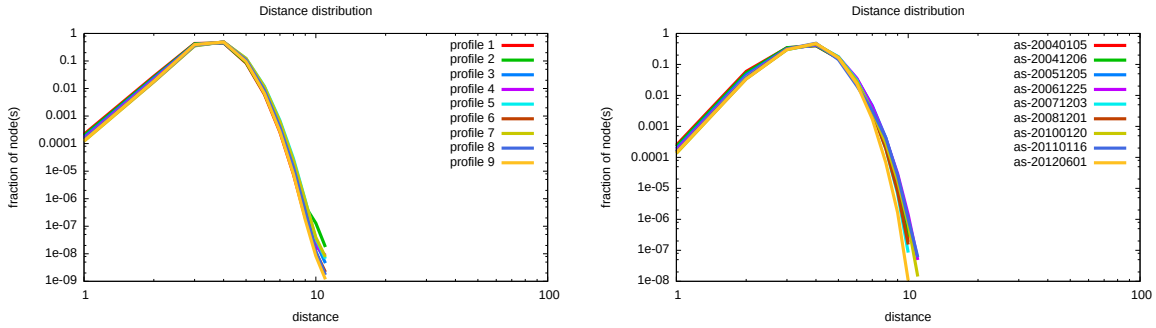
This can be explained by the fact that, in the AS network, a small set of ASes are much more important than other. On the other hand, in GLP graphs, nodes that emerge with a bigger degree will grow in parallel due to the preferential attachment function. It would be interesting to put a weight on a small fraction of the nodes among the ones with large degree to bias the preferential attachment.

Shortest path length. On GLP graphs we observe a diameter comprised between 8.9 and 9.34 which is close to the diameter of 10 in CAIDA maps. In the GLP graphs bi-connected components the diameter is comprised between the smaller values 6.19 and 6.92 compare to the diameter of 8 in the bi-connected component of the CAIDA maps.

The average shortest path length in GLP has increased from 3.60 (profile 1) to 3.75 (profile 5) and has then oscillated between 3.68 and 3.76. This is very similar to CAIDA maps shortest path lengths which has increased from 3.77 to 3.83 and oscillated between 3.81 and 3.87. In the bi-connected component of the GLP graphs the average shortest path length increased between 3.09 to 3.15 and has then oscillated between 3.14 and 3.13.

Figure 5.3 shows the distance distributions on a loglog scale on GLP graphs and CAIDA maps. As expected from the previous observations, both of the plots are very similar. These similarities were also observed in [BT02].

To conclude, GLP graphs and CAIDA maps have similar behavior for most of the properties. They mainly differ concerning the clustering coefficient that was already known as a drawback of the GLP model. Here, we have confirmed that these similarities keep unchanged through the long-term evolution. Another differences that we pointed out in D3.1 and here are the smaller size of the largest bi-connected component of the graphs generated via the GLP model, such as the difference in their maximum degree. Here we propose an alternative

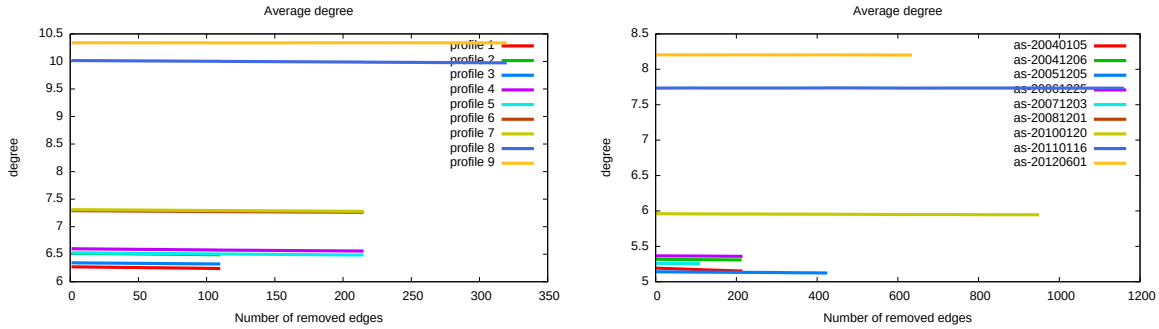


(a) GLP graphs (b) CAIDA maps

Figure 5.3: Shortest path length distribution.

to cope with these defaults (see the corresponding paragraphs above).

5.1.3.3 Short-term evolution



(a) GLP graphs (b) CAIDA maps

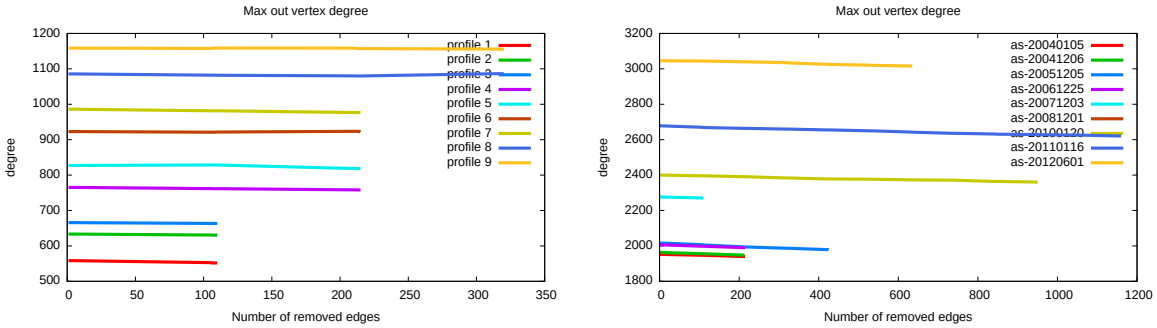
Figure 5.4: Short term evolution of the average degree.

As described in the experimental protocol section (Section 5.1.2), in each of the considered graphs, we have sequentially removed small set of randomly selected edges of the largest biconnected component until it becomes disconnected. The results obtained so far are reported in Figures 5.4 to 5.9.

The results show no significative evolution of the measured properties (average and maximum degrees, assortativity, density, clustering coefficient, average shortest path length, diameter and radius). Some pathological behaviors appear for the CAIDA map of 2004 and the corresponding GLP graphs which can be explained by their small sizes and smaller average degrees.

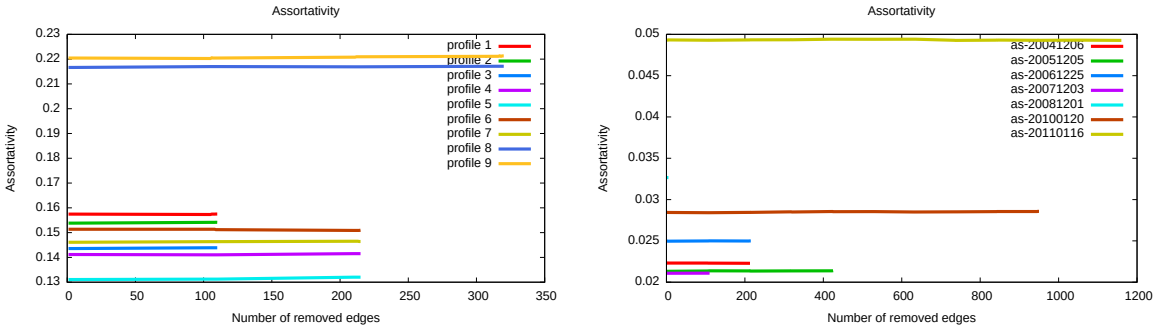
These experiments highlight that the AS network is very robust to random (small) failures. Moreover, the similarity between the CAIDA maps and the GLP graphs behaves well under this kind of short term dynamic.

However, these properties reflect a global behavior of the considered topologies and do not reflect the local changes in the connectivity. For instance, shortest path lengths are



(a) GLP graphs (b) CAIDA maps

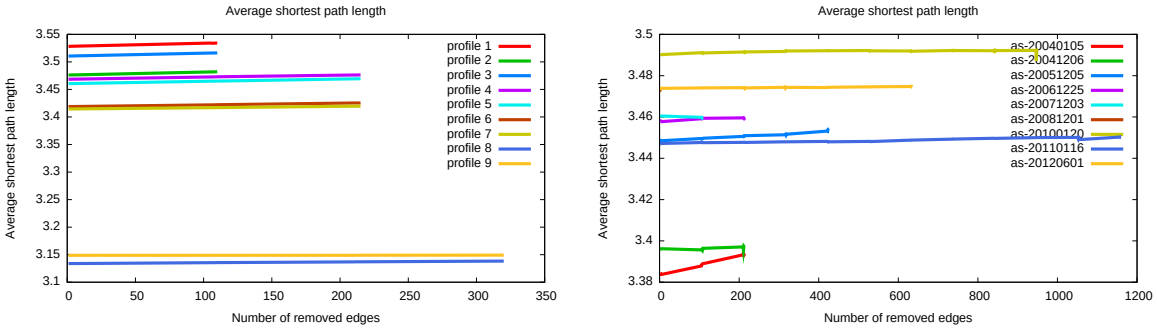
Figure 5.5: Short term evolution of the maximum degree.



(a) GLP graphs (b) CAIDA maps

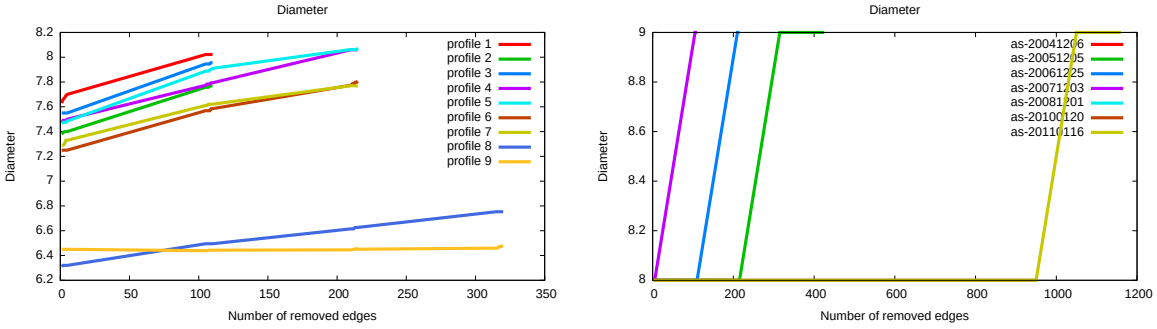
Figure 5.6: Short term evolution of the assortativity.

very stable but the actual shortest paths could differ a lot: routing tables entries could be drastically modified. Therefore, the next steps will be as follows. On one hand, we will study more clever dynamic scenarii modeling more localized or correlated failures (e.g., shared risk link groups). On the other hand, we will study the local impact of these topological changes. We have initiated such a study in Chapter 6.



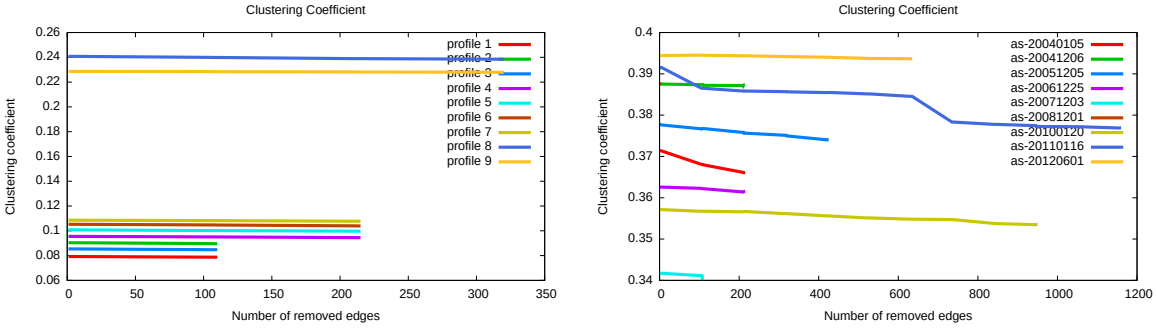
(a) GLP graphs (b) CAIDA maps

Figure 5.7: Short term evolution of the average shortest path length.



(a) GLP graphs (b) CAIDA maps

Figure 5.8: Short term evolution of the diameter.



(a) GLP graphs (b) CAIDA maps

Figure 5.9: Short term evolution of the clustering coefficient.

5.2 Evolution of the hyperbolicity

5.2.1 Objectives and Motivation

In Deliverable D3.1 and in Section 3 page 11 of this deliverable, we recalled that the (Gromov) *hyperbolicity* of a graph reflects how the metric (distances) of the graph is close to the metric of a tree. This structural property has been proven effective for the design of distance decreasing routing schemes (greedy routing), in particular for graph topologies with small hyperbolicity. More precisely,

Definition 10. *Given four nodes u, v, w, x of a graph G, the half difference between the two larger of the distance sums* $\mathrm{dist}(u,v)+\mathrm{dist}(w,x)$, $\mathrm{dist}(u,w)+\mathrm{dist}(v,x)$, $\mathrm{dist}(u,x)+\mathrm{dist}(v,w)$ *is called the hyperbolicity of the 4-tuple* (u,v,w,x).

The hyperbolicity δ of an n-node graph G is the maximum hyperbolicity of its $\binom{n}{4}$ 4-tuples.

The Internet topology evolves over time due to its growth (addition of new nodes and links) and to equipments or softwares errors causing the temporary unavailability of some nodes and links. In this section, we perform an extensive study of the evolution of the hyperbolicity. This study mainly relies on CAIDA maps but also on simulations performed on random generated graphs (GLP, Erdos-Reyni, etc.) with same kind of dynamics.

This bunch of data allows us to show that the hyperbolicity of Internet-like graphs is resistant to the deletion of up to 15% of the edges. More important, it appears that this stability occurs at an even more local level since the hyperbolicity of the 4-tuples is almost unchanged (both for the hyperbolicity of the 4-tuples independently and for their distribution). This reveals that there are many shortest paths between (almost) any pairs of nodes. It would be interesting to better understand the corresponding hidden structure that is related to the density and to the betweenness centrality of such networks.

We start in Section 5.2.2 with some theoretical facts on the short term evolution of the hyperbolicity of a graph. Then, in Sections 5.2.3 and 5.2.4, we discuss the short and long terms evolution of the hyperbolicity of maps of the Internet.

5.2.2 Theoretical worst case behavior

In this section, we focus on very simple dynamic scenarii when random edges are removed or added. For simple graph classes, we show that in the worst case, the hyperbolicity may differ drastically and that it may both increase or decrease in an unpredictable way.

The short term evolution of the hyperbolicity corresponds to the evolution of this value when one or few links are added to or removed from a given graph. From a theoretical perspective, it is easy to show that the addition or the removal of an edge can change drastically the hyperbolicity of a graph. Furthermore, this simple modification of the graph may either increase or decrease the hyperbolicity. To understand this, let us consider some examples. We first recall the hyperbolicity of some simple graph classes.

- *Block* graphs (i.e., connected graph in which every 2-connected subgraph is a clique) are 0-hyperbolic, and so are trees and cliques;
- Cycles of order $n = 4p+\varepsilon$, with $p \geq 1$ and $\varepsilon \in \{0,1,2,3\}$, are $(p-1/2)$-hyperbolic when $\varepsilon = 1$, and p-hyperbolic otherwise;
- $n \times m$ grids, with $2 \leq n \leq m$, are $(n-1)$-hyperbolic.

We now show that the addition of an edge may decrease the hyperbolicity of a graph, and reciprocally that the removal of an edge can increase the hyperbolicity.

- Let C_4 be the cycle of order 4 with vertex set $V = \{0, 1, 2, 3\}$ and edges $\{(i, i + 1 \bmod 4),\ i \in V\}$. We know that C_4 is 1-hyperbolic. Adding edge $(0, 2)$, we obtain a 1/2-hyperbolic graph. Adding further edge $(1, 3)$, we obtain the complete graph K_4 which is 0-hyperbolic.
- Let C_{4p}, with $p \geq 2$, be the cycle of order $4p$. The addition of edge $(0, i)$, for $i \in [2 \dots 2p - 1]$, reduces the hyperbolicity from p to $\lceil (4p - i)/4 \rceil - \alpha_i$, where $\alpha_i = 0$ when $i \equiv 0, 1 \bmod 4$, $1/2$ when $i \equiv 2 \bmod 4$, and 1 when $i \equiv 3 \bmod 4$.

Next, we show through examples that adding an edge to a graph may drastically increase its hyperbolicity, and so reciprocally that the removal of an edge may drastically decrease the hyperbolicity of a graph.

- The path P_{4p} of order $4p$, with $p \geq 1$, is 0-hyperbolic but the cycle C_{4p} is p-hyperbolic;
- Let G be the 1-hyperbolic $2 \times 4q$ grid, with $q \geq 2$. Let now H be the graph G plus the edge $((0, 0), (1, 4q - 1))$. Then H is q-hyperbolic with certificate $\{(0, 2q), (0, 3q + 1), (1, 0), (1, q)\}$.

With above examples, we have seen that the addition or the removal of a well chosen edge can change drastically the hyperbolicity of a graph. The question is now to check whether such extreme behaviors may happen in practice, and in particular on Internet-like graphs. In next subsections, we show that the hyperbolicity of Internet-like networks actually tolerates very well random edge deletions.

5.2.3 Long term evolution of the hyperbolicity of Internet-like networks

CAIDA maps. In Deliverable D3.1, we have reported on the hyperbolicity of some CAIDA maps of the AS topology of the Internet. Thanks to our new algorithm for determining the hyperbolicity of a graph, presented in Section 3.1.2 page 13 of this deliverable, we have been able to compute the hyperbolicity of all available CAIDA maps since 2004 (173 maps). Although the new algorithm improves upon the naive one, the overall computation time represents several months of computations. Indeed, the determination of the hyperbolicity of the 2011 CAIDA map only took 12 days of computations (but maps with hyperbolicity 3 requires only a few minutes of computations). The computed values are reported in Figure 5.10 where the first axis corresponds to the dates the maps where produced. We have one dot per CAIDA map and we have also reported the linear interpolation of the hyperbolicity.

Figure 5.10 provides several informations. First, we observe some measurement bias, for instance for the map of 07/06/2004 which has hyperbolicity 2 while maps produced in the months before and after have hyperbolicity 2.5. The same holds for the maps with hyperbolicity 3 (05/09/2005 and 06/02/2006). A frequent variation is also observed between consecutive maps in the period from 2007 till 2009. We still do not know whether this behavior is due to some bias of the measurement or whether it comes from some hidden fact.

The main observation resulting from this experimet is that the hyperbolicity of CAIDA maps has decreased in average from 2.5 to 2 and the hyperbolicity is stable since 2009. This is not surprising and it is clearly due to the fact that the AS network has become bigger during the last decade. In particular, many new links have appeared and have broken large cycles that made 4-tuples with bigger hyperbolicity.

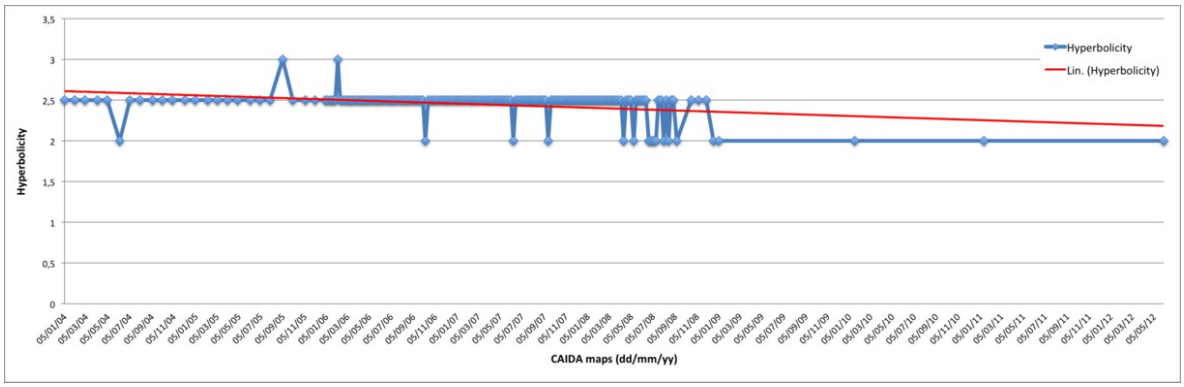


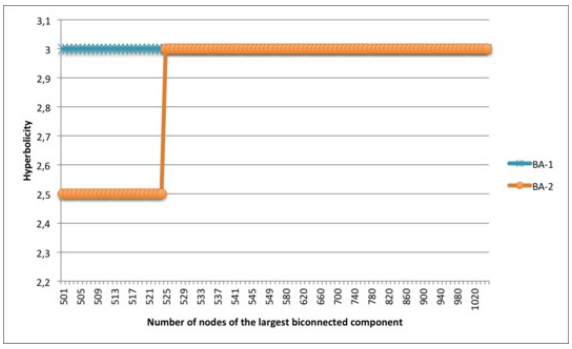
Figure 5.10: Evolution of the hyperbolicity of CAIDA maps since 2004.

Random generated graphs. To have a deeper understanding of this behavior, we have conducted similar experiments on generated graphs (some of them modeling Internet-like networks). More precisely, we have observed the evolution of the hyperbolicity of three random graph models, GNP, BA, GLP (see Chapter 1 of D3.1 for definitions). It appears that the hyperbolicity keeps the same "stability" over the growth of such evolving networks.

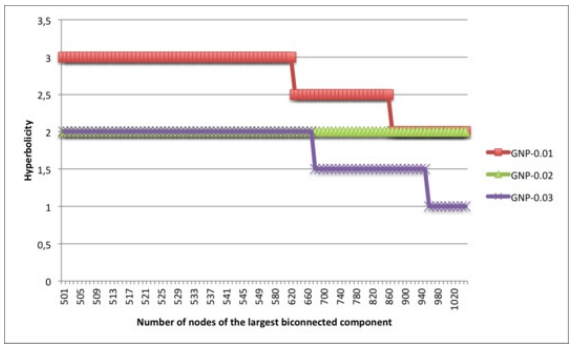
For the GNP graph model, we start from a GNP graph with $n = 500$ nodes generated with probability $p \in \{0.01, 0.02, 0.03\}$ and successively add a new node connected to previous nodes with probability p. For the BA graph model, we start with a BA graph with $n = 500$ nodes generated with degree $k = 2$ (new nodes are connected to k existing nodes) and then we successively add a new node of degree k connected to the previous ones according the preferential attachement principle. Last, for the GLP graph model, we start from a GLP graph whose largest biconnected component has $n = 500$ nodes, generated with parameters $(m_0, m, \beta, p) \in \{(m_0 = 6, m = 1.13, \beta = 0.6447, p = 0.4695),\ (m_0 = 20, m = 1, \beta = 0.75, p = 0.55)\}$ and then we perform successively growth steps, that is with probability p we add a node and connect it with preferential attachement to one or two existing nodes (growth), or with probability $1 - p$ we add one or two edges between existing nodes (densification). We have computed the value of the hyperbolicity after every addition of 10 vertices. The results are reported in Figure 5.11.

Note that we stopped the experiments after the size of the largest bi-connected component achieved 1000 nodes. This is because the stability of the hyperbolicity already clearly appears in such experiments and experiments on larger size graphs would give similar results. For instance, the hyperbolicity of BA models on 10000 nodes have been shown to be around 3.5 in Figure 3.2a page 17.

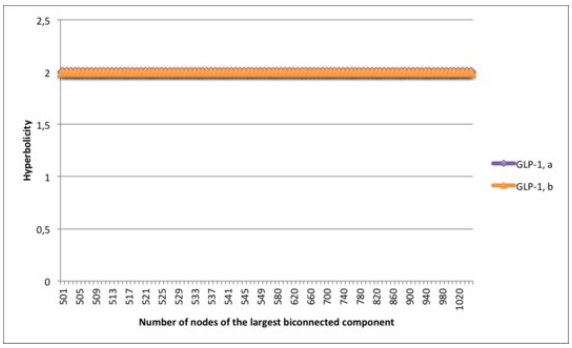
In Figure 5.11a, we observe that the hyperbolicity of BA graphs growth rapidely to 2, and then stays stable. This is compatible with the plots of Figure 3.2a page 17 showing that BA graphs with 1 000 vertices and $k = 2$ have in average hyperbolicity 2. Furthermore, thanks to Figure 3.2a we know that the hyperbolicity of BA graphs would continue to growth with the growth of the network ($3.5 \leq \delta \leq 4$ when $n = 10\,000$). Concerning GNP graphs, it has been observed in [Sha11, NST12] that the hyperbolicity of GNP graphs generated with probability $p = c/n$, where c is a constant, increases with n since the probability of having induced cycles of size up to $O(\log n)$ is positive. In our experiments, reported in Figure 5.11b, the probability p is a constant and so the ratio $p/(c/n)$ increases with n. Indeed, when $n = 1\,000$, we have $(\log_2 n)/n \simeq 0.01$. In addition, it is well known that GNP



(a) BA



(b) GNP



(c) GLP($m_0 = 6, m = 1.13, \beta = 0.6447, p = 0.4695$)

(d) GLP($m_0 = 20, m = 1, \beta = 0.75, p = 0.55$)

Figure 5.11: Evolution of the hyperbolicity with the growth of some graph models.

graphs with $p = O((\log_2 n)/n)$ have a single connected component with low diameter and that the diameter decreases when p increases. Since the hyperbolicity of a graph is upper bounded by half its diameter, we expect the hyperbolicity to decrease when p increases or when p is a constant and n increases. This is exactly what we observe in Figure 5.11b. In Figure 5.11c we observe that the hyperbolicity of GLP graphs generated with parameters ($m_0 = 6, m = 1.13, \beta = 0.6447, p = 0.4695$) is always 2, while the hyperbolicity of GLP graphs generated with parameters ($m_0 = 20, m = 1, \beta = 0.75, p = 0.55$) and reported in Figure 5.11d varies between 1.5 and 2.

Altogether, these experiments show that the observed variations of the hyperbolicity are small compared to the variations that could theoretically occur.

5.2.4 Short term evolution of the hyperbolicity

Next we evaluate the evolution of the hyperbolicity of these graph models when removing sets of randomly chosen edges. We consider in particular major failure events resulting in the removal of 1, 5 or 10% of the edges. We have reported in Table 5.6 the evolution of the hyperbolicity for GNP, BA and GLP graph models. Results are averages over 100 experiments (sets of randomly chosen edges). To conduct these experiments, we use the following parameters:

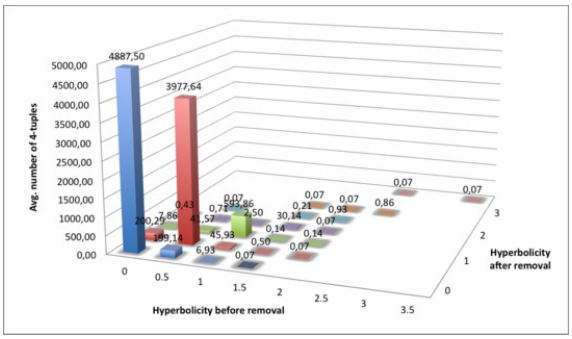
- GNP-1 and GNP-2 are biconnected GNP graphs of order 5 000 generated with probability $p = 0.002$ and 0.005 respectively;

- BA-1 and BA-2 are two BA graphs generated with parameter $k = 2$ (newly added nodes are connected to two nodes). BA-1 has 5 000 nodes and BA-2 10 000 nodes. These graphs are biconnected by construction;
- GLP-1 and GLP-2 are GLP graphs with respectively 15 000 and 18 000 nodes, generated with parameters $m_0 = 6$, $m = 1.13$, $\beta = 0.6447$, and $p = 0.4695$ (See Deliverable D3.1 for more details on the choice of the parameters). The largest biconnected components have respectively 5 029 and 5 836 nodes.
- GLP-3 and GLP-4 are GLP graphs with respectively 20 000 and 25 000 nodes, generated with parameters $m_0 = 20$, $m = 1$, $\beta = 0.75$, and $p = 0.55$. The largest biconnected components have respectively 3 943 and 5 317 nodes.

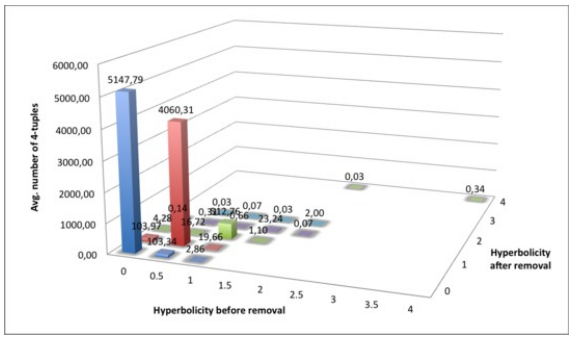
Topology	δ	1% of the edges			5% of the edges			10% of the edges		
		min	mean	max	min	mean	max	min	mean	max
GNP-1	3	3	3	3	3	3	3	3	3	3
GNP-2	2	2	2	2	2	2	2	2	2	2
BA-1	3.5	3.5	3.505	4	3.5	3.56	4	3.5	3.65	4
BA-2	3.5	3.5	3.555	4	3.5	3.8	4	3.5	3.985	4
GLP-1	2	2	2	2	2	2	2	2	2	2
GLP-2	2	2	2	2	2	2	2	2	2	2
GLP-3	2	2	2	2	2	2	2	2	2	2
GLP-4	2	2	2	2	2	2	2	2	2	2

Table 5.6: Evolution of the hyperbolicity when removing 1, 5 or 10% of the edges.

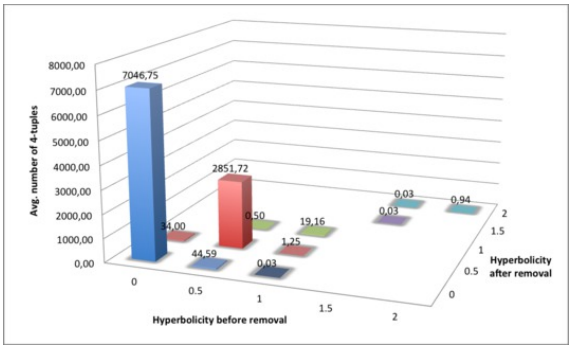
We observe that the hyperbolicity of GNP and GLP graphs are robusts to edge removals and that the hyperbolicity of BA graphs increases in average of less than 0.5. This is due to the fact that between the majority of pairs of vertices, there are many shortest paths. Hence, the removal of few edges do not change their distance. Globally, the all pair shortest paths are stable. To illustrate it, we "followed" a fixed set of 4-tuples and evaluated the evolution of their *hyperbolicity* (i.e., the measure of the hyperbolicity of this 4-tuple) during the sequence of edges removal. More precisely, we did the following experiment: (i) we have chosen randomly 10 000 4-tuples in the largest biconnected component plus the *certificate* for the hyperbolicity of the graph, that is a 4-tuple with maximum hyperbolicity (returned by Algorithm 1); (ii) we removed 100 edges of the graph and compute the new hyperbolicity of the chosen 4-tuples; (iii) repeat edge removals until 15% of the edges have been removed; (iv) plots in Figure 5.12 the average number of 4-tuples going from one given value of the hyperbolicity (axis "hyperbolicity before removal") to a new value (axis "hyperbolicity after removal") between consecutive iterations. It appears that the large majority of this sample keeps the same value (around 99%). Moreover, it is interesting to note that when a variation occurs it is a small one: i.e., the number of tuples whose hyperbolicity is modified by strictly more than 0.5 is very very small. We did the same experiment on CAIDA AS maps of 2004 and 2009 and we observe a similar behavior for these maps and GLP graphs.



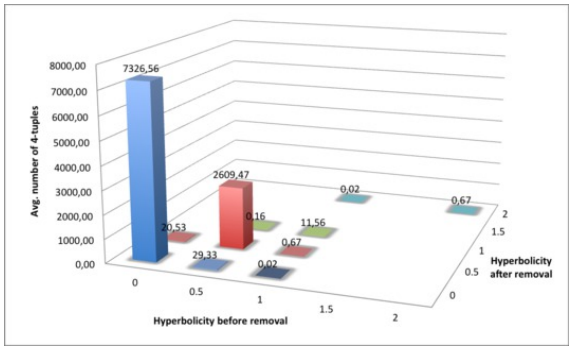
(a) Avg. BA-1



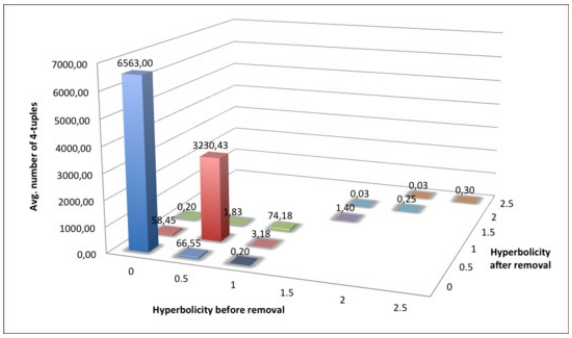
(b) Avg. BA-2



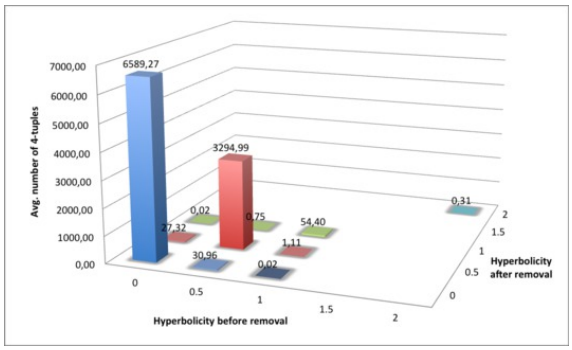
(c) Avg. GLP-2



(d) Avg. GLP-3



(e) CAIDA 2004/01/05



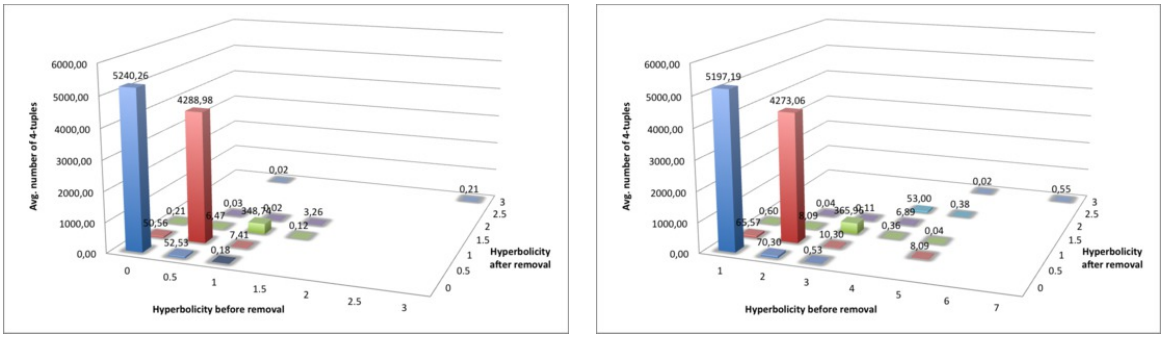
(f) CAIDA 2009/01/05

Figure 5.12: Evolution of the hyperbolicity of 4-tuples when removing edges

Bipartite model. Last, we did the same experiments on graphs produced using the bipartite graph model presented in Section 4 page 25. We have reported in Table 5.7 the hyperbolicity of these graphs which is in average 0.5 higher that CAIDA maps. Then, we have reported in Figure 5.13 the evolution of the hyperbolicity of randomly chosen 4-tuples when removing up to 15% of the edges. Globally, these graphs are very similar to BA, GLP and CAIDA maps with respect to hyperbolicity.

Instance name	δ
PROJ_BIP_BCOMP_collect_2006-09-07	6.0
PROJ_RANDOM_BIP_BCOMP_collect_2006-09-07-i, $i \in [1..6] \cup [8..10]$	3.0
PROJ_RANDOM_BIP_BCOMP_collect_2006-09-07-7	3.5
ADJ_MOLLOYREED_PROJ_BIP_BCOMP_collect_2006-09-07-i, $i \in [1..6] \cup [8..10]$	3.0
ADJ_MOLLOYREED_PROJ_BIP_BCOMP_collect_2006-09-07-7	3.5

Table 5.7: Hyperbolicity of UPMC maps



(a) PROJ_RANDOM_BIP_BCOMP_collect_2006-09-07-1

(b) ADJ_MOLLOYREED_PROJ_BIP_BCOMP_collect_2006-09-07-1

Figure 5.13: Evolution of the hyperbolicity of 4-tuples when removing edges

Chapter 6

Impact of edge deletions on Routing/Forwarding paths

In previous chapter, we have given an overview of the evolution of structural properties of Internet-like networks when they are subject to addition/removal of links/nodes. This gives us a good understanding on the long/short term evolution of these properties. In particular, we have seen that several global properties of Internet-like graphs are not affected by the short/long term evolution (modeled by the removal or addition of links/nodes), however local variations may have strong consequences on the routing of packets. For instance, while the distance distribution of the CAIDA maps remains unchanged, the existing paths such as the paths stored in routing tables actually evolve. Therefore, in this chapter we focus on the impact of these evolutions on routing.

We consider different levels of paths that may be impacted due to the topology evolution.

Topological path: (loopfree) path defined between pair of vertices (u, v) in a graph G;

Routing path: topological path between two nodes as computed by a routing function on the topology and/or distance information. The set of routing paths computed on a given topology defines the routing topology.

Forwarding path: topological path actually followed by incoming message guided by the local decision as determined by the routing function.

For instance, links deletion due to some failures in the AS network may impact the routing in different ways. First, the information stored in the routing tables (following BGP) may become out of date. Second, the paths actually followed by messages using the (out-dated) stored information may become faulty, either by creating loops or increasing drastically the length of forwarding paths. In Section 6.1, we provide a theoretical analysis of such a behavior. After the removal of some edges, routing tables that have become inaccurate are called *liars* (following the terminology of [HKK04, HKKK08, HIKN10]). The main studied question is to establish the relationship between the number of removed links and the number of liars (i.e., of faulty routing paths) that appear.

In Section 6.2, we focus on the impact of edges deletion in greedy routing algorithm. More precisely, we consider the embedding of CAIDA maps into Euclidean metric space, guided by the embedding of spanning trees. In this context, we evaluate the percentage of nodes that

become disconnected (around 15%-20%) after edge removals. Among the remaining nodes, we show that the greedy algorithm achieves almost optimal performance with respect to the (average) stretch.

6.1 Impact of Edge Deletions on shortest path Routing Tables

In this section, we deal with an error model in distributed networks. For a target t, every node is assumed to give an advice, i.e., to point to a neighbor that take closer to the destination. Any node giving a bad advice is called a *liar*. Starting from a situation without any liar, we study the impact of topology changes on the number of liars. More precisely, we establish a relationship between the number of liars and the number of distance changes after one edge deletion. Whenever ℓ deleted edges are chosen uniformly at random, for any graph with n nodes, m edges and diameter D, we prove that the expected number of liars and distance changes is $O(\frac{\ell^2 \cdot D \cdot n}{m})$ in the resulting graph. The result is tight for $\ell = 1$. For some specific topologies, we give more precise bounds. These results have been presented in [GIH11].

6.1.1 Objectives and Motivation

Nowadays, in a communication network, a corresponding situation can occur. Let us consider the routing task. Due to its dynamicity (change of topology, time required to update local information) and its large-scale size, current networks are not immune to faults and crashes. It is no more realistic to blindly trust the data stored locally at each node. For instance, the Border Gate Protocol (BGP) used in Internet to route messages between autonomous systems implicitly assumes that some paths are known to reach any target. Ideally, these paths are as short as possible. Unfortunately, many messages do not reach their destination because no paths are temporally known although some paths could exist. Is there a way to find such paths ?

In the following, for a given target t, we informally refer to a ***liar*** as a node containing bad information about the location of t. A series of papers [HKK04, HKKK08, HIKN10] tackle the problem of locating a target (node, resource, data, ...) in presence of liars.

A first model was introduced by Kranakis and Krizanc [KK99]. They designed algorithms for searching in distributed networks having the ring or the torus topology, when a node has a constant probability of being a liar. A more realistic model was proposed by Hanusse *et al.* [HKK04]: the number of liars is a parameter k and during a routing query, the information stored at every node is unchanged. The main performance measure is the number of edge traversals during a request. Several algorithms, either generic or dedicated to some topologies, and bounds are presented in [HKK04, HKKK08, HIKN10] and are typically of the form $O(\text{dist} + k^{O(1)})$ (for path,grids, expanders,...) or $\Theta(\text{dist} + 2^{O(k)})$ for bounded degree graphs, dist being the distance between the source and the target.

In these papers, there is an implicit assumption: the number of liars is small. Our goal is to evaluate whether this is realistic or not. Starting from a network without any liar, we aim at estimating bounds on the number of liars obtained after few changes of topology. It turns out that this problem is related to the problem of estimating the number of distance changes after few edge/node deletions or insertions. In this section, we focus on edge deletions for the following reasons: it is a more atomic event than node deletion (any node deletion can be represented as a sequence of edge deletions) and a deletion is much more dramatic than an insertion in our context. On the one hand, after one deletion, there is potentially no *known*

or existing path toward the target and on the other hand, after one insertion, we could only miss a shortcut.

6.1.2 Related works

The influence of topology changes on graph parameters is studied in several works. In [CG84, SBvL87], it is proved that for any sequence of ℓ edge deletions that do not disconnect the graph, the diameter D of any unweighted graph turns to be less than $D(\ell + 1)$. Our work is also related to the computation of the most vital node of a shortest path [NPW03], that is the node whose removal results in the largest increase of the distance for a given pair of source/target, and the Vickrey pricing of edges [HS01].

Recently, some work on *dynamic* data structures for shortest paths/distance computation problems has been proposed. By dynamic, we mean that the data structures can tolerate some topology changes in a given network. A dynamic network model defines how the underlying graph changes/evolves over time. More precisely, the following type of models are usually considered:

- *Evolving models without constraint*: it consists in an "online" insertion and/or suppression of links and/or nodes. Roughly speaking, if $G(t)$ is the network at time t then $G(0)$ and $G(t)$ can be quite different.
- *Failure model*: $G(t)$ is a subgraph of $G(0)$. In practice, we consider that few nodes/links are removed from $G(0)$.

The most standard model of dynamic network is the following: starting from an initial graph, a sequence of ℓ insertions/deletions of edges/nodes is done. Each query has to be answered taking into account the ℓ updates. The most naive solution consists in recomputing all shortest paths after any update but it is generally quite costly. For instance, the update time of the fastest dynamic algorithms for the all-pairs shortest path takes $O(n^2 \cdot \text{polylog}(n))$ [DI04, Tho04]. It turns out that in the failure model, it is not always necessary to recompute all shortest paths. Some solutions provide efficient data structures dedicated to the problem of reporting shortest path or distance queries for $\ell = 1$. More precisely, we can distinguish data structures dedicated to *exact* solution [DTCR08, BK09] or *constant approximation of the solution* [KB10, CLPR10], that is a constant factor of shortest path/distance after one edge/node deletion. The challenge is to handle efficiently more than $\ell > 1$ updates. To our knowledge, the more general result is the ℓ-sensitivity distance [CLPR10] oracle for which a data structure of size $O(\ell \cdot s \cdot n^{1+1/s} \log n)$ is able to approximate the distance between any node pairs within a factor $O(s \cdot \ell)$ for undirected graphs in $O(\ell \log^{O(1)} n)$ time. Note that the data structures report distances / routing paths, given the knowledge of the ℓ nodes/edges to avoid. They provide a similar result for weighted graphs and, only if $\ell \leq 2$, for compact routing.

In these works, the implicit model is the one of a *strong adversary model*: the worst sequence of updates. This is sometimes too pessimistic to explain and to model macroscopic observations done on real dynamic networks. In the following, we will also consider the *average adversary model*: any sequence of ℓ updates has the same probability to occur. Estimating the number of distance changes in a dynamic network can be used to get a tight analysis of the update time. In King's algorithm analysis ([Kin99] - section 2.1 or [Ber09]), the update time to maintain a shortest path tree turns to be $O(D \cdot \#\text{number of distance changes from the root})$

for connected bounded degree graphs whenever $\ell = 1$. Our results allow to analyze the average case.

6.1.3 Contributions and Methodology

Models. The network is modeled by a graph $G = (V, E)$ of $|V| = n$ nodes and $|E| = m$ edges. G is assumed to be connected and unweighted. The neighbourhood of vertex u is noted $\Gamma(u)$ and includes u itself. Given a target located at a node t, each node $u \in V \setminus \{t\}$ has an *advice* $\mathsf{Adv}(u) \in \Gamma(u) \setminus \{u\}$. Node u is a *truthteller* if $\mathsf{Adv}(u)$ belongs to a shortest path from u to t and otherwise u is a liar. The set of advice A can also define a directed subgraph of G, noted G_A. There is an arc (u, v) in G_A if and only if $v = \mathsf{Adv}(u)$. Whenever there exists no *liar*, G_A is a shortest path spanning tree rooted at t.

We shall investigate two main parameters:

- the number of liars $k = k_G(A)$ for a set of advice A in graph G
- and the size of the set $\mathcal{S}$ of nodes whose distance to t has changed after one edge deletion.

For instance, in Figure 6.1, we have $n - D$ *lying* nodes pointing toward a dead-end in the rightmost drawing and $D - 1$ nodes whose distance to t has changed after one edge deletion.

Given a graph G *without any liar* and a target t, we aim at analysing the combined effect of the choice of set of advice A and the set of ℓ edges. Note that A is not arbitrary since we assume that G has no liar. After a deletion, it may happen that the resulting graph turns to be disconnected. Nodes that do not belong to the connected component of node t become liars. The set of advice is unchanged with a potential exception: if a deleted edge was used as an advice, one extremity needs to draw another advice among its current neighbours. We focus on two models:

- The adversary model: this model represents a worst-case analysis. An adversary has the capacity of choosing A, the set of edges to remove and the potential new advice to draw. Thus, k is maximal in this model.
- The average/random model: A is assumed to be chosen uniformly at random in the universe of set of advice without liars for the given graph. The set of edges to delete and the potential new advice are chosen uniformly at random.

$\widetilde{G}$ is the resulting subgraph of G after ℓ deletions.

Results. The majority of our results focus on the average model since most of the results in the adversary model are simpler. However, it is interesting to take the two models in order to see a potential gap between them.

More precisely, our main result in the average model is the following: after ℓ deleted edges are chosen uniformly at random, for any graph of n nodes, m edges and diameter D, we prove that the expected number of liars, $\mathbb{E}(k)$, and the expected number of distance changes $\mathbb{E}(|\mathcal{S}|)$ is in $O(\frac{\ell^2 \cdot D \cdot n}{m})$ in the resulting graph.

Tagle 6.1 shows our results after one deletion in both models. Note that the notation $\Theta(\cdot)$ simultaneously stands for a lower bound *and* an upper bound. The lower bound means that *there exists* a graph of the family for which the number of liars is in $\Omega(\cdot)$.

Note that an edge deletion does not necessarily imply the creation of a liar even if some nodes have changed their distance to t (the complete graph is not the only example[1]). Conversely, some liars can appear without any change of distance within the graph.

Topology	Adversary	Average
Graphs of diameter D	$\Theta(n)$	$\Theta(\frac{Dn}{m})$
Square Grid	$\Theta(\sqrt{n})$	$\Theta(1)$
Erdös-Rényi model	$\frac{n-1}{4}+1$	$\Theta(\frac{1}{n})$
Hypercube	$\log n - 1$	$\Theta(\frac{1}{\log n})$

Table 6.1: Number of liars induced by a single edge deletion

For the family of graphs of diameter D, it is easy to reach the bound for the adversary model : just take a path of D nodes and add a star of $n-D$ leaves to one extremity. If t is located to the other extremity, one edge deletion can disconnect the graph implying k and $|\mathcal{S}|$ to be of linear size. Even if somebody would restrict edge deletion to connected graphs, we can easily claim a lower bound of $\Omega(n-D)$ (see Figure 6.1).

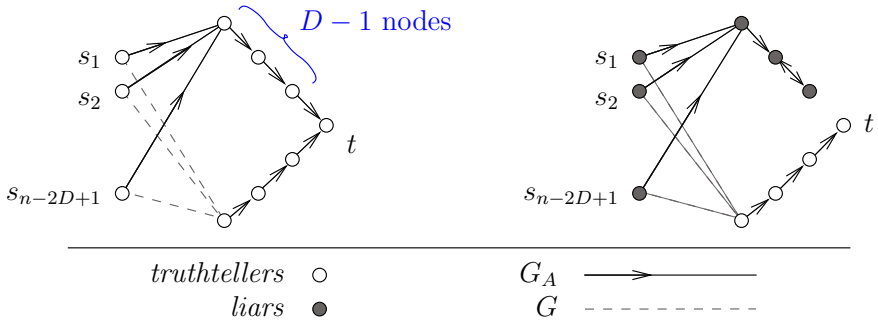


Figure 6.1: An example of an edge deletion that creates $n-D$ liars

The remaining part of this section is structured as follows. We start by exhibiting a relationship between the number of distance changes and the number of liars induced by an arbitrary edge deletion (Lemma 13). Then, we prove that, in the average model, $\mathbb{E}(|S|) \leq \frac{Dn}{m}$. Combining with Lemma 13, we show that $\mathbb{E}(k) < 2D$ (Theorem 17). This result is then improved (Theorem 19) and generalized to ℓ edge deletions (Theorem 22) . More precisely, we prove that the deletion of ℓ random edges creates at most $O(\frac{\ell^2 \cdot n \cdot D}{m})$ liars. In the last section, we give more precise bounds for specific topologies (see Table 6.1).

[1]In the complete graph, for every source node there is only one disconnecting edge. If this edge is removed then every possible advice is correct (point toward a node at distance 1 of the destination)

6.1.4 General Results

We start by presenting some notations and some easy facts used in this work.

Notations.

$\widetilde{G}_e$	G after deletion of edge e, $\widetilde{G}_e = (V, E \setminus e)$, or simply $\widetilde{G}$.
$\text{dist}(u, v)$	distance in G from u to v.
$\text{dist}_{\widetilde{G}}(u, v)$	distance in $\widetilde{G}$ from u to v.
$\Gamma(X)$	X's neighbourhood in G, $\Gamma(X) = \bigcup_{x \in X} \Gamma(x)$
$\mathsf{Adv}^{-1}(X)$	set of nodes advising another node that belongs to X, i.e., $\mathsf{Adv}^{-1}(X) = \{u \in V \mid \mathsf{Adv}(u) \in X\}$
$\mathcal{F}(e)$	indicates if edge $e = \{x, y\}$ belongs to the set of advised edges G_A. More precisely, $\mathcal{F}(e) = 1$ if $\mathsf{Adv}(x) = y \vee \mathsf{Adv}(y) = x$ and $\mathcal{F}(e) = 0$ otherwise.

Many of our proofs are based on the notion of (s, t)-disconnecting edges:

Definition 11. *An edge $\{x, y\}$ is (s, t)-disconnecting if it belongs to* all *shortest paths from s to t*

The deletion of a (s, t)-disconnecting edge implies

$$\text{the event } \mathcal{E}_{s,t} : \text{dist}_{\widetilde{G}}(s, t) > \text{dist}(s, t) \tag{6.1}$$

Otherwise, there exists a shortest path from s *to* t which does not contain $\{x, y\}$. The set of disconnecting edges from s to t is denoted $\mathcal{C}_{s,t}$. It follows that

Lemma 12. *The distance from s to t is modified by a single edge deletion if and only if this edge belongs to $\mathcal{C}_{s,t}$.*

6.1.4.1 Relationships between the number of liars and the number of distance changes

Let us denote $\mathcal{S} = \mathcal{S}_t^e = \{s \in V \mid \text{deletion of } e \Rightarrow \mathcal{E}_{s,t}\}$ the set of nodes that have changed their distance to t after the deletion of some edge e.

Lemma 13. *In any graph containing k' liars, the number of liars k after deletion of an edge e always satisfies*

$$\left|\mathsf{Adv}^{-1}(\mathcal{S}) \setminus \mathcal{S}\right| \leq k \leq \left|\mathsf{Adv}^{-1}(\mathcal{S})\right| + \mathcal{F}(e) + k' \tag{6.2}$$

Proof. In any graph with k' liars, after one edge deletion, we study the impact for every node (i.e., advice) on the resulting number of liars k. For every node u with $v \in V, \mathsf{Adv}(u) = v$, we have :

$$\text{dist}_G(u, t) - \text{dist}_G(v, t) \in \begin{cases} \{1\} & \text{if } u \text{ is a truthteller} \\ \{0, -1\} & \text{if } u \text{ is a liar} \end{cases}$$

If $u \notin \mathcal{S}$ and $v \in \mathcal{S}$ then

$$\text{dist}_{\widetilde{G}}(u, t) - \text{dist}_{\widetilde{G}}(v, t) \begin{cases} \{0, -1\} & \text{if } u \text{ was a truthteller} \\ \{-1\} & \text{if } u \text{ was a liar} \end{cases}$$

hence u becomes (or remains) a liar. The minimum number of liars added by the deletion is then

$$k \geq \left|\mathsf{Adv}^{-1}(\mathcal{S}) \setminus \mathcal{S}\right|$$

Let us now consider the upper bound. First assume that the removed edge $e \neq \{u, v\}$. If $v \notin \mathcal{S}$ then u remains a liar if it was :

- $u \in \mathcal{S}$ and $v \notin \mathcal{S}$ then :

$$\mathrm{dist}_{\widetilde{G}}(u,t) - \mathrm{dist}_{\widetilde{G}}(v,t) \in \begin{cases} [2,\infty] & \text{(impossible}^2\text{)} \quad \text{if } u \text{ was a truthteller} \\ \{0,1\} & \text{(could be a liar)} \quad \text{if } u \text{ was a liar} \end{cases}$$

- if $u \notin \mathcal{S}$ and $v \notin \mathcal{S}$ then $\mathrm{dist}_{\widetilde{G}}(u,t) - \mathrm{dist}_{\widetilde{G}}(v,t) = \mathrm{dist}_G(u,t) - \mathrm{dist}_G(v,t)$.

If $u \in S$ and $v \in S$, then $\mathrm{dist}_{\widetilde{G}}(u,t) - \mathrm{dist}_{\widetilde{G}}(v,t) \in \{1, 0, -1\}$, so u could be a liar or not independently of its previous state. So, the maximum number of liars added by one edge deletion is

$$k \leq \left|\mathsf{Adv}^{-1}(\mathcal{S})\right|$$

If $\mathsf{Adv}^{-1}(\mathcal{S})$ was not containing any liar then the resulting number of liars after deletion is

$$k \leq \left|\mathsf{Adv}^{-1}(\mathcal{S})\right| + k'$$

Finally, if the removed edge $e = \{u, v\}$, i.e., $\mathcal{F}(e) = 1$, then u has to change its advice and become a liar,

$$\left|\mathsf{Adv}^{-1}(\mathcal{S}) \setminus \mathcal{S}\right| \leq k \leq \left|\mathsf{Adv}^{-1}(\mathcal{S})\right| + \mathcal{F}(e) + k'$$

□

6.1.4.2 Upper bounds for $\ell = 1$ deleted edge

According to our model, and as we have already seen in Lemma 13, liars apparition is due to distance changes and advice deletion.

Number of distance changes in the average model

Lemma 14. *In any m-edge graph $G = (V, E)$, if an edge, chosen uniformly at random, is removed from E then the number $|S|$ of distance changes satisfies*

$$\forall t \in V : \mathbb{E}(|\mathcal{S}|) = \frac{1}{m} \sum_{s \in V \setminus \{t\}} |\mathcal{C}_{s,t}|. \tag{6.3}$$

Proof. From Lemma 12, if edge $\{x, y\}$ is chosen uniformly at random in E then $\forall s \in V$:

$$\mathbb{P}(\mathcal{E}_{s,t}) = \frac{|\mathcal{C}_{s,t}|}{m}$$

Let $X_{s,t}$ be a random variable defined by $X_{s,t} = 1$ if $\mathcal{E}_{s,t}$, and $X_{s,t} = 0$ otherwise. We get

$$\mathbb{E}(|\mathcal{S}|) = \mathbb{E}(\sum_{s \in V \setminus \{t\}} X_{s,t}) = \sum_{s \in V \setminus \{t\}} \mathbb{E}(X_{s,t}) = \sum_{s \in V \setminus \{t\}} \mathbb{P}(\mathcal{E}_{s,t}) = \frac{1}{m} \sum_{s \in V \setminus \{t\}} |\mathcal{C}_{s,t}|$$

□

[2] impossible because u and v are neighbours

Corollary 15. *For any n-node, m-edge graph of diameter D, after one random edge deletion, we have in the average model*

$$\mathbb{E}(|\mathcal{S}|) \leq \frac{D(n-1)}{m}. \tag{6.4}$$

Proof. In a graph of diameter D, by definition, all shortest paths lengths are at most D. So, $\forall s \in V \setminus \{t\}$, there is at most D (s,t)-disconnecting edges in E. □

Number of liars in the average model Applying Lemma 13, we get

Corollary 16. *For any n-node, m-edge graph of diameter D and maximal degree Δ without liar, after one random edge deletion, we have $\mathbb{E}(k) \leq \frac{(D\Delta+1)(n-1)}{m}$.*

This turns to be optimal up to a constant factor for bounded degree graphs (see Theorem 20). However, this is not the case whenever the graph has nodes of unbounded degree.

Theorem 17. *For graphs of diameter D without liar, after random one edge deletion , we have*

$$\mathbb{E}(k) \leq 2D \tag{6.5}$$

Proof. According to Lemma 13, for any edge e, if $|\mathcal{S}^e_{e,t}|$ nodes change their distance to t, then the number of added liars after deletion of edge e, is at most $|\mathsf{Adv}^{-1}(\mathcal{S})| + \mathcal{F}(e) \leq \sum_{s\in\mathcal{S}} |\mathsf{Adv}^{-1}(s)| + \mathcal{F}(e)$.

Take the possible m edge deletions trials and consider the m corresponding sets $\mathcal{S}_i$ for i going from 1 to m. In a given trial in which event $\mathcal{E}_{s,t}$ occurs, each node s adds at most $|\mathsf{Adv}^{-1}(s)| \leq \deg(s)-1$ liars (excluding itself) since G contains initially no liar and at least one neighbour of s is closer to t than s. Since $\forall s \in V\setminus\{t\}$, event $\mathcal{E}_{s,t}$ can occur in at most $|\mathcal{C}_{s,t}| \leq D$ instances among the m ones. It follows that for given s, $\sum_{i:s\in\mathcal{S}_i} |\mathsf{Adv}^{-1}(s)| \leq D(\deg(s)-1)$.

Thus, for any $i \in [1,m]$, we have $k_{\widetilde{G}_i} \leq |\mathsf{Adv}^{-1}(\mathcal{S}_i)| + \mathcal{F}(e) \leq \sum_{s\in\mathcal{S}_i} \deg(s)$. Summing over all values of i, we get

$$\sum_{i=1}^{m} k_{\widetilde{G}_i} \leq \sum_{i=1}^{m} \sum_{s\in\mathcal{S}_i} \deg(s) = \sum_{s\in V\setminus\{t\}} \sum_{i:s\in\mathcal{S}_i} \deg(s) \leq \sum_{s\in V\setminus\{t\}} D\cdot\deg(s) = 2m\cdot D$$

It turns out that $\mathbb{E}(k_{\widetilde{G}}) \leq \frac{2mD}{m} = 2D$. □

A more precise bound can be found by reasoning on a hierarchical cutaway of G from distance 0 to D with respect to target t. The following part shows a detailed proof based on this principle to get a tighter upper bound ($\leq D\cdot n/m$).

Nodes in danger Let $\mathcal{T}_{u,t}$ be the set of nodes that have at least one shortest path to t through $u \in V$ (see Figure 6.2). Let $L_i = \{u \in V \mid \mathrm{dist}(u,t) = i\}$ be the set of nodes at distance i from t. Every node $v \in \mathcal{T}_{u,t}$ with $u \in L_i$ is *in danger*[3] with respect to level i if and only if only one shortest path from u to t exists. In Figure 6.2, all nodes from sets $\mathcal{T}_{u_2,t}$ and $\mathcal{T}_{u_3,t}$ are in danger.

[3] can potentially turns into a liar

Distances and shortest paths Let $\mathcal{C}_i = \{\{x, y\} \mid x \in L_i, y \in L_{i-1} \wedge \Gamma(x) \cap L_{i-1} = \{y\}\}$ be the set of disconnecting edges between L_i and L_{i-1}. Let $B_t(i-1)$ be the set of nodes at distance at most $i-1$ from t. If G is not connected then the set of edges that does not belong to the connected component of t is $\mathcal{C}_\infty$.

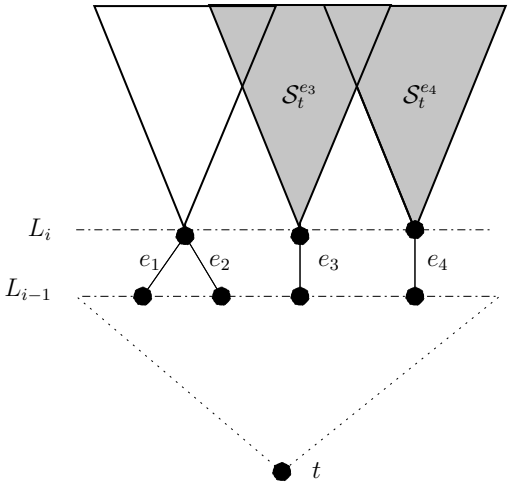


Figure 6.2: G levels and nodes in *danger* (grey filled areas).

Lemma 18. *For any graph, without any assumption on the number of liars, if the edge $\{x, y\}$ is deleted and $i \leq D$ then*

$$\mathbb{E}(k \mid \{x, y\} \in \mathcal{C}_i) \leq \frac{n - |B_t(i-1)|}{|\mathcal{C}_i|} \tag{6.6}$$

and $\mathbb{E}(k \mid \{x, y\} \in \mathcal{C}_\infty) = 0$.

Proof. The number of disconnecting edges between L_i and L_{i-1} is

$$|\mathcal{C}_i| = |\{x \in L_i, |\Gamma(u) \cap L_{i-1}| = 1\}|$$

The average number of liars added by a deletion between levels L_i and L_{i-1} is at most[4]

$$\mathbb{E}(k \mid \{x, y\} \in \mathcal{C}_i) \leq \frac{|\bigcup_{x \in L_i} \mathcal{T}_{x,t}|}{|\mathcal{C}_i|} \leq \frac{n - B_t(i-1)}{|\mathcal{C}_i|}$$

□

Theorem 19. *For $D \geq 2$, the numbers of liars added by deleting an edge chosen uniformly at random in E is*

$$\mathbb{E}(k) \leq \frac{D(n - \frac{D-1}{2})}{m} \leq \frac{D(n-1)}{m} \tag{6.7}$$

For $D = 1$, $\mathbb{E}(k) = k = 0$. This result holds for arbitrary graphs, unnecessarily connected.

[4]some already lying nodes could belong to $\bigcup_{x \in L_i} \mathcal{T}_{x,t}$, these liars will be counted twice

Proof. The expected average number of liars is the sum of the expected number of liars induced by deletions between every levels $L_1, L_2, \ldots, L_D$

$$\mathbb{E}(k) = \sum_{i=1}^{\infty}(\mathbb{E}(k \mid \{x,y\} \in \mathcal{C}_i) \times \mathbb{P}(\{x,y\} \in \mathcal{C}_i)) = \sum_{i=1}^{D}(\mathbb{E}(k \mid \{x,y\} \in \mathcal{C}_i) \times \mathbb{P}(\{x,y\} \in \mathcal{C}_i))$$

The probability of deleting an edge at level i is

$$\mathbb{P}(\{x,y\} \in \mathcal{C}_i) = \frac{|\mathcal{C}_i|}{m}$$

Thus,

$$\mathbb{E}(k) \leq \sum_{i=1}^{D} \frac{n - |B_t(i-1)|}{|\mathcal{C}_i|} \times \frac{|\mathcal{C}_i|}{m} \leq \frac{Dn}{m} - \frac{1}{m}\sum_{i=1}^{D}|B_t(i-1)|$$

$\forall i \in D, |B_t(i-1)| \geq i-1$, hence, the average number of liars added is

$$\mathbb{E}(k) \leq \frac{Dn}{m} - \frac{D(D-1)}{2m} \leq \frac{D(n - \frac{D-1}{2})}{m}$$

□

6.1.4.3 Lower bound

Theorem 20. *For any integers n, m, D such that $m \geq n \geq 2D \geq 20$,*

- *there exists a graph of $n+O(1)$ nodes, $\Theta(m)$ edges and diameter D for which the expected number of liars after a random edge deletion is greater than $\frac{(D-8)n}{32m}$.*
- *there exists a graph of $\Theta(n)$ nodes, $\Theta(m)$ edges and diameter D for which the expected number of distance changes after a random edge deletion is $\Omega(\frac{Dn}{m})$.*

Proof. Let us consider a graph H (see H_1 in Figure 6.3) built in the following way: take a complete graph of size r and a stable of size r'. Add two extra nodes u, v and link them to the $r + r'$ nodes. This graph has diameter 2, $r + r' + 2$ nodes and $\frac{r(r-1)}{2} + 2(r + r')$ edges. Take now four copies of H named H_1, H_2, H_3 and H_4. For i going from 1 to 4, link u_i to $v_{(i \mod 4)+1}$ by a path of $D/2 - 4$ edges. The resulting graph G has diameter D. We set up $r = \left\lceil \sqrt{\frac{m-D}{2}} \right\rceil$ and $r' = \left\lceil \frac{n-D}{4} - r \right\rceil$. It follows that G has $n + O(1)$ nodes. The total number of edges is $\Theta(n)$. This graph is presented in Figure 6.3.

Without loss of generality, assume now that target t is either between u_1 and v_2 or belongs to H_1. In the first case, it follows that every node of H_3 (excluding v_3 and potentially u_3) has v_3 as advice toward t. The probability that the deleted random edge belongs to the path from u_2 to v_3 is $p = \frac{D-8}{2m}$. The expected number of liars/distance changes is at least $p(r + r') \geq \frac{(D-8)n}{16m}$.

For the second case, every node of H_3 excluding u_3 or v_3 can point arbitrarily to u_3 and v_3. Take the node given by the majority. If v_3 (resp. u_3) is chosen, then p corresponds to the probability that the deleted random edge belongs to the path from u_2 to v_3 (resp. v_3 to u_4). The expected number of liars turns to be greater than $p(\frac{r+r'}{2}) \geq \frac{(D-8)n}{32m}$.

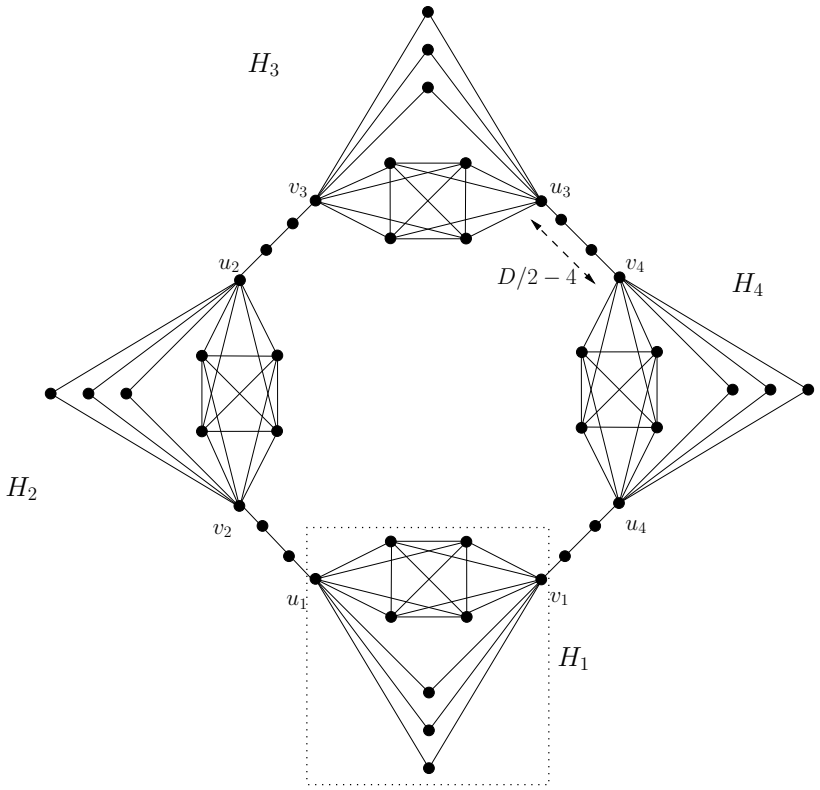


Figure 6.3: Sample graph in which the lower bound is reached.

In this last case, in order to get a similar lower bound for the expected number of distance changes, we just have to slightly modify each H_i copy. We just substitute each node of the stable set by an edge between two nodes. Each copy turns to have $r + 2r'$ nodes and $\frac{r(r-1)}{2} + 2r + 3r'$ edges. We only have to consider the distance change from t and r' nodes of this new set. To have $r' = \Theta(n)$, we might have to consider a graph G with $\Theta(n)$ nodes (at most $2n$ is enough). □

6.1.5 Number of liars after ℓ deletions

Lemma 21. *After ℓ edge deletions in any graph G of diameter D, every connected component of the resulting graph have diameter at most $D(\ell + 1)$.*

Proof. As claimed in [SBvL87], given ℓ, the maximum diameter of the graph obtained by deleting ℓ edges from a graph G of diameter D is $D(\ell+1)$, assuming that the resulting graph is still connected. Now, if a single deletion disconnect in two parts a connected component of diameter D, both resulting components will have diameter at most D. So, after $\ell + 1$ deletions, any connected component has at most diameter $D(\ell + 1)$. □

Theorem 22. *Let G be a n-nodes, m-edges graph of diameter D without any liars. For any $\ell \leq m$, the deletion of ℓ edges chosen uniformly at random in G creates an expected number of liars of $O(\min(\frac{\ell^2 \cdot D \cdot n}{m}, n))$.*

Proof. As stated in Theorem 19, deleting one edge into a graph of diameter D without liars creates an average of at most $D(n-1)/m$ liars. From Lemma 21, the deletion of ℓ edges

creates

$$\mathbb{E}(k) \leq \sum_{i=1}^{\ell} \frac{Di(n-1)}{m-(i-1)} \leq \sum_{i=1}^{\ell} \frac{Di(n-1)}{m-(\ell-1)} \leq \frac{D(n-1)}{m-(\ell-1)} \sum_{i=1}^{\ell} i$$

or

$$\mathbb{E}(k) \leq \frac{D(n-1)}{m-(\ell-1)} \times \frac{\ell(\ell-1)}{2}$$

Since the number of liars could not exceed the number of nodes, the expected number of liars is $O(\min(\frac{\ell^2 \cdot D \cdot n}{m}, n))$. □

6.1.6 Specific Topologies

In this section, we show how tight the bounds are for some specific topologies. We just briefly describe the sketch of proofs. The study gives a justification for the introduction of the adversary model. In order to get tight bounds in the average model, we exhibit the worst configurations of advice and evaluate their probabilities in the average model.

Theorem 23. *In the adversary model,*

- $k = \Theta(n)$ *for Erdös-Rényi's random graphs with parameter* $p = 1/2$*;*
- $k = \Theta(\sqrt{n})$ *for square grids;*
- $k = \log_2 n - 1$ *for hypercube.*

In the average model,

- $k = \Theta(1/n)$ *for Erdös-Rényi's random graphs with parameter* $p = 1/2$*;*
- $k = \Theta(1)$ *for square grids;*
- $k = \Theta(1/\log n)$ *for hypercube.*

Here is some clue about the behaviour of the different graph families in the adversary model :

- Erdös-Rényi's random graphs: each pair of nodes is connected with probability p. For $p = 1/2$, almost all graphs have diameter 2. If the deleted edge is between L_1 and L_2 then only 1 node can turn into a liar. However, a deletion between $L_0 = \{t\}$ and L_1 can create $\Theta(n)$ liars since on average, there are $(n-1)/4$ neighbours in L_2 of any individual node of L_1.
- grids: only nodes that share a coordinate (same row or column) with t have (s,t)-disconnecting edges and thus can change their distance to t. The number of distance changes is then $|\mathcal{S}| = \Theta(\sqrt{n})$ for square grids. An adversary can force all neighbours of $\mathcal{S}$ to point to $\mathcal{S}$. From Lemma 13, we get that $k = \Theta(\sqrt{n})$.
- hypercube: only target's neighbours can increase their distance to t after one edge deletion, so $|\mathcal{S}| \leq 1$ and only $k \leq \log_2 n - 1$ nodes of level L_2 can become liars.

In order to get tight bounds for the average model, we simulate the m possible edge deletions and average k:

- Erdös-Rényi's random graphs: only edges leading to advice deletion can create liars. Condition on this event, on average, only $\Theta(1)$ liars appear. However, this event occurs with probability $\Theta(1/n)$. In the other cases, no liar are obtained.
- grids: with probability $1 - \Theta(1/\sqrt{n})$, there is no (s,t)-disconnecting edge between a random node and t. It follows that, with probability $1-\Theta(1/\sqrt{n})$, we have at most one new liar (if the deleted edge contains an advice). With probability $\Theta(1/\sqrt{n})$, we have $\Theta(\sqrt{n})$ liars.
- hypercube: only edges leading to an advice deletion or being neighbours of t can create liars. However neighbours of t can not become liars. For nodes of levels $L_{i\geq 2}$, there is no distance change after one edge deletion. Since $\mathbb{E}(\mathcal{F}(e)) = \frac{n-1}{n\log_2 n} = \Theta(1/\log n)$, we have $\mathbb{E}(k) = O(1/\log n)$. To get a lower bound of $\Omega(1/\log n)$, we just have to consider the $n/2$ closest nodes from t. The probability that the deleted edge is linked to one of these nodes is at least $1/2$ and condition on this event, with probability at least $\frac{1}{2\log_2 n}$, a new advice is required and create a liar.

6.1.7 Conclusion

This work shows the importance of the diameter for the number of distance changes and liars appearances in a dynamic graph model. Of course, it would be interesting to consider edge/node addition. Contrary to edge deletion, an edge addition can drastically change the distance within the graph. Even for grids, the number of distance changes would be $\Omega(n)$ after a random edge addition.

6.2 Greedy routing and embeddings

In this section, we describe recent theoretical and experimental work on greedy routing and embeddings. Parts of these results have been presented in [CK12] and [Kal12].

6.2.1 Objectives and Motivation

Greedy routing utilizes a particular assignment of node addresses so that routing of packets can be performed using only the address of the current node of a traveling packet, the addresses of its neighbors, and the address of the destination node. The node addresses are usually defined using a *greedy embedding*. Formally, a greedy embedding of a graph $G = (V, E)$ into a host metric space (X, dist) is a function $f : V \to X$ so that the following property holds: for any two nodes u, t of G, there exists a node v in the neighborhood $\Gamma(u)$ of u in G so that $\text{dist}(f(v), f(t)) < \text{dist}(f(u), f(t))$. A typical example of a host metric space is the d-dimensional Euclidean space $\mathbb{R}^d$ equipped with the Euclidean distance ℓ_2. Given a greedy embedding f, the coordinates of point $f(u)$ can be used as the address of node u. Then, when node u has to take a decision about the next hop for a packet with destination address $f(t)$, it has to select among its neighbors a node with address $f(v)$ such that $\text{dist}(f(v), f(t)) < \text{dist}(f(u), f(t))$. It is clear that, in this way, the packet is guaranteed to reach its destination within a finite number of steps.

Greedy embeddings were first defined by Papadimitriou and Ratajczak [PR05]. They proved that any 3-connected planar graph can be greedily embedded into the 3-dimensional Euclidean space using a non-Euclidean distance function. They also conjectured that every such graph can be greedily embedded in the Euclidean space with Euclidean distance; this conjecture was later proved by Moitra and Leighton [LM10]. Kleinberg [Kle07] showed that any tree can be greedily embedded in the 2-dimensional hyperbolic space. This immediately yields a greedy embedding for any graph (by just embedding a spanning tree). Epstein and Woodrich [EG11] observed that the coordinates of the nodes in Kleinberg's embedding require too much space and modified it so that each coordinate is represented with $O(\log n)$ bits (where n is the size of the graph). Greedy embeddings into $O(\log n)$-dimensional Euclidean spaces (with ℓ_∞ distance) are also known [Kle07] and exploit an isometric embedding of trees into Euclidean spaces due to Linial et al. [LLR94].

Note that the approach of computing the greedy embedding of a spanning tree ignores several links of the network. Hence, it may be the case that even though a packet could potentially reach its destination with a few hops using a shortest path, it is greedily routed through a path that has to travel across a constant fraction of the nodes of the whole network. The measure that can quantify this inefficiency is the stretch of a greedy routing algorithm, i.e., the ratio between the length of the path used by the algorithm over the length of the corresponding shortest path.

Let us proceed with a formal definition of the stretch of a greedy embedding.

Definition 24. *Let f be a greedy embedding of a graph G into a metric space (X, dist). A path $\langle u_0, u_1, ..., u_t \rangle$ is called a greedy path if $u_{i+1} \in \text{argmin}_{v \in \Gamma(u_i)}\{\text{dist}(f(v), f(u_t))\}$, where $\Gamma(u_i)$ denotes the neighborhood of u_i in G. We say that f has stretch ρ for graph G if, for every pair of nodes u, v of G, the length of every greedy path from u to v is at most ρ times the length of the shortest path from u to v in G. The stretch of f is simply the maximum stretch over all graphs.*

We use the terms no-stretch and optimal stretch to refer to embeddings with stretch equal to 1.

Maymounkov [May06] considers the question of whether no-stretch greedy embeddings into low-dimensional spaces exist. Among other results, he presents a lower bound of $\Omega(\log n)$ on the dimension of the host hyperbolic space for greedy embeddings with optimal stretch. Furthermore, he conjectures that any graph can be embedded into Euclidean or hyperbolic spaces with a polylogarithmic number of dimensions with no stretch. We remark that a proof of this conjecture would probably justify greedy routing as a compelling alternative to compact routing [PU89].

Flury et al. [FPW09] present a greedy embedding of any n-node graph into an $O(\log^2 n)$-dimensional Euclidean space that has stretch $O(\log n)$. Each coordinate in their embedding uses $O(\log n)$ bits. They used the min-max$_c$ distance function which views the d-dimensional Euclidean space as composed by d/c c-dimensional spaces and, for a pair of points x, y, takes the ℓ_∞ norm of the projections of x and y into those spaces, and finally takes the minimum of those ℓ_∞ distances as the min-max$_c$ distance between them. Their embedding uses an algorithm of Awerbuch and Peleg [AP90] to compute a tree cover of the graph and the algorithm of Linial et al. [LLR94] to embed each tree in the cover isometrically in a low-dimensional Euclidean space.

6.2.2 Theoretical investigations

In our recent paper [CK12], we present lower bounds on the number of dimensions required for low-stretch greedy embeddings into Euclidean spaces. We first disprove Maymounkov's conjecture by showing that greedy embeddings into $(\mathbb{R}^d, \ell_2)$ have optimal stretch only if the number of dimensions d is linear in n. The proof uses an extension of the hard crossroads construction in [May06] and exploits properties of random sign pattern matrices. Namely, we make use of a linear lower bound due to Alon et al. [AFR85] on the minimum rank of random $N \times N$ sign pattern matrices. We also obtain an $\Omega(\sqrt{n})$ lower bound through an explicit construction that uses Hadamard matrices.

Furthermore, we present trade-offs between the stretch of greedy embeddings into $\mathbb{R}^d$ and the number of dimensions d for different distance functions. Namely, for every integer parameter $k \geq 3$, we show that greedy embeddings into $\mathbb{R}^d$ with ℓ_p distance have stretch smaller than $\frac{k+1}{3}$ only if $d \in O\left(\frac{n^{1/k}}{\log p}\right)$. This implies that the best stretch we can expect with a polylogarithmic number of dimensions is $\Omega\left(\frac{\log n}{\log \log n}\right)$. Our arguments use a result of Erdös and Sachs [ES63] on the density of graphs of high girth and a result of Warren [War68] that upper-bounds the number of different sign patterns of a set of polynomials. In particular, starting from a dense graph with high girth, we construct a family of graphs and show that, if d is not sufficiently large, any embedding f fails to achieve low stretch for some graph in this family.

We extend our lower bound arguments to greedy embeddings into $\mathbb{R}^d$ that use the min-max$_c$ distance function that has been used in [FPW09]. Note that the lower bound does not depend on any other characteristic of the embedding of [FPW09] and applies to every embedding in $(\mathbb{R}^d, \text{min-max}_c)$. We show that the best stretch we can hope for with $d \in \text{polylog}(n)$ is $\Omega\left(\frac{\log n}{\log \log n}\right)$. This lower bound indicates that the embedding of Flury et al. [FPW09] is almost optimal among all greedy embeddings in $(\mathbb{R}^d, \text{min-max}_c)$. Furthermore, our proof applies to

a larger class of distance functions including ℓ_∞.

6.2.3 Experimental results

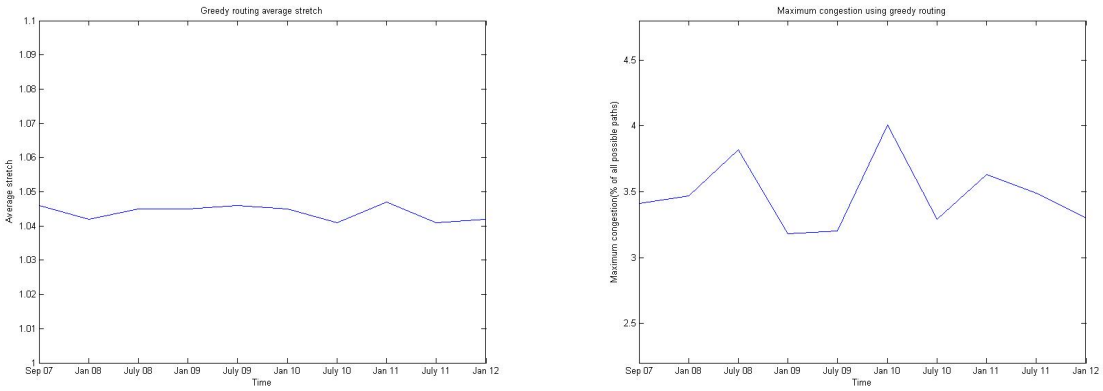
Our recent experimental work on real Internet snapshots [Kal12] comes in sharp contrast with the pessimistic worst-case lower bounds of the previous section. Our work makes extensive use of isometric embeddings of trees into Euclidean spaces, an algorithmic idea that has also been used in [Kle07] and [FPW09]. An isometric embedding of a tree into $(\mathbb{R}^d, \ell_\infty)$ is a mapping of the nodes of the tree into points of $\mathbb{R}^d$ such that the ℓ_∞-distance of any two points is equal with the distance of the corresponding nodes in the tree. Such an embedding can be computed using an algorithm due to Linial et al. [LLR94]. In particular, the algorithm of [LLR94] guarantees that $d \in O(\log n)$, where n is the number of nodes in the input tree. Furthermore, the points computed have integer coordinates. So, our basic routing algorithm is a greedy one using the node addressing computed by the isometric embedding of a spanning tree of the network.

We have tested the basic routing algorithm using data provided by `caida.org`. After processing the data, we have created ten snapshots of the Internet AS-graph that cover the period between September 2007 and January 2012. The size of these snapshots is depicted in the following table and is indicative (but not precise) information for the evolution of the Internet during that period.

snapshot date	#nodes	#edges	av. degree
13/9/2007	12,190	25,822	4.24
2/1/2008	14,038	28,714	3.38
2/7/2008	14,128	29,218	4.14
3/1/2009	15,205	31,328	4.12
2/7/2009	15,427	31,706	4.12
1/1/2010	16,695	33,315	4.00
1/7/2010	17,143	35,765	4.18
2/1/2011	18,427	39,462	4.28
1/7/2011	19,281	41,131	4.26
2/1/2012	20,098	42,338	4.22

It is important to briefly comment on the structure of these graphs since the good performance of our routing algorithms is mainly due to the particular structure of such graphs. Each snapshot consists of a small set of high-degree nodes that are almost fully connected. The rest of the network mainly consists of tree-like structures that are rooted on core nodes while there are additional few edges between different tree-like structures. In our implementations, we have used spanning trees that are rooted in the (core) node of highest degree. Such a spanning tree contains approximately half of the network edges and naturally excludes most of the edges in the core. However, our algorithm extensively uses edges that are not included in the spanning tree and result in efficient routing.

In our experiments, we have mainly measured the stretch of greedy paths and the congestion in the network links assuming uniform (all-to-all) traffic. As it can be seen in Figure 6.4, the average stretch (over all source-destination pairs of nodes) is almost optimal; it is sharply concentrated between 1.04 and 1.05 for all instances. This implies that a limited expansion (i.e., doubling the AS-graph size) has almost no effect to average stretch. In contrast, the



(a) Average stretch

(b) Maximum congestion

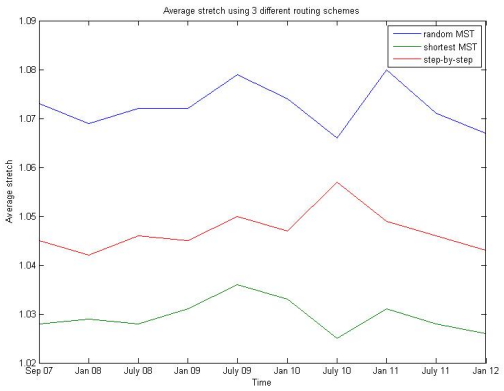
Figure 6.4: Average stretch (left) and maximum congestion (right) resulted by the application of the basic routing algorithm.

results related to congestion are suboptimal. Figure 6.4 indicates that maximum congestion (i.e., the maximum number of greedy paths that cross some link) can be up to 4% of the total (all-to-all) network traffic. Such a single-digit percentage is to be expected due to the network structure but, at first glance, there is room for improvements.

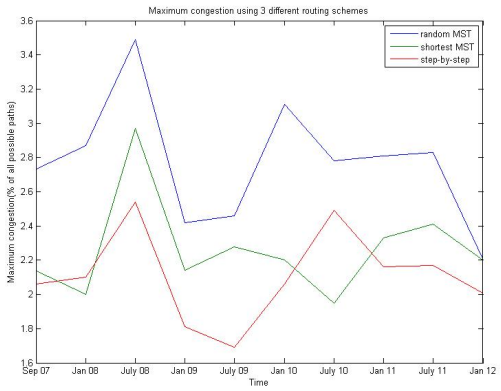
So, we have also considered variations of the basic routing algorithm in order to balance the traffic among the links in the core (and the incoming/outgoing ones). We have implemented three such alternative routing algorithms. Each of them uses several spanning trees and their embeddings into $(\mathbb{R}^d, \ell_\infty)$.

- random ST: Greedy routing is performed using the isometric embedding of a randomly selected spanning tree.
- shortest ST: Greedy routing is performed using the embedding of that spanning tree that minimizes the ℓ_∞-distance between the source and the destination. Ties are broken randomly.
- step-by-step ST: Greedy routing is performed using the embedding of different spanning trees in the following way: for each intermediate node, the next hop is selected using the embedding of that spanning tree that minimizes the ℓ_∞-distance from the destination node. Again, ties are broken randomly.

The experiments with the variations of the basic routing algorithm use four different spanning trees (produced using random breadth-first-search). As Figure 6.5 indicates, a naive use of randomness is not beneficial. The average stretch of random ST is considerably higher (between 1.07 and 1.08) than that of the basic routing algorithm and of the other two variations. Shortest ST outperforms both step-by-step ST and the basic routing algorithm, achieving an average stretch around 1.03. A slightly different picture holds for congestion; the three variations improve the basic routing algorithm rather significantly. Again, random ST is inferior to the other two variations; among shortest ST and step-by-step ST, the latter leads to better congestion for most (8 out of 10) instances.



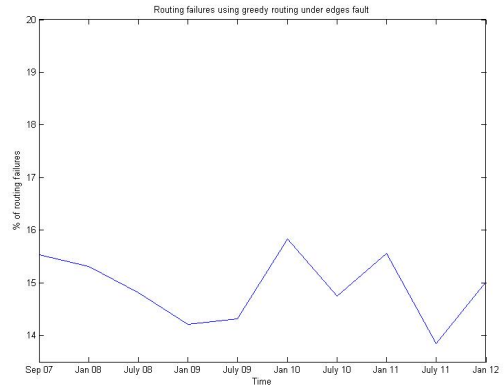
(a) Average stretch



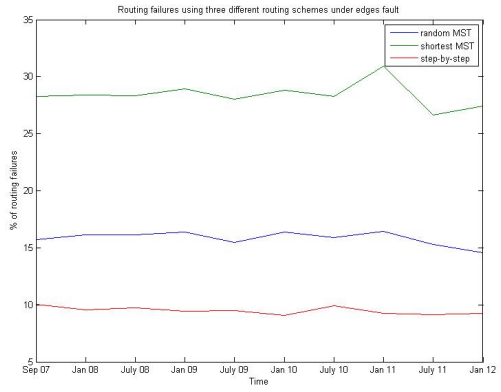
(b) Maximum congestion

Figure 6.5: Average stretch (left) and maximum congestion (right) resulted by the application of the three variations of the basic routing algorithm.

In addition, we have considered dynamic scenarios, where a part of the network fails, with the objective to assess the performance of our routing algorithms in these cases. In the following, we present results for the case when a large number of links (in particular, 10% of the links in the network) fail. We have performed additional experiments for node failures; the corresponding results are qualitatively similar and have been omitted from this report.



(a) Percentage of route failures



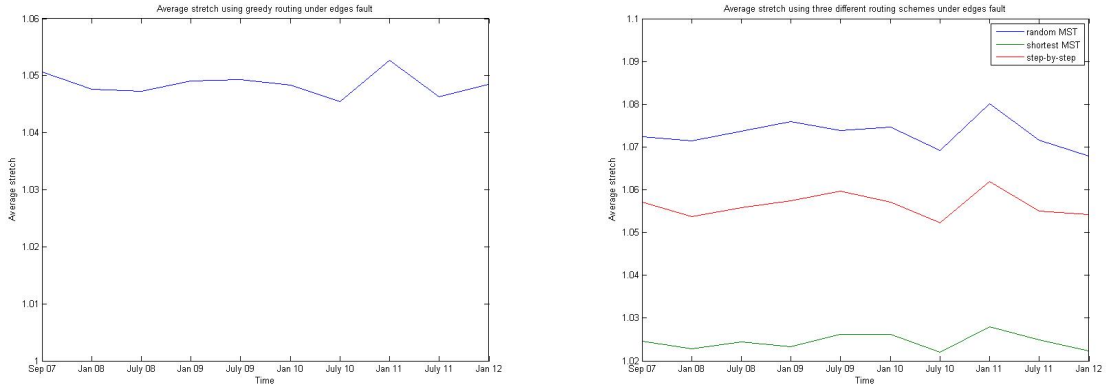
(b) Variations

Figure 6.6: Percentage of route failures (over all source-destination node pairs) for the basic routing algorithm (left) and its variations (right) when 10% of the network edges fail.

We remark that, in the case of faults, routing some source-destination pairs will be impossible. This will obviously be the case when the network loses connectivity but it can also arise due to the fact that the embedding information has been computed on a graph that is different from the one that survived after faults. Figure 6.6 (left side) represents the

percentage of source-destination node pairs which the basic routing algorithm fails to route. This is between 14% and 16% of all source-destination node pairs, a rather high value which one would expect to improve using the three variations of the basic routing algorithm. Interestingly, only step-by-step ST yields such an improvement (to approximately 10% of all source-destination node pairs).

Next, we have measured the average stretch of the successful greedy paths; for the basic routing algorithm, this turns out to be almost equal to the original one (i.e., around 1.05). Similarly to the case without faults, shortest ST improves average stretch (see Figure6.7).



(a) Average stretch for the basic routing algorithm

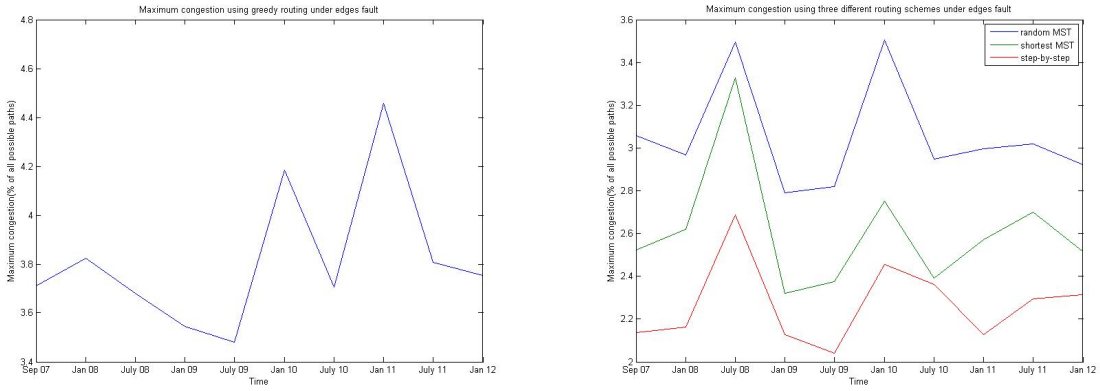
(b) Variations

Figure 6.7: Average stretch for the basic routing algorithm (left) and its variations (right) when 10% of the network edges fail.

The congestion does not deteriorate significantly due to faults (this is probably due to the uniformity in failures). For the basic routing algorithm, the corresponding values are between 3.5% and 4.5%. The three variations yield even better congestion values; step-by-step ST outperforms any other algorithm with respect to congestion in all instances.

In conclusion, the basic routing algorithm achieves almost optimal performance with respect to the (average) stretch. Its main drawback is the high congestion which is probably due to its resemblance to routing in a spanning tree. In order to mitigate this effect, we have considered three variations that rely on many different spanning trees. Concerning the dynamic scenarios considered, the main outcome is that our two metrics are rather unaffected by dynamicity (edge/node faults) albeit the unavoidable fact that there is a non-negligible probability of routing failure. The step-by-step ST algorithm performs satisfactorily with respect to this metric.

Besides the metrics considered above, the major advantage of our approach is the efficiency in computing the embedding and is an indication that embeddings into Euclidean spaces may be more preferable than embeddings into hyperbolic ones [BPK10, EG11, Kle07]. For example, as a comparison to the work of Boguna et al. [BPK10] where a greedy embedding of a $24K$-node Internet snapshot into the hyperbolic plane uses a sophisticated algorithm that runs for 24 hours, our technique required only 22 seconds.



(a) Maximum congestion

(b) Variations

Figure 6.8: Maximum congestion (as a percentage of the total successful source-destination pairs) for the basic routing algorithm (left) and its variations (right) when 10% of the network edges fail.

Chapter 7

Conclusion

In this deliverable, we have reported on activities performed in the context of Task T3.1 during the second year of EULER. Our work is a direct continuation of D3.1 where a rich set of graph properties (characteristic of the Internet topology and/or useful for routing) and models for Internet-like topologies were identified. We have proposed efficient algorithms to test several of these properties, focusing on those properties whose computation is challenging in large-scale graphs (hyperbolicity, chordality). The proposed algorithms have been implemented through our simulator and allowed us to obtain new measurements by scaling the large size of considered networks. During this period, we also have proposed a new graph model of the Internet topology based on bipartite relationship between routers and Internet switches. This model is really promising since the measurements we performed on it show that it well fits many properties of Internet-like topology. In particular, it satisfies both the power-low distribution and the high clustering coefficient of the Internet.

An important part of this deliverable is dedicated to the study of the short and long term evolution of the Internet. We have performed an extensive (and very time-consuming) campaign of simulations in order to analyze the behaviors of the properties identified in D3.1 under dynamic conditions both in the Internet topology (CAIDA maps) and in graph theoretical models. These measurements pointed out two main facts. First, Internet-like topologies (both CAIDA maps and graph theoretical models) are very robust to random link failures (even when up to 5-10% of links are removed). This highlights the fact that, in such topologies, there are many shortest paths between nodes. This important property should be taken into account in the design of routing scheme in Tasks T3.3 and T4.3 since it means that many paths can be used providing a good stretch. Moreover, topological changes should have a low impact on the long term routing schemes performances. Second, the behavior of theoretical graph models follows the one of "real" Internet topology (CAIDA maps) under the same kind of dynamic. This is a new hint for showing the relevance of these models.

For studying more "important" failures such as the removal of a set of correlated links (e.g., shared risk link groups or set of links modeling an attack), we proposed theoretical study of the worse case behaviour of some properties when a finite set of links is removed. More precisely, we have studied the impact of edge deletions on the number of modified forwarding routes entries and established theoretical bounds. We have also evaluated the impact of edge deletions on the stretch and the congestion of greedy routing. In this worse case dynamic model, we have observed that local properties are subject to large variations. In particular, few edge deletions could force to update many forwarding routes in the router nodes. Therefore,

topological changes have a strong impact on the short term routing scheme performances. More precisely, this enforce the need for routing schemes able to quickly compute new routes to restore data flows and connectivity, and then converge to optimal state. The analysis of the impact of edge deletions on the short term behavior of routing schemes requires specific simulations and experimentations as planed in Tasks T3.3 and T4.3.

All the results obtained in Task 3.1 (and documented in D3.1 and D3.3) confirms that the design of novel routing algorithms must be guided by structural properties of Internet-like graphs. In particular, it suggests that the routing schemes to be designed must intrinsically take into account the existence of numerous shortest paths to be more robust and to avoid frequent updates. On the theoretical point of view, our study of properties and models both in static and dynamic conditions lead to many open problems. For instance, it would be interesting to provide a theoretical study of the evolution of properties under edge removal such as the one we provide on the evolution of the distances. To conclude, the development of new graph models for the Internet and the study (both theoretical analysis and measurements) of structural properties will have important benefits in the design and the evaluation of routing scheme in Tasks T3.3 and T4.3. In particular, the simulator developed during this task will clearly be useful.

Bibliography

[AB00] R. Albert and A. L. Barabàsi. Topology of evolving networks: Local events and universality. *Physical Review Letters*, 85(24):5234–5237, December 2000.

[ACL00] W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. In *Proc. ACM Symposium on Theory of Computing (STOC)*, May 2000.

[ACP87] S. Arnborg, D. G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM J. Alg. Discrete Methods*, 8:277–284, 1987.

[AFR85] N. Alon, P. Frankl, and V. Rodl. Geometrical realization of set systems and probabilistic communication complexity. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science*, SFCS '85, pages 277–280, Washington, DC, USA, 1985. IEEE Computer Society.

[ALWD05] D. Alderson, L. Li, W. Willinger, and J. C. Doyle. Understanding Internet topology: Principles, models and validation. *IEEE/ACM Transactions on Networking*, 13(6):1205–1218, December 2005.

[AP90] B. Awerbuch and D. Peleg. Sparse partitions. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, SFCS '90, pages 503–513 vol.2, Washington, DC, USA, 1990. IEEE Computer Society.

[ATML12] O. Allali, L. Tabourier, C. Magnien, and M. Latapy. Internal links and pairs as a new tool for the analysis of bipartite complex networks. *Social Network Analysis and Mining*, 2012. to appear.

[BA99] A. L. Barabási and R. Albert. Emergence of scaling in random networks. In *Science*, volume 286, pages 509–512, October 1999.

[BC03] H-J. Bandelt and V. Chepoi. 1-hyperbolic graphs. *SIAM J. Discrete Math.*, 16(2):323–334, 2003.

[Ber09] A. Bernstein. Fully dynamic (2 + epsilon) approximate all-pairs shortest paths with fast query and close to linear update time. In *FOCS*, pages 693–702, 2009.

[BK96] H. L. Bodlaender and Ton Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *J. Algorithms*, 21(2):358–402, 1996.

[BK09] A. Bernstein and D. R. Karger. A nearly optimal oracle for avoiding failed vertices and edges. In *STOC*, pages 101–110, 2009.

[BKM01] G. Brinkmann, J. H. Koolen, and V. Moulton. On the hyperbolicity of chordal graphs. *Annals of Combinatorics*, 2001.

[BM93] H. L. Bodlaender and Rolf H. Möhring. The pathwidth and treewidth of cographs. *SIAM J. Discrete Math.*, 6(2):181–188, 1993.

[Bod93] H. Bodlaender. A linear time algorithm for finding tree-decompositions of small treewidth. In *STOC*, pages 226–234, 1993.

[Bod98] H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45, 1998.

[BPK10] M. Boguna, F. Papadopoulos, and D. Krioukov. Sustaining the internet with hyperbolic mapping. *Nature Communications*, 1(art. 62), 2010.

[BT97] H. L. Bodlaender and D. M. Thilikos. Treewidth for graphs with small chordality. *Disc. Ap. Maths*, 79(1-3):45–61, 1997.

[BT02] T. Bu and D. Towsley. On distinguishing between internet power law topology generators. In *21th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 2 of *Lecture Notes in Computer Science*, pages 37–48. IEEE, 2002.

[CAI] The cooperative association for internet data analysis (caida). autonomous systems maps. `http://as-rank.caida.org/data/`.

[CCG+02] Q. Chen, H. Chang, R. Govindan, S. Jamin, S. J. Shenker, and W. Willinger. The origin of power laws in internet topologies revisited. In *21th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 2, pages 608–617. IEEE, 2002.

[CDE+08] V. Chepoi, F. F. Dragan, B. Estellon, M. Habib, and Y. Vaxès. Notes on diameters, centers, and approximating trees of delta-hyperbolic geodesic spaces and graphs. *Electronic Notes in Discrete Mathematics*, 31:231–234, 2008.

[CF07] Y. Chen and J. Flum. On parameterized path and chordless path problems. In *CCC*, pages 250–263, 2007.

[CFHM12] W. Chen, W. Fang, G. Hu, and M. W. Mahoney. On the hyperbolicity of small-world and tree-like random graphs. Technical Report arXiv:1201.1717, September 2012.

[CG84] F. R. K. Chung and M. R. Garey. Diameter bounds for altered graphs. *Journal of Graph Theory*, 8(4):511–534, 1984.

[CK12] I. Caragiannis and C. Kalaitzis. Space lower bounds for low-stretch greedy embeddings. In G. Even and M. M. Halldórsson, editors, *SIROCCO*, volume 7355 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2012.

[CLPR10] S. Chechik, M. Langberg, D. Peleg, and L. Roditty. f-sensitivity distance oracles and routing schemes. In *Proceedings of the 18th annual European conference on Algorithms: Part I*, ESA'10, pages 84–96, Berlin, Heidelberg, 2010. Springer-Verlag.

[CM93] B. Courcelle and M. Mosbah. Monadic second-order evaluations on tree-decomposable graphs. *TCS*, 109:49–82, 1993.

[Col87] Charles J. Colbourn. *The Combinatorics of Network Reliability*. Oxford University Press, Inc., New York, NY, USA, 1987.

[CRA] CRAN: The Comprehensive R Archive Network project. `http://cran.r-project.org/`.

[CSTW09] W. Chen, C. Sommer, S-H. Teng, and Y. Wang. Compact routing in power-law graphs. In *23rd Int. Symp. on Distributed Computing (DISC)*, volume 5805 of *LNCS*, pages 379–391. Springer, 2009.

[Dee89] S. Deering. Host extensions for IP multicasting. RFC 1112, Internet Engineering Task Force, August 1989.

[DF07] B. Donnet and T. Friedman. Internet topology discovery: a survey. *IEEE Communications Surveys and Tutorials*, 9(4):2–15, December 2007.

[DG07] Y. Dourisboure and C. Gavoille. Tree-decompositions with bags of small diameter. *Discrete Mathematics*, 307(16):2008–2029, 2007.

[DI04] C. Demetrescu and G. F. Italiano. A new approach to dynamic all pairs shortest paths. *J. ACM*, 51(6):968–992, 2004.

[dis] distory: Distance between phylogenetic histories. `http://cran.r-project.org/web/packages/distory/index.html`.

[DKK07] M. Dynia, M. Korzeniowski, and J. Kutylowski. Competitive maintenance of minimum spanning trees in dynamic graphs. In J. van Leeuwen, G. G. Italiano, W. van der Hoek, C. Meinel, H. Sack, and F. Plasil, editors, *SOFSEM (1)*, volume 4362 of *Lecture Notes in Computer Science*, pages 260–271, Harrachov, Czech Republic, January 2007. Springer.

[dMSV11] F. de Montgolfier, M. Soto, and L. Viennot. Treewidth and hyperbolicity of the internet. In *10th IEEE International Symposium on Networking Computing and Applications (NCA)*, pages 25–32. IEEE Comp. Soc., 2011.

[DTCR08] C. Demetrescu, M. Thorup, R. Alam Chowdhury, and V. Ramachandran. Oracles for distances avoiding a failed node or link. *SIAM J. Comput.*, 37:1299–1318, January 2008.

[EG11] D. Eppstein and M.T. Goodrich. Succinct greedy geometric routing using hyperbolic geometry. *IEEE Transactions on Computers*, 60(11):1571 –1580, nov. 2011.

[ER59] P. Erdos and A. Renyi. On random graphs. *Pub. Math. Debrecen*, 6:290–297, 1959.

[ES63] P. Erdös and H. Sachs. Reguläre graphen gegebener taillenweite mit minimaler knotenzahl. *Wiss. Z. Martin-Luther-Univ. Halle-Wittenberg Math.-Natur.*, 12:251–257, 1963.

[Fen97] W. Fenner. Internet group management protocol (IGMP), version 2. RFC 2236, Internet Engineering Task Force, November 1997.

[FPW09] R. Flury, S. V. Pemmaraju, and R. Wattenhofer. Greedy routing with bounded stretch. In *INFOCOM*, pages 1737–1745. IEEE, 2009.

[GIH11] C. Glacet, D. Ilcinkas, and N. Hanusse. The impact of edge deletions on the number of errors in networks. In *15th International Conference on Principles of Distributed Systems (OPODIS)*, volume 7109 of *Lecture Notes in Computer Science*, pages 378–391. Springer, December 2011.

[GL06] J.-L. Guillaume and M. Latapy. Bipartite graphs as models of complex networks. *Physica A*, 371(2):795–813, 2006.

[Gro87] M. Gromov. Hyperbolic groups. *Essays in Group Theory*, 8:75–263, 1987.

[GS07] M. Gunes and K. Sarac. Inferring subnets in router-level topology collection studies. In *Proc. ACM/USENIX Internet Measurement Conference (IMC)*, November 2007.

[HIKN10] N. Hanusse, D. Ilcinkas, A. Kosowski, and N. Nisse. Locating a Target with an Agent Guided by Unreliable Local Advice. In *Proceedings of the 29th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing PODC 2010*, pages 355–364, Zurich Suisse, 2010. ACM New York, NY, USA.

[HIM+08] H. Haddadi, G. Iannaccone, A. Moore, R. Mortier, and M. Rio. Network topologies: Inference, modeling and generation. *IEEE Communications Surveys and Tutorials*, 10(2):48–69, April 2008.

[HKK04] N. Hanusse, E. Kranakis, and D. Krizanc. Searching with mobile agents in networks with liars. *Discrete Applied Mathematics*, 137:69–85, 2004.

[HKKK08] N. Hanusse, D. J. Kavvadias, E. Kranakis, and D. Krizanc. Memoryless search algorithms in a network with faulty advice. *Theor. Comput. Sci.*, 402(2-3):190–198, 2008.

[HS01] J. Hershberger and S. Suri. Vickrey prices and shortest paths: What is an edge worth? In *FOCS*, pages 252–259, 2001.

[HUM+08] H. Haddadi, S. Uhlig, A. Moore, R. Mortier, and M. Rio. Modeling Internet topology dynamics. *ACM SIGCOMM Computer Communication Review*, 38(2):65–68, March 2008.

[IMF04] A. Iamnitchi, R. Matei, and I. Foster. Small world file-sharing communities. In *Proc. IEEE INFOCOM*, April 2004.

[Ita08a] G. F. Italiano. Fully dynamic all pairs shortest paths. In Kao [Kao08].

[Ita08b] G. F. Italiano. Fully dynamic minimum spanning trees. In Kao [Kao08].

[Kal12] C. Kalaitzis. Efficient addressing and routing in large-scale communication networks. Msc thesis, University of Patras, 2012.

[Kao08] M-Y. Kao, editor. *Encyclopedia of Algorithms*. Springer, 2008.

[KB10] N. Khanna and S. Baswana. Approximate shortest paths avoiding a failed vertex: Optimal size data structures for unweighted graphs. In *STACS*, pages 513–524, 2010.

[Kin99] V. King. Fully dynamic algorithms for maintaining all-pairs shortest paths and transitive closure in digraphs. In *FOCS*, pages 81–91, 1999.

[KK95] T. Kloks and D. Kratsch. Treewidth of chordal bipartite graphs. *J. Algorithms*, 19(2):266–281, 1995.

[KK99] E. Kranakis and D. Krizanc. Searching with uncertainty. In *Proc. SIROCCO'99,*, pages 194–203, 1999.

[KK09] Y. Kobayashi and K. Kawarabayashi. Algorithms for finding an induced cycle in planar graphs and bounded genus graphs. In *20th Annual ACM-SIAM Symp. on Discrete Alg. (SODA)*, pages 1146–1155. SIAM, 2009.

[Kle07] R. Kleinberg. Geographic routing using hyperbolic space. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 1902 –1909, may 2007.

[KLNS12a] A. Kosowski, B. Li, N. Nisse, and K. Suchan. k-chordal graphs: from cops and robber to compact routing via treewidth. In *14es Rencontres Francophones sur les Aspects Algorithmiques de Télécommunications (AlgoTel)*, pages 83–86, 2012.

[KLNS12b] A. Kosowski, B. Li, N. Nisse, and K. Suchan. k-chordal graphs: from cops and robber to compact routing via treewidth. In *39th International Colloquium on Automata, Languages and Programming (ICALP, track C)*, volume 7392 of *Lecture notes in computer science*, pages 610–622. Springer, 2012.

[Knu05] D. Knuth. *The art of computer programming*, volume 4, Fascicle 2, chapter Generating All tuples and permutations. Addison-Wesley, February 2005.

[KPBV09] D. V. Krioukov, F. Papadopoulos, M. Boguñá, and A. Vahdat. Greedy forwarding in scale-free networks embedded in hyperbolic metric spaces. *SIGMETRICS Performance Evaluation Review*, 37(2):15–17, 2009.

[LLR94] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 577 –591, nov 1994.

[LM10] T. Leighton and A. Moitra. Some results on greedy embeddings in metric spaces. *Discrete Comput. Geom.*, 44(3):686–705, October 2010.

[LMDV08] M. Latapy, C. Magnien, and N. Del Vecchio. Basic notions for the analysis of large two-mode networks. *Social Networks*, 30(1):31–48, January 2008.

[Lok10] D. Lokshtanov. On the complexity of computing treelength. *Discrete Applied Mathematics*, 158(7):820–827, 2010.

[LPCN10] M. Latapy, T.H.D. Phan, C. Crespelle, and T.Q. Nguyen. Termination of multipartite graph series arising from complex network modeling. In *Proc. International Conference on Combinatorial Optimization and Applications (COCOA)*, December 2010.

[May06] P. Maymounkov. Greedy embeddings, trees, and euclidean vs. lobachevsky geometry. Manuscript, 2006.

[MDBP10] P. Mérindol, B. Donnet, O. Bonaventure, and J.-J. Pansiot. On the impact of layer-2 on node degree distribution. In *Proc. ACM/USENIX Internet Measurement Conference (IMC)*, November 2010.

[MOVL09] C. Magnien, F. Ouedraogo, G. Valadon, and M. Latapy. Fast dynamics in Internet topology: Observations and first explanations. In *Proc. 4th International Conference on Internet Monitoring and Protection (ICIMP)*, May 2009.

[MVdSD+09] P. Mérindol, V. Van den Schriek, B. Donnet, O. Bonaventure, and J.-J. Pansiot. Quantifying ASes multiconnectivity using multicast information. In *Proc. ACM/USENIX Internet Measurement Conference (IMC)*, November 2009.

[New01] M. Newman. Scientific collaboration networks. network construction and fundamental results. *Physical Review E*, 64(1), June 2001.

[New09] M. E. J. Newman. Random graphs with clustering. *Physical Review Letters*, 103(5), July 2009.

[NPW03] E. Nardelli, G. Proietti, and P. Widmayer. Finding the most vital node of a shortest path. *Theor. Comput. Sci.*, 296:167–177, March 2003.

[NST12] O. Narayan, I. Saniee, and G. H. Tucci. Lack of spectral gap and hyperbolicity in asymptotic Erdös-Renyi sparse random graphs. In *5th International Symposium on Communications, Control and Signal Processing (ISCCSP)*, pages 1–4, Rome, Italy, May 2012. IEEE.

[NWS01] M. E. J. Newman, D. J. Watts, and S. H. Strogatz. Random graphs with arbitrary degree distribution and their applications. *Phys. Rev. E.*, 64, July 2001.

[OP09] T. Opsahl and P. Panzarasa. Clustering in weighted networks. *Social Networks*, 31(2):155–163, 2009.

[Pan] J.-J. Pansiot. `mrinfo` dataset. see `http://svnet.u-strasbg.fr/mrinfo/`.

[PR05] C. H. Papadimitriou and D. Ratajczak. On a conjecture related to geometric routing. *Theor. Comput. Sci.*, 344(1):3–14, November 2005.

[PSV04] R. Pastor-Satorras and A. Vespignani. *Evolution and Structure of the Internet: a Statistical Physics Approach.* Cambridge University Press, February 2004.

[PU89] D. Peleg and E. Upfal. A trade-off between space and efficiency for routing tables. *Journal of the ACM*, 36(3):510–530, 1989.

[Pus03] T. Pusateri. Distance vector multicast routing protocol version 3 (DVMRP). Internet Draft (Work in Progress) draft-ietf-idmr-dvmrp-v3-11, Internet Engineering Task Force, October 2003.

[RS84] N. Robertson and P. D. Seymour. Graph minors. iii. planar tree-width. *J. Comb. Theory, Ser. B*, 36(1):49–64, 1984.

[SBvL87] A. Schoone, H. Bodlaender, and J. van Leeuwen. Improved diameter bounds for altered graphs. In Gottfried Tinhofer and Gunther Schmidt, editors, *Graph-Theoretic Concepts in Computer Science*, volume 246 of *Lecture Notes in Computer Science*, pages 227–236. Springer Berlin / Heidelberg, 1987.

[Sha11] Y. Shang. Lack of gromov-hyperbolicity in colored random networks. *PanAmerican Mathematical Journal*, 21(1):27–36, 2011.

[SSR94] R. Sundaram, K. S. Singh, and C. P. Rangan. Treewidth of circular-arc graphs. *SIAM J. Discrete Math.*, 7(4):647–655, 1994.

[SW05a] T. Schank and D. Wagner. Approximating clustering coefficient and transitivity. *Journal of Graph Algorithms and Applications (JGAA)*, 9(2):265–275, 2005.

[SW05b] T. Schank and D. Wagner. Finding, counting and listing all triangles in large graphs, an experimental study. In *Proc. International Workshop on Experimental and Efficient Algorithms (WEA)*, May 2005.

[Tho04] M. Thorup. Fully-dynamic all-pairs shortest paths: Faster and allowing negative cycles. In *SWAT*, pages 384–396, 2004.

[Ueh99] R. Uehara. Tractable and intractable problems on generalized chordal graphs. Technical Report COMP98-83, IEICE, 1999.

[WAD09] W. Willinger, D. Alderson, and J. C. Doyle. Mathematics and the Internet: a source of enormous confusion and great potential. *Notices of the American Mathematical Society*, 56(5), May 2009.

[War68] H. E. Warren. Lower bounds for approximation by nonlinear manifolds. *Transactions of the American Mathematical Society*, 133(1):pp. 167–178, 1968.

[Wax88] B. M. Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, December 1988.

[WL10] X. Wang and D. Loguinov. Understanding and modeling the Internet topology: Economics and evolution perspective. *IEEE/ACM Transactions on Networking*, 18(1):257–270, February 2010.

[WS98] D. J. Watts and S. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[WZ11] Y. Wu and C. Zhang. Hyperbolicity and chordality of a graph. *Electr. J. Comb.*, 18(1), 2011.

[XL07] X. Xu and F. Liu. A novel configuration model for random graphs with given degree sequence. *Chinese Physics*, 16(2), February 2007.

[ZCB96] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee. How to model an internetwork. In *Proc. IEEE INFOCOM*, March 1996.

[ZCD97] E. Zegura, K. Calvert, and M. Donahoo. A quantitative comparison of graph-based models for internetworks. In *IEEE/ACM Transactions on Networking*, volume 5, pages 770–783, December 1997.