A Radar for the Internet

Matthieu Latapy [1], **Clémence Magnien** [1] **and Frédéric Ouédraogo** [1,2]

Abstract. *Mapping the internet's topology is a challenge in itself, and studying its dynamics is even more difficult. Achieving this would however provide key information on the nature of the internet, crucial for modeling and simulation. Moreover, detecting anomalies in this dynamics is a key issue for security. We introduce here a new measurement approach which makes it possible to capture internet dynamics at a scale of a few minutes in a radar-like manner. By conducting and analyzing large-scale measurements of this kind, we rigorously and automatically detect events in the observed dynamics, which is totally out of reach of previous approaches.*

Since the end of the 90s, mapping the internet as a large set of nodes and links received much attention. However, due to its distributed nature and its sheer size, accurately measuring this topology is extremely difficult. The main method to do so relies on the classical traceroute tool [8], which gives a path from a machine connected to the internet (called monitor) to any other (called destination). Such paths are composed of IP addresses of internet routers and links between them. One may then obtain a (partial) map of the internet by running traceroute from many monitors to many destinations, and merging the obtained paths, see Figure 1. For various reasons, however, this is far from trivial and the obtained maps are not satisfactory [6, 3, 4]. Therefore, much effort is nowadays devoted to the improvement of available maps, in particular by increasing the number of monitors, e.g. [43, 24], and by designing more accurate measurement tools, e.g. [12].

In this situation, it must be clear that studying the *dynamics* of the internet's topology, and in particular detecting *events* in this dynamics, is totally out of reach of current approaches. Even the study of global trends in the evolution of the internet is extremely difficult [31].

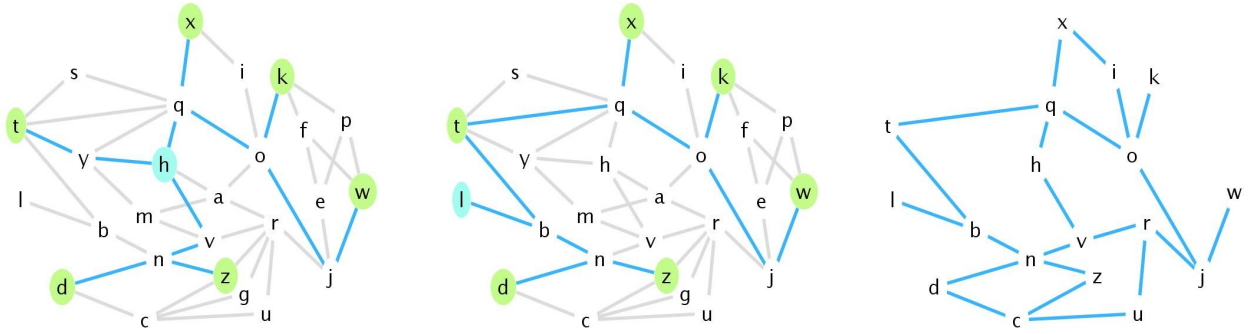
We propose here an approach which makes it possible, for the first time, to observe the dynamics of the internet's topology at the scale of a few minutes. It consists in focusing on the part of the internet's topology viewed from a single monitor, which we call an *ego-centered view*, see Figure 1. Such views are far from representative maps of the internet, but they have several key advantages. In particular, they can be obtained very rapidly with low network load, and thus may be repeated at a high frequency. This makes it possible to study their dynamics, and then to gain insight on the dynamics of the internet's topology itself.

The measurement tool.

In principle, one may use the traceroute tool to obtain ego-centered views: it suffices to run it from a monitor towards a given set of destinations, and then merge the obtained paths, as in Figure 1. However, this approach has severe drawbacks. In particular, it is highly redundant and it induces a very heterogeneous load on links: since traceroute sends one probe for each link on the path to discover, the links close to the monitor are overloaded. For instance, the traceroute ego-centered measurement from l described in Figure 1 discovers link l–b six times, using six probes. The situation is even worse in practice, see supplementary material [10].

[1]LIP6 – CNRS and UPMC (University Pierre and Marie Curie – Paris 6), 104 avenue du Président Kennedy, 75016 Paris, France. Firstname.Lastname@lip6.fr

[2]University of Ouagadougou, 03 B.P. 7021 Ouagadougou 03, Burkina-Faso.



ego-centered view from h ego-centered view from l obtained map

Figure 1: Measurement of an internet-like topology. The classical approach consists in running traceroute from a number of monitors, here h and l, towards some destinations, here t, x, k, w, z and d. We obtain here the following paths from h: h–y–t; h–q–x; h–q–o–k; h–q–o–j–w; h–v–n–z; h–v–n–d. We obtain the following paths from l: l–b–t; l–b–t–q–x; l–b–t–q–o–k; l–b–t–q–o–j–w; l–b–n–z; l–b–n–d. The final map of the network is obtained by merging all these paths. Instead, one may focus on the set of paths obtained from a *single* monitor, which we call its *ego-centered* view of the network.

In order to perform fast ego-centered measurements with low and balanced network load, we therefore had to design a dedicated measurement tool, called *tracetree*. The traceroute tool discovers a path by sending a series of probes towards the destination in a forward manner: the first probe discovers the first link, the second probe discovers the second link, and so on. Instead, the tracetree tool discovers a tree by sending probes towards all destinations in parallel in a backward manner and avoids redundancy by stopping probing towards some destinations when paths collapse: given a set of destinations, it first discovers the last link on the path to each of them, then the previous link on each of these paths, and so on; when two (or more) paths reach the same node then it stops probing towards all corresponding destinations except one. See Figure 2 for an illustration, and supplementary material for a detailed specification and implementation of tracetree [10].

The tracetree tool performs ego-centered measurements very efficiently, both regarding time and network load. It sends exactly *one* probe for each link to discover, and thus induces a perfectly balanced load. Radar measurements then consist in iterating ego-centered measurements with tracetree, from a monitor to a given set of destinations.

Measurements and dataset.

In order to conduct radar measurements relevant for our goals, one has to decide on several parameters. In particular, there is a trade-off between the frequency at which one conducts these measurements (it should be as high as possible), their size (they should capture the dynamics of as many nodes as possible), and the induced network load (it must be low enough to avoid problems with network administrators and bias it may induce).

Many parameters have an impact on these desirable features. As it is impossible to test all combinations to choose the best ones, we used the following approach. We first chose a set of seemingly reasonable parameters, which we call *base parameters*. Then we started measurements with these parameters from several monitors in parallel. We kept some monitors, called *control monitors*, with these parameters constant; on others, called *test monitors*, we alternated periods

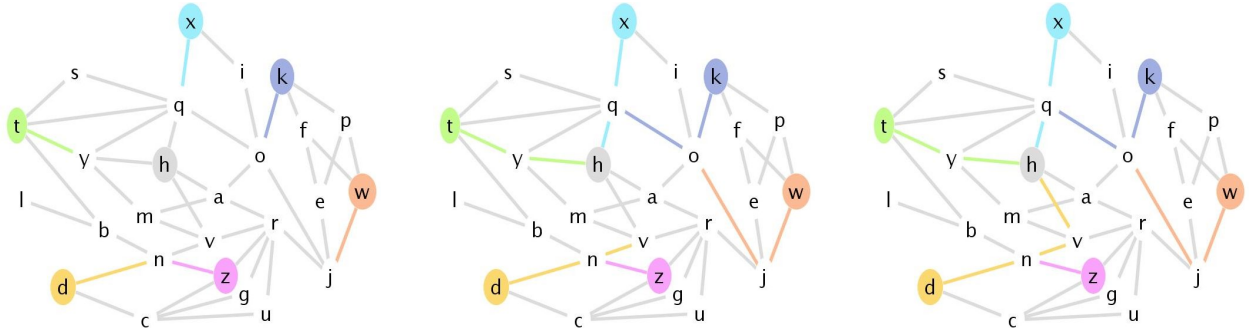


Figure 2: Illustration of the tracetree measurement method. Left: the first series of probes discovers the last link before each destination; tracetree stops probing towards z because the paths to d and z collapse (at n). Center: the second series of probes discovers the links just before, and tracetree stops probing towards most destinations: towards t and x because the full path towards them has been discovered; towards k and w because the corresponding paths collapse with previously discovered nodes. Right: the third (and last) series of probes contains only one probe, sent towards d, which ends the tracetree measurement. Finally, tracetree sent exactly one probe for each link, thus avoiding redundancy and reducing the network load significantly.

with base parameters and periods where we changed one parameter. The observed changes made it possible to study the influence of this parameter. Control monitors made it possible to check that the changes observed from test monitors was caused by changes of parameters, not events in the network. The alternation of periods with base parameters and modified ones also made it possible to confirm this.

We provide a detailed study of parameters and their influence as supplementary material [10]. These experiments made it possible to identify several sets of parameters which reach the trade-off we pointed out above.

We finally conducted independent measurements from more than one hundred monitors scattered around the world, towards sets of 3 000 destinations chosen at random among valid IP addresses. We sent probes at a maximal distance of 30 hops, and waited for answers until a timeout of 2 seconds. With these parameters, each ego-centered measurement lasted around 4 minutes. As we wait 10 minutes between two rounds (to reduce the network load), we obtain one such measurement every 14 minutes approximately, or close to 100 per day. We ran these measurements continuously during several months, thus obtaining a very rich dataset which we provide as supplementary material [10].

Events in the dynamics.

The most natural idea to detect events in the dynamics captured by a radar measurement from a given monitor certainly is to study the number N_i of nodes observed at each round i. We plot it for a typical case in Figure 3 (middle row, black plot). Clear events appear under the form of sharp decreases of N_i for some values of i, but this brings little information: most such decreases are due to local failures of the network through which the monitor accesses the internet, which suddenly make its ego-centered view almost blank. Instead, we notice that this plot exhibits no sharp increase, which is confirmed by the distribution of the N_i for all i, plotted in Figure 3 (top row, black plot). This is a nontrivial fact, as one may very well imagine scenarios where such increases

would appear. In practice, however, the value of N_i is very stable, except for sharp decreases which bring little information, if any.

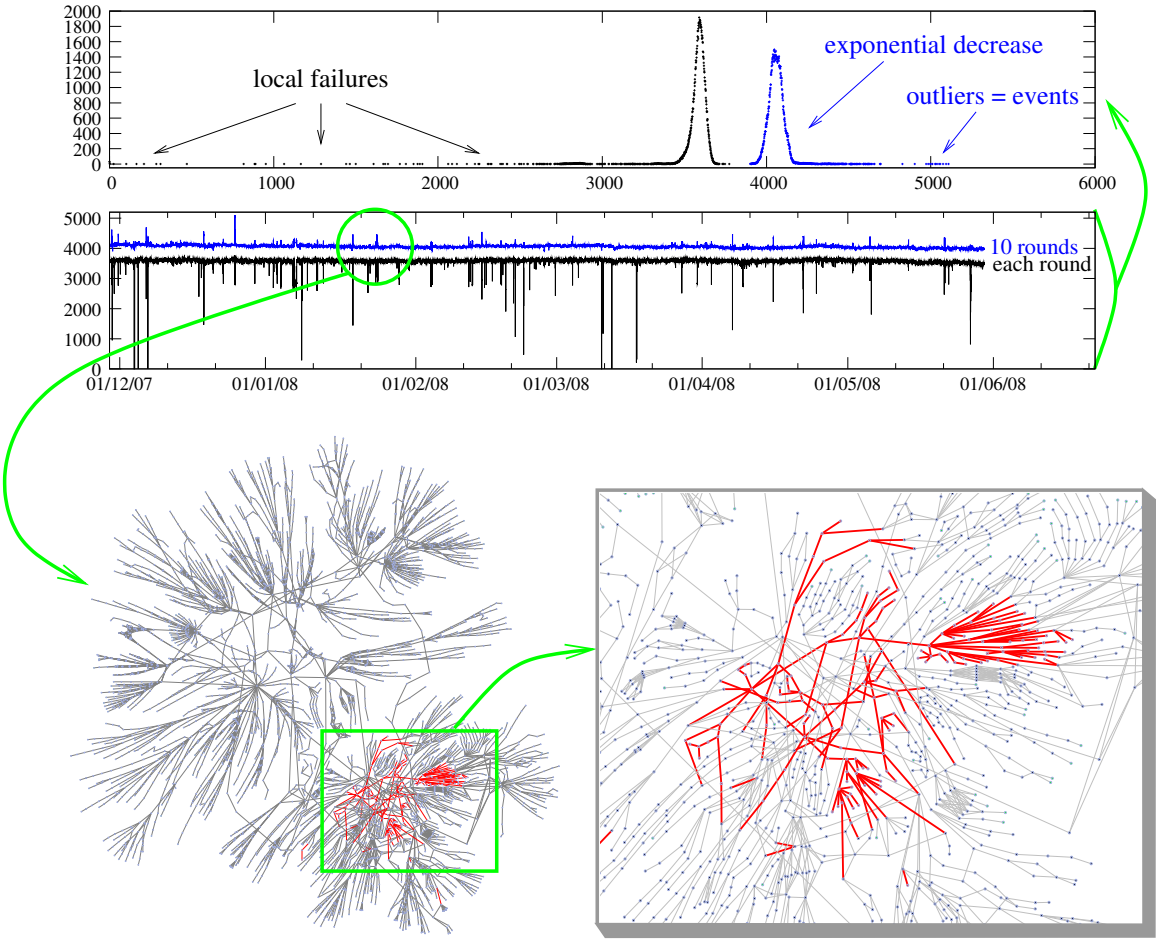


Figure 3: Middle row: plot of the number of distinct nodes N_i (resp. $N_{(i-9)..i}$) observed during each round (resp. ten consecutive rounds) of measurement, as a function of time. Top row: the distributions of these values, which confirms that N_i exhibits abnormally small values only, never abnormally large ones, unlike $N_{(i-9)..i}$. Bottom row: topology changes observed during an event identified by an abnormally large value of $N_{(i-9)..i}$, with a zoom on the part of the network where the event occurred.

The fact that the number N_i of nodes observed at each round i is very stable does not mean that there is no dynamics: consecutive rounds may consist of very different sets of nodes with the same size. Suppose for instance that we conduct a radar measurement of the network in Figure 1 from monitor h. Let us consider the i-th round of measurement, for a given index i at which the ego-centered measurements from h is the one depicted in Figure 1 (left). Then suppose that at round $i+1$ the path from h to w changes to h–q–o–k–f–w; and that at round $i+2$ the path from h to w changes to h–q–o–k–p–w and the one from h to t changes to h–q–s–t. In this situation, we have $N_i = N_{i+1} = N_{i+2} = 13$, despite the fact that the ego-centered views changed significantly.

Such changes may be observed in the number of distinct nodes seen in series of consecutive rounds. Suppose for instance that, in the scenario above, the ego-centered view from h did not change from round j to round i, for a $j < i$. Then the number of nodes observed during three consecutive rounds experiences an increase of more than 23%: if we denote by $N_{x..y}$ the number of distinct nodes observed from rounds x to y, then we have $N_{j..(j+2)} = N_{(j+1)..(j+3)} = \ldots = N_{(i-2)..i} = 13$, but $N_{(i-1)..(i+1)} = 14$ and $N_{i..(i+2)} = 16 > N_{(i-2)..i} + 23\%$.

Similarly, we display in Figure 3 (middle row, blue plot) the number of nodes observed in ten consecutive rounds in a typical practical case, which exhibits significant increases, thus revealing events in the dynamics (it also experiences significant decreases, which we removed to improve readability, as they indicate local network failures only).

This plot has another key feature: the observed values are well centered around a typical value but also reach some extremal *outlier* values, see the distribution plotted in Figure 3 (top row, blue distribution). This means that the sharp increases indeed reveal *events* in a rigorous statistical sense.

Finally, we are able to point out precise times where events occur in the dynamics of the observed topology. This opens the way to further investigation of the shape and nature of these events, for instance by drawing the topology and the changes it experienced at these precise times. We display a typical case in Figure 3 (bottom row): it shows that, whereas the dynamics is in general scattered in all the network, the events we detect correspond to a significant change in a specific part of the topology. This confirms that these events make sense from a networking point of view. They correspond to major changes in specific parts of the internet, which we are able to automatically detect at a time scale of a few minutes, much more precisely than all previous work in this area.

Acknowledgements

This work is partly funded by the European Commission through the EULER project (grant 258307), part of the *Future Internet Research and Experimentation (FIRE)* objective of the Seventh Framework Programme (FP7). It is also partly funded by the *DynGraph* grant from the *Agence Nationale de la Recherche* with reference ANR-10-JCJC-0202.

*References

[1] Supplementary material, including programs and data. `http://www-rp.lip6.fr/~latapy/Radar/`.

[2] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with paris traceroute. In *Proc. ACM/USENIX Internet Measurement Conference (IMC)*, Oct. 2006.

[3] Luca Dall'Asta, J. Ignacio Alvarez-Hamelin, Alain Barrat, Alexei Vázquez, and Alessandro Vespignani. Exploring networks with traceroute-like probes: Theory and simulations. *Theor. Comput. Sci.*, 355(1):6–24, 2006.

[4] J. L. Guillaume and M. Latapy. Relevance of massively distributed explorations of the internet topology: Simulation results. In *Proc. IEEE INFOCOM*, Mar. 2005.

[5] B. Huffaker, D. Plummer, D Moore, and k claffy. Topology discovery by active probing. In *Proc. Symposium on Applications and the Internet*, Jan. 2002.

[6] A. Lakhina, J. Byers, M. Crovella, and P. Xie. Sampling biases in IP topology measurements. In *Proc. IEEE INFOCOM*, Apr. 2003.

[7] Ricardo Oliveira, Beichuan Zhang, and Lixia Zhang. Observing the evolution of internet AS topology. In *Proc. ACM SIGCOMM*, 2007.

[8] J. Postel. Internet control message protocol. RFC 791, September 1981.

[9] Y. Shavitt and E. Shir. DIMES: Let the internet measure itself. *ACM SIGCOMM Computer Communication Review*, 35(5):71 – 74, October 2005.

– Supplementary material –

Related work.

Mapping the internet is a very active area [43, 24, 11, 26, 20, 29, 14, 47, 22, 45, 34, 16], to which our work is naturally related. In particular, there is a constant effort in designing more accurate and powerful measurement tools [13, 12, 19, 18, 17, 30, 46, 49, 44, 22, 45, 43]. Some of them aim (at least in part) at reducing the load induced on the network [19, 18, 17, 44, 22, 45, 43], and in this context the idea of backward probing in order to avoid redundancy has already been studied in depth [17]. Donnet et al, in particular, proposed the DoubleTree algorithm as a way to reduce significantly this redundancy, and provide a prototype [19, 18] directed towards massively distributed measurements [11].

The idea of routing tree collection also naturally occurs in multicast studies, and several tools have been proposed in this context [33, 41, 42, 28]. Pansiot even conducted daily multicast measurements for approximately two years and studied the observed dynamics [33]. All these authors generally insist on the fact that a tool like tracetree would be of high interest.

Studying the dynamics of the internet's topology as a whole also received some attention, both at AS (Autonomous System), router, and interface levels [37, 38, 31, 15, 36, 35]. Because of the sheer size of the internet and the difficulties encountered in measuring it, these studies consider relatively coarse grained measurements (several orders of magnitude coarser than ours) and focus on basic properties like size, degrees, or diameter. Their aim is to describe long-term evolution of the internet (like its growth for instance).

Other contributions deal with routing anomalies and dynamics at BGP or traceroute levels, see for instance [13, 12, 21, 48, 39, 25]. They study wide varieties of phenomena, like load balancing, routing stability, node reachability, routing convergence, etc.

Unbalanced load induced by traceroute ego-centered measurements.

One may use traceroute directly to collect ego-centered views by probing a set of destinations. This approach however has serious drawbacks. First, as detailed in [18], the measurement load is highly unbalanced between nodes and there is much redundancy in the obtained data (intuitively, one probes links close to the monitor much more than others). Even worse, this implies that the obtained information is not homogeneous, and thus much more difficult to analyze rigorously.

Figure 4 illustrates this imbalance. The distribution of the link load (left) shows that some links are probed a very high number of times with traceroute, up to 3 000 times. This load is placed on links close to the monitor (center, black plot): a very large number of probes are sent a small distance from the monitor. Comparatively, our tracetree tool imposes a much lighter load: it sends a much smaller number of probes, in particular to links close to the monitor (Figure 4, center, blue plot). Finally, repeated ego-centered measurements with tracetree discover more IP addresses (and therefore more interesting information) with fewer probes sent than comparable traceroute measurements (Figure 4, right).

The tracetree tool. [2]

As already discussed in various contexts [19, 18, 17, 33, 30, 45], one may avoid the issues described above by performing tree-like measurements in a backward way: given a set of destinations to probe, one first discovers the last link on the path to each of them, then the previous link on

[2] An open-source implementation is available at [10].

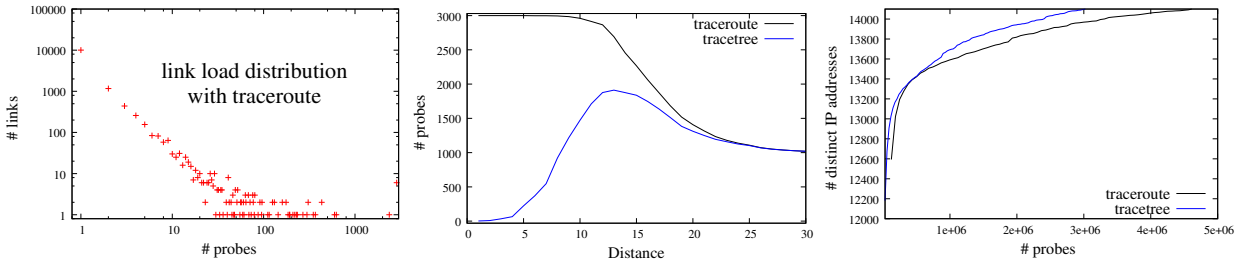


Figure 4: Left: typical link load distribution with a traceroute ego-centered measurement. For each value x on the horizontal axis, we give the number of links which are discovered x times during a traceroute ego-centered measurement with 3 000 destinations (base value). In an equivalent tracetree measurement, exactly one probe is sent for each link. Center: number of probes sent, as a function of the distance from the monitor, with traceroute and tracetree ego-centered measurements with 3 000 destinations. Right: number of distinct IP addresses discovered with several successive traceroute and tracetree ego-centered measurement rounds, as a function of the number of probes sent.

each of these paths, and so on; when two (or more) paths reach the same node then the probing towards all corresponding destinations, except one, stops.

Note that this requires knowing the distance towards each destination, which is not trivial [30]. This plays a key role here, since over-estimated distances lead to several packets hitting destinations. Under-estimated distances, instead, miss the last links towards the destinations. Each tracetree measurement therefore relies on an *estimation* of the distance towards each destination (we detail in the next section how we obtain this estimation). If the distance is over-estimated, then more than one packet reaches the corresponding destination; we only output the information corresponding to the last packet to which the destination replies, which corresponds to the accurate distance. If the distance happens to be under-estimated (we do not see the destination at this distance), then we set it to a default maximal value (generally equal to 30) and start the measurement from there.

However, as illustrated in Figure 5, such naive measurements encounter serious problems because of routing changes and other events. We provide a solution in the tracetree algorithm below: the tree nodes are not IP addresses anymore, but pairs composed of an IP address (or a star if a timeout occurred) and the TTL at which it was observed (see Figure 5 for an illustration). This is sufficient to ensure that the obtained view is a tree, while keeping the algorithm very simple. It sends only one packet for each link, and thus is optimal. Moreover, each link is discovered exactly once, which gives an homogeneous view of the topology and balances the measurement load.

Algorithm 1: tracetree algorithm.

```
   Input: set D of destinations, with d ∈ D at distance ttl_d.
   to_probe ← empty queue, to_receive ← ∅, seen ← ∅
   foreach d ∈ D do add (d, ttl_d) to to_probe
   while to_probe not empty or to_receive ≠ ∅ do
α      if to_probe not empty then
           pop (d, ttl) from to_probe and send a probe to it
           add (d, ttl, current_time()) to to_receive
       // here necessarily to_receive ≠ ∅
β      if answer p to a probe to (d, ttl) received then
           // p sent by p.source, reply to a probe to (d, ttl)
           if (d, ttl, _) ∈ to_receive then
               // else timeout
               remove (d, ttl, _) from to_receive;
               print p.source ttl d
               if (p.source, ttl) ∉ seen then
                   add (p.source, ttl) to seen
                   push (d, ttl − 1) in to_probe if ttl > 1
       for (d, ttl, t) ∈ to_receive if timeout exceeded do
           remove (d, ttl, t) from to_receive
           print * ttl d
           if ttl > 1 then
               push (d, ttl − 1) in to_probe
```

From such trees with (IP,TTL) nodes, one obtains a tree on IP addresses by applying the following filter (illustrated in Figure 5) [3]: first merge all nodes of the tree which correspond to the same IP; remove loops (links from an IP to itself); iteratively remove the stars with no successor; merge all the stars which are successor of a same node into a unique star; construct a BFS tree of the obtained graph which leads to a tree on IP addresses; iteratively remove the leaves which are not the last nodes encountered when probing any destination.

The key point is that the obtained tree is a possible IP routing tree from the monitor to the destinations (similar to a broadcast tree). The obtained tree contains almost as much information as the original tracetree output and has the advantage of being much more simple to analyze. We evaluated the impact of this filtering on our observations, and found that it was negligible.

Many non-trivial points would deserve more discussion. For instance, one may apply a greedy sending or receiving strategy (by replacing line α or β in Algorithm 1 by a `while`, respectively); identifying reply packets is non-trivial, as well as extracting the relevant information from the read packets; introducing a delay may be necessary to stay below the maximal ICMP sending rate of the monitor; one may consider answers received after the timeout but before the end of the measurement (whereas we ignore them); one may use other protocols than ICMP (the classical traceroute uses UDP or ICMP packets); the initial order of the destinations may have an impact on

[3] The measurement would be slightly more efficient if the filter was included directly in tracetree; however, to keep things simple and modular, we preferred to separate the two.

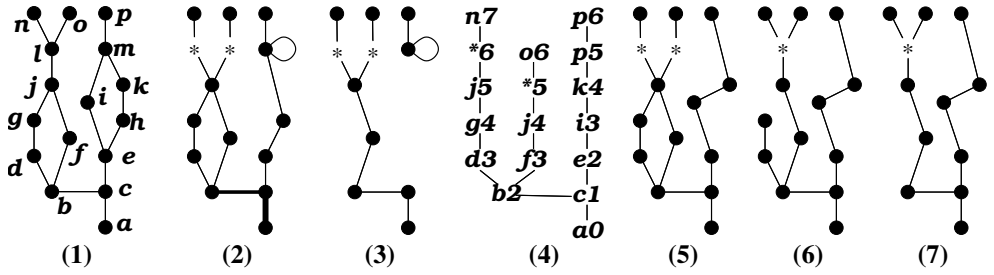


Figure 5: Typical outputs of various measurements schemes. (1) – Real topology. a is the monitor, n, o, and p are the destinations. We suppose that l does not answer to probes, that b is a per-destination load balancer, forwarding traffic for n to d, and traffic for o to f, and that e is a per-packet load balancer forwarding packets alternately to i and h. Such situations are frequent in practice. (2) – Measurement with traceroute. Three routes are collected, leading to a higher load on links close to the monitor (represented by thicker lines here). (3) – Naive tree measurement. Because of a route change due to per-packet load balancer e, one obtains a disconnected part. (4) – Measurement with tracetree. Nodes are pairs of IP addresses and TTL, with redundancy in the addresses; one necessarily obtains a tree. (5–7) – Main steps of the filtering process. (5) – Pairs with same IP address are merged and loops are removed; (6) – Appropriate stars are merged and a BFS tree is computed; (7) – Leaves which are not the last node on a path towards a destination are iteratively removed. This is the final output of the filter.

the measurement; there may be many choices for the BFS tree in the filter; etc. However, entering in such details is far beyond the scope of this paper, and we refer to the code and its documentation [10] for full details.

Radar measurements.

With the tracetree tool and its filtered version, we have the ground material to conduct radar measurements: given a monitor and a set of destinations, it suffices to run periodic ego-centered measurements. The measurement frequency must be high enough to capture interesting dynamics, but low enough to keep the network load reasonable. We will discuss this in the next section.

We also use each tracetree measurement to estimate the distances towards all destinations for the next round of measurement: one may indeed suppose that the distance between the monitor and any destination generally is stable between consecutive rounds. Then, the distances at the start of a given round are the ones observed during the previous round (or the maximal value if the destination was not seen).

Influence of parameters.

First notice that many parameters (including the monitor and destination set) may have a deep impact on the obtained data. Estimating this impact is a challenging task since testing all combinations of parameters is totally out of reach. In addition, the continuous evolution of the measured object makes it difficult to compare several measurements: the observed changes may be due to parameter modifications or to actual changes in the topology.

To bypass these issues while keeping the study rigorous, we propose the following approach. We first choose a set of seemingly reasonable parameters, which we call *base parameters*. Those parameters are described in Table 1. Then we conduct measurements with these parameters from several monitors in parallel. On some monitors, called *control monitors*, we keep these parameters

Parameter	Base value
# destinations	3 000
choice of destinations	random, answering to ping
choice of monitor	PlanetLab
maximal TTL	30
timeout	2s
delay between rounds	10 minutes

Table 1: Parameters of our radar measurements, and base values of these parameters.

constant; on others, called *test monitors*, we alternate periods with base parameters and periods where we change (generally one of) these parameters. Control monitors make it possible to check that the changes observed from test monitors are due to changes of parameters, not to events on the network. The alternation of periods with base parameters and modified ones also makes it possible to confirm this, and to observe the induced changes in the observations. In many cases, it is also possible to simulate what one would have seen in principle if the parameters had stayed unchanged, which gives further insight (we will illustrate this below).

We focus on a few representative parameters only, the key conclusion being that the base parameters fit our needs very well.

Figure 6 (left) shows the impact of the inter-round delay: on the rightmost part the delay was significantly reduced, leading to an increase in the observation's time resolution (*i.e.* more points per unit of time). It is clear from the figure that this has no significant impact on the observed behavior. In particular, the variations in the number of IP addresses seen, though they have a higher resolution after the speed-up, are very similar before and after it. Moreover, the control monitor shows that the base time scale is relevant, since improving it does not reveal significantly higher dynamics.

Figure 6 (middle) shows the impact of the number of destinations. As expected, increasing this number leads to an increase in the number of observed IP addresses. The key point however is that increasing the number of destinations may lead to a relative loss of efficiency: simulations of what we would have seen with 3 000 or 1 000 destinations display a smaller number of IP addresses than direct measurements with these numbers of destinations (the control monitor proves that this is not due to a simultaneous topology change). This is due to the fact that probing towards 10 000 destinations induces too high a network load: since some routers answer to ICMP packets with a limited rate only [23], overloading them makes them invisible to our measurements. Importantly, this does not occur in simulations of 1 000 destination measurements from ones with 3 000, thus showing that the load induced with 3 000 destinations is reasonable, to this regard.

Figure 6 (right) shows the impact of the timeout value. As expected, decreasing the timeout leads to a decrease in the round duration. However, it also causes more replies to probe packets to be ignored because we receive them after the timeout. A good value for the timeout is a compromise between the two. We observe that the round duration is only slightly larger with a timeout of $2s$ than with a timeout of $1s$ (contrary to the change between a timeout of 4 and $2s$). The base value of the timeout (2s) seems therefore appropriate, because it is rather large and does not lead to a long round duration.

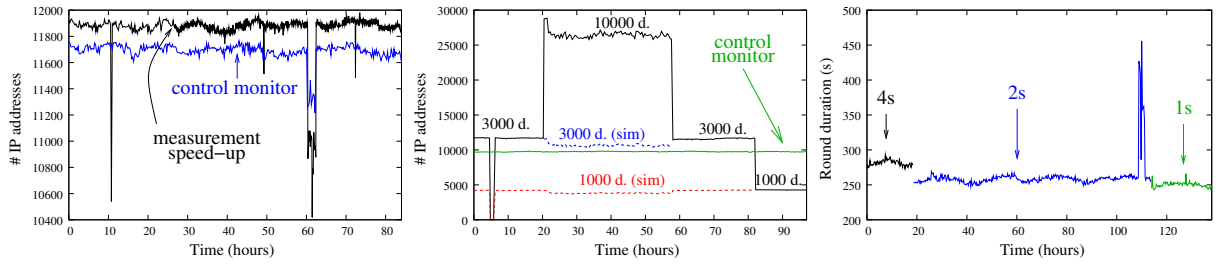


Figure 6: **Impact of measurement parameters. Left: impact of inter-round delay.** Number of distinct IP addresses viewed at each round. The bottom plot corresponds to a control monitor with the base parameters; the other monitor starts with the base parameters, and about 27 hours later we reduce the inter-round delay from 10 minutes to 1 (each ego-centered measurement takes around 4 minutes). **Center: impact of the number of destinations.** Number of distinct IP addresses viewed at each round. The plot close to $y = 10\,000$ corresponds to a control monitor with the base parameters. The other plain-line plot is produced by a monitor which starts with the base parameters, thus with a destination set D of size 3 000, changes to a set D' of 10 000 destinations containing D, goes back to D, and finally turns to a subset D'' of size 1 000 of D. In addition, the dotted plots are simulations of what we would have seen from this monitor with D during the measurement using D' (obtained by dropping all nodes and links which are on paths towards destinations that are not in D), and what we would have seen with D'' during the measurements using D or D' (obtained similarly). **Right: impact of timeout value.** Round duration (in seconds). The monitor starts with a timeout value of 4 s, then we change it to 2 s, and finally to 1 s.

We also considered other observables (like the number of stars seen at each round, and the number of packets received after the timeout), for measurements obtained from various monitors and towards various destinations; in all cases, the conclusion was the same: the base parameters meet our requirements.

Dataset.

Finally, we ran massive radar measurements. We use a wide set of more than one hundred monitors scattered around the world, provided by PlanetLab [40] and other structures (small companies and individual DSL links) [10].

In order to be as general as possible, and to simplify the destination setup, we used destinations chosen by sampling random valid IP addresses and keeping those answering to ping *at the time of the list construction.*

All our measurements were conducted with variations of the base parameters (see Table 1); wherever it is not explicitly specified, the parameters were the base ones. We ran measurements continuously during several weeks, with some interruptions due to monitors and/or local network shutdowns. The obtained data is available at [10].

Observed growth.

A most natural assumption is that radar measurements first discover basically all the IP addresses visible from the monitor, and then enter a regime where only major changes in the network cause new IP addresses to appear. Indeed, the number of distinct IP addresses one can see during a radar measurement is necessarily bounded, at the very least by the total number 2^{32} of possible

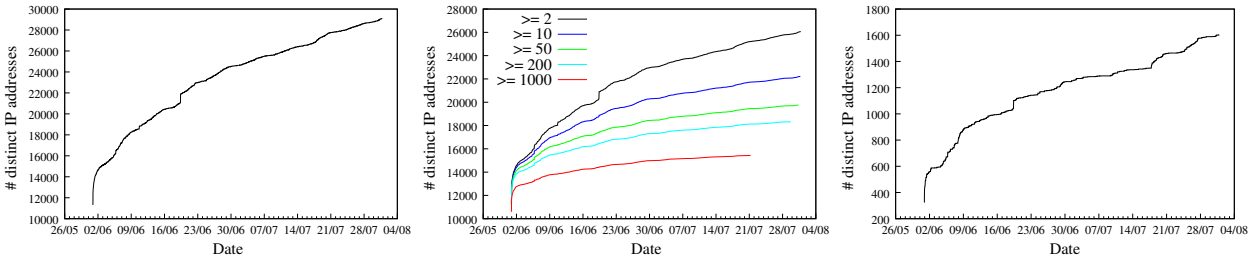


Figure 7: Left: Number of distinct IP addresses observed since the beginning of the measurement as a function of time. Center: Number of IP addresses observed in at least 2, 10, 50, 200 or 1 000 different rounds since the beginning of the measurement. Right: Number of distinct IP addresses seen with stable destinations only.

addresses. Therefore, after a sufficient number of rounds, one has necessarily seen all the IP addresses which can be seen from this monitor. One may expect that this number of rounds is rather small, but no study ever gave evidence for this intuition.

The study of our data however shows that it is actually quite false [27]. Figure 7 (left) shows that, even after several thousands rounds of measurements, the number of distinct IP addresses seen since the beginning still significantly grows. In the example of the figure, one discovers during the last week of two-months measurements 1 118 new IP addresses (on a total of 29 100).

This seemingly never-ending growth of the number of distinct IP addresses seen is strikingly counter-intuitive. However, one may try to explain it by two possible measurement artifacts.

One possible cause for these observations would be that some routers reply with random IP addresses; Figure 7 (center) show that this is not the case. Indeed, in this case, the growth would only concern IP addresses observed only once, or a small number of times. When restricting our observations to IP addresses observed in a larger and larger number of rounds (up to 1 000), we however still discover new addresses consistently. This means that addresses observed only once or a small number of times are not the main causes for our observations.

Another possible cause would be that some of our destinations are dynamic addresses, *i.e.* dynamically allocated to different hosts over time. Since such hosts could be in different locations, depending on network operation, these dynamic addresses could lead us to discover new paths and as a result new addresses in the measurements.

Figure 7 (right) shows that this is not the case. The idea is to select the destinations that were *stable during the measurements*. Using a similar approach to geolocation studies (see for instance [32]), we considered that a destination address is not dynamic if the address immediately before it in the measurements is always the same; 35 out of 3 000 destinations satisfied this condition. We then simulated the measurements by keeping only these stable addresses, and we still clearly see a constant appearance of new IP addresses: dynamic addresses are therefore not the cause of this renewal [4].

*References

[4] Note that our criterion for characterizing stable addresses is very restrictive; we do not imply that the addresses that do not satisfy it are dynamic. We tested other criteria, which provided the same results.

[10] Supplementary material, including programs and data. http://www-rp.lip6.fr/~latapy/Radar/.

[11] traceroute@home project. http://trhome.sourceforge.net/.

[12] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with paris traceroute. In *Proc. ACM/USENIX Internet Measurement Conference (IMC)*, Oct. 2006.

[13] B. Augustin, T. Friedman, and R. Teixeira. Multipath Tracing with Paris Traceroute. In *Proc. Workshop on End-to-End Monitoring, E2EMON*, May 2007.

[14] S. Branigan, H. Burch, W. R. Cheswick, and F. Wojcik. What can you do with traceroute? *IEEE Internet Computing*, 5(5), Sept./Oct. 2001.

[15] Q. Chen, H. Chang, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. The origin of power laws in internet topologies revisited. In *Proc. IEEE INFOCOM*, Jun. 2002.

[16] B Cheswick, H. Burch, and S. Branigan. Mapping and visualizing the internet. In *Proc. USENIX Annual Technical Conference*, Jun. 2000.

[17] B. Donnet, P. Raoult, and T. Friedman. Efficient route tracing from a single source. cs.NI 0605133, arXiv, May 2006.

[18] B. Donnet, P. Raoult, T. Friedman, and M. Crovella. Efficient algorithms for large-scale topology discovery. In *Proc. ACM SIGMETRICS*, Jun. 2005.

[19] B. Donnet, P. Raoult, T. Friedman, and M. Crovella. Deployment of an algorithm for large-scale topology discovery. *IEEE Journal on Selected Areas in Communications, Sampling the Internet: Techniques and Applications*, 24(12):2210–2220, Dec. 2006.

[20] F. Georgatos, F. Gruber, D. Karrenberg, M. Santcroos, A. Susanj, H. Uijterwaal, and R. Wilhelm. Providing active measurements as a regular service for ISPs. In *Proc. of Passive and Active Measurement Workshop*, 2001. See also the RIPE NCC TTM service: http://www.ripe.net/test-traffic/.

[21] R. Govindan and A. Reddy. An analysis of internet inter-domain topology and route stability. In *Proc. IEEE INFOCOM*, Apr. 1997.

[22] R. Govindan and H. Tangmunarunkit. Heuristics for internet map discovery. In *Proc. IEEE INFOCOM*, Mar. 2000.

[23] Ramesh Govindan and Vern Paxson. Estimating router ICMP generation delays. In *Proc. of Passive and Active Measurement Workshop*, March 2002.

[24] B. Huffaker, D. Plummer, D Moore, and k claffy. Topology discovery by active probing. In *Proc. Symposium on Applications and the Internet*, Jan. 2002.

[25] Craig Labovitz, G. Robert Malan, and Farnam Jahanian. Origins of internet routing instability. In *INFOCOM*, pages 218–226, 1999.

[26] M. Luckie. IPv6 scamper, 2005. WAND Network Research Group. See http://www.wand.net.nz/~mjl12/ipv6-scamper/.

[27] Clémence Magnien, Frédéric Ouédraogo, Guillaume Valadon, and Matthieu Latapy. Fast dynamics in internet topology: preliminary observations and explanations. 2008. Submitted.

[28] David B. Makofske and Kevin C. Almeroth. Real-time multicast tree visualization and monitoring. *Softw., Pract. Exper.*, 30(9):1047–1065, 2000.

[29] A. McGregor, H.-W. Braun, and J. Brown. The NLANR network analysis infrastructure. *IEEE Communications Mag.*, 38(5):122–128, May 2000. See also the NLANR AMP project: http://watt.nlanr.net/.

[30] T. Moors. Streamlining traceroute by estimating path lengths. In *Proc. IEEE International*

Workshop on IP Operations and Management (IPOM), Oct. 2004.

[31] Ricardo Oliveira, Beichuan Zhang, and Lixia Zhang. Observing the evolution of internet AS topology. In *Proc. ACM SIGCOMM*, 2007.

[32] Venkata N. Padmanabhan and Lakshminarayanan Subramanian. An investigation of geographic mapping techniques for internet hosts. In *SIGCOMM*, pages 173–185, 2001.

[33] J.-J. Pansiot. Local and dynamic analysis of internet multicast router topology. *Annales des télécommunications*, 62:408–425, 2007.

[34] J. J. Pansiot and D. Grad. On routes and multicast trees in the internet. *ACM SIGCOMM Computer Communication Review*, 28(1):41–50, Jan. 1998.

[35] S.-T. Park, A. Khrabrov, D.M. Pennock, S. Lawrence, C. Lee Giles, and L.H. Ungar. Static and dynamic analysis of the internet's susceptibility to faults and attacks. In *IEEE Infocom*, 2003.

[36] S.-T. Park, D. M. Pennock, and C. L. Giles. Comparing static and dynamic measurements and models of the internet's as topology. In *IEEE Infocom*, 2004.

[37] R. Pastor-Satorras, A. Vazquez, and A. Vespignani. Dynamical and correlation properties of the internet. *Physical Review Letters*, 87(25), 2001.

[38] R. Pastor-Satorras, A. Vazquez, and A. Vespignani. Large-scale topological and dynamical properties of internet. *Physical Review E*, 65(6), 2002.

[39] V. Paxson. End-to-end internet packet dynamics. *IEEE/ACM Trans. Networking*, 7(3):277–292, June 1999.

[40] PlanetLab Consortium. PlanetLab project, 2002. See `http://www.planet-lab.org`.

[41] Kamil Sarac and Kevin C. Almeroth. Tracetree: a scalable mechanism to discover multicast tree topologies in the internet. *IEEE/ACM Trans. Netw.*, 12(5):795–808, 2004.

[42] Puneet Sharma, Ed Perry, and Radhika Malpani. Ip multicast operational network management: Design, challenges and experiences. In *IEEE Workshop on IP Operations and Management (IPOM)*, 2002.

[43] Y. Shavitt and E. Shir. DIMES: Let the internet measure itself. *ACM SIGCOMM Computer Communication Review*, 35(5):71 – 74, October 2005.

[44] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with Rocketfuel. In *Proc. ACM SIGCOMM*, Aug. 2002.

[45] N. Spring, D. Wetherall, and T. Anderson. Scriptroute: A public internet measurement facility. In *Proc. 4th USENIX Symposium on Internet Technologies and Systems*, 2002. see also `http://www.cs.washington.edu/research/networking/scriptroute/`.

[46] M. Toren. tcptraceroute, April 2001. See `http://michael.toren.net/code/tcptraceroute/`.

[47] D. G. Waddington, F. Chang, R. Viswanathan, and B. Yao. Topology discovery for public IPv6 networks. *ACM SIGCOMM Computer Communication Review*, 33(3):59–68, Jul. 2003.

[48] Feng Wang, Nick Feamster, and Lixin Gao. Quantifying the effects of routing dynamics on end-to-end internet path failures. *Technical report (TR-05-CSE-03) in Department of Electrical and Computer Engineering in the University of Massachusetts at Amherst*, 2006.

[49] B. Yao, Viswanathan R., F. Chang, and D. Waddington. Topology inference in the presence of anonymous routers. In *Proc. IEEE INFOCOM*, Apr. 2003.