Université Pierre et Marie Curie
(umpc-paris 6)

Thèse de Doctorat

Spécialité :
Informatique

En co-tutelle avec :
Université de Ouagadougou

Par
Tounwendyam Frédéric Ouédraogo

Titre :

Une approche pour l'étude de la topologie de l'internet : vues ego-centrées et radar

Dirigée par :
Matthieu Latapy (Directeur)
Oumarou Sié (Directeur)
Clémence Magnien (Co-directeur)

Table des matières

1 Préliminaires **7**

1.1 Rappels . . . 7
1.1.1 Graphes . . . 7
1.1.2 Statistiques . . . 8
1.1.3 Notions de réseau et de l'internet . . . 10
1.2 Contexte et état de l'art . . . 12
1.2.1 Graphes de terrain . . . 12
1.2.2 Mesure de l'internet . . . 13
1.2.3 Biais dans la mesure . . . 17
1.2.4 Améliorations de l'outil de mesure . . . 18
1.2.5 Les grandes campagnes de mesure . . . 19
1.2.6 Dynamique de la topologie de l'internet . . . 21
1.2.7 Positionnement . . . 22

2 Un radar pour l'internet **23**

2.1 Méthodes de mesure . . . 23
2.1.1 Vue ego-centrée de la topologie . . . 24
2.1.2 L'outil `tracetree` . . . 25
2.1.3 Radar . . . 30
2.2 Paramètres de mesure . . . 31
2.2.1 Nombre de destinations . . . 31
2.2.2 *Timeout* des réponses . . . 32
2.2.3 Délai entre les passes . . . 33
2.2.4 Envoi et réception des paquets . . . 33
2.3 Mesures . . . 35
2.3.1 Choix des moniteurs et destinations . . . 35
2.3.2 Données . . . 36
2.4 Comparaison de `tracetree` et `traceroute` . . . 36
2.5 Filtre . . . 37
2.5.1 Filtrage et intégrité des données . . . 39
2.6 Analyses préliminaires . . . 40
2.6.1 Caractéristiques d'une vue ego-centrée . . . 40
2.6.2 Évolution de la vue ego-centrée . . . 41
2.6.3 Effet jour-nuit . . . 42
2.6.4 Dynamique et détection d'événements . . . 45

3 Stabilité et instabilité des vues ego-centrées **51**
3.1 Dynamique des adresses IP vues par un moniteur . . . 52
3.1.1 Nouvelles adresses observées par passe . . . 52
3.1.2 Pérennité des adresses IP . . . 54
3.2 Dynamique des systèmes autonomes (AS) . . . 57
3.2.1 Méthodologie et données . . . 57
3.2.2 Observations . . . 58
3.3 Rôle des changements de routage . . . 60
3.3.1 Découverte d'adresses IP . . . 61
3.3.2 Disparition d'adresses IP . . . 62
3.4 Conclusion . . . 63

4 Mesure de la topologie de l'internet **65**
4.1 Description des données . . . 65
4.2 Méthodologie . . . 66
4.2.1 Les explorations de G . . . 66
4.2.2 Courbes en niveaux de gris . . . 67
4.3 Impact du choix des moniteurs et des destinations . . . 67
4.3.1 Nombre de nœuds et nombre de liens . . . 68
4.3.2 Impact de l'ordre sur d'autres propriétés . . . 70
4.4 Propriétés statistiques . . . 72
4.4.1 Degré moyen et densité . . . 72
4.4.2 Coefficient de clustering . . . 74
4.4.3 Distance moyenne . . . 77
4.4.4 Distribution des degrés . . . 78
4.5 Conclusion . . . 81

Introduction

Depuis quelques années, un grand nombre de travaux ont été effectués sur la topologie de l'internet, afin de mieux comprendre sa structure. Ces études visent d'une part à mieux connaitre la topologie de l'internet, et d'autre part à comprendre les divers phénomènes pouvant se produire sur le réseau, tels que des phénomènes de propagation (par exemple la propagation d'une congestion), la résistance aux pannes, etc.

L'internet est constitué de plusieurs millions de machines. On peut étudier la topologie de l'internet à plusieurs niveaux : au niveau des adresses IP, des routeurs ou au niveau des systèmes autonomes (AS). L'étude de la topologie de l'internet passe d'abord par un processus de mesure car elle n'est pas directement disponible. Ce processus de mesure peut s'avérer très complexe à mettre œuvre.

Diverses méthodes ont été déployées pour fournir une carte de la topologie de l'internet, que ce soit au niveau des adresses IP, des routeurs ou des AS. Obtenir une carte la plus complète possible de la topologie de l'internet reste toutefois une question difficile, car les vues obtenues par ces différentes méthodes sont partielles.

De nombreux travaux ont de plus montré que le processus de mesure peut induire un important biais sur la topologie observée, c'est à dire que l'on peut observer des choses qui n'existent pas dans la topologie réelle. Il est donc apparu nécessaire d'étudier le processus de mesure, plus particulièrement le biais qu'il induit afin de trouver les moyens pour le corriger ou le réduire. Ces études relèvent du domaine de la *métrologie*.

Tous ces travaux s'inscrivent dans le cadre général de l'étude des graphes de terrain, c'est-à-dire les graphes issus de contextes réels. On peut citer par exemple le graphe des acteurs reliés s'ils ont joué dans un même film, les graphes d'interactions entre protéines ou entre gènes, les graphes obtenu des réseaux de transports routiers, d'électricité, les graphes du Web, et bien sûr la topologie de l'internet, et d'autres encore. Malgré leurs origines diverses, la plupart de ces graphes ont des propriétés *qualitatives* communes. De plus, un certain nombre de questions sont communes à l'ensemble des graphes de terrain. De nombreux travaux se sont ainsi intéressés à la mesure et à la métrologie des graphes de terrain, à leur analyse et à leur modélisation [35].

La dynamique de l'internet est un sujet extrêmement difficile. Il est déjà très difficile de mesurer la topologie de l'internet ; obtenir sa dynamique qui consiste à répéter plusieurs fois la mesure de sa topologie est encore plus délicat. Il y a cependant des travaux qui ont débuté dans le but de comprendre la dynamique de l'internet, mais ces travaux concernent principalement les systèmes autonomes (AS). Il faut aussi noter que la mesure est effectuée souvent à une échelle de temps élevée. Il existe donc des résultats qui concernent les AS, ou certaines anomalies de routage au niveau des adresses IP, mais en général on connait très peu de choses sur la dynamique de l'internet.

Afin d'aborder l'étude de la dynamique de la topologie au niveau IP, nous introduisons une nouvelle approche : nous nous intéressons à la dynamique observée à partir d'une machine donnée. Une *vue ego-centrée* consiste en ce qu'une machine peut voir de la topologie autour d'elle. C'est

un objet facile à mesurer en un temps relativement court, ce qui permet de répéter la mesure à une fréquence élevée et capturer une dynamique intéressante. Nous avons donc conçu un outil permettant de mesurer efficacement une vue ego-centrée avec lequel nous avons effectué massivement des mesures, dites *radar*, qui consistent en des mesures périodiques de vues ego-centrées.

Ce mémoire est organisé autour de ce principe comme suit.

Le chapitre 1 constitue un préambule. Nous rappelons les notions et les définitions nécessaires pour comprendre ce mémoire, puis nous présentons l'état de l'art ainsi que notre positionnement.

Dans le chapitre 2 nous présentons le nouvel outil de mesure que nous avons conçu, appelé `tracetree`. Il permet d'effectuer une mesure ego-centrée de manière rapide et efficace. Nous avons effectué une campagne de mesure à partir d'une centaine de moniteurs répartis dans le monde. Nous présentons les données obtenues et les analyses préliminaires que nous avons effectuées. Les données sont librement disponibles. Les résultats de ce chapitre ont fait l'objet d'une publication [37].

Dans le chapitre 3 nous analysons la dynamique des vues ego-centrées en utilisant les données radar. Nous nous intéressons à une observation inattendue : la découverte continuelle d'adresses IP par un moniteur à un rythme important. Nous présentons les différentes investigations que nous avons menées pour comprendre et apporter une explication à ce phénomène. Nous montrons que la part de la dynamique de routage dans ce phénomène est très importante. Ces travaux ont fait l'objet d'une publication [39].

Le chapitre 4 concerne une étude métrologique à partir des données radar. Nous étudions comment les propriétés de la topologie mesurée évoluent quand on fait varier les ensembles de moniteurs et de destinations. Nous montrons que l'influence du choix et du nombre de moniteurs et de destinations n'est pas la même selon la propriété considérée. Nous confirmons certains résultats obtenus par simulations, et apportons un nouvel éclairage sur cette question. Un article présentant ces travaux a été soumis.

Chapitre 1

Préliminaires

Ce chapitre contient les éléments qui permettront au lecteur de se situer dans le contexte de notre travail et de se familiariser avec certains termes et notions qui sont couramment utilisés dans le mémoire.

Ce chapitre est constitué de deux grandes parties. La première contient essentiellement un rappel sur les graphes et des notions liées à l'internet et plus généralement au réseau. La deuxième partie fait un état de l'art du domaine et donne le contexte dans lequel notre contribution se positionne.

1.1 Rappels

Cette section a pour but de rappeler les différentes notions, définitions et terminologies utilisées dans ce mémoire.

Nous commençons par rappeler des définitions de la théorie des graphes, puis nous donnons les principales notions statistiques utilisées pour décrire de grands graphes, et enfin nous présentons de manière rapide les notions du fonctionnement de l'internet qui nous seront nécessaires.

1.1.1 Graphes

Un *graphe* G est un couple d'ensembles V et E. On écrit $G = (V, E)$. V est un ensemble fini dont les éléments sont appelés les *sommets* (ou *nœuds*) de G. E est un sous-ensemble de $V \times V$ et ses éléments sont appelés les *arêtes* (ou *liens*) de G. Le graphe G est dit *orienté* quand on considère l'ordre dans les couples qui constituent l'ensemble E. Dans ce cas on utilise le terme *arc* (au lieu d'*arête*) pour désigner un élément de E. Toutes les définitions qui vont suivre concernent uniquement les graphes non orientés, sauf indication contraire.

Pour toute arête $e = (u, v) \in E$, on dit que l'arête e est *incidente* à u et v, qui sont appelés les *extrémités* de e. Pour toute arête $(u, v) \in E$, on dit que u est *voisin* de v et inversement. Le *voisinage* d'un nœud $u \in V$ est l'ensemble de ses voisins ; on le note $N(u) = \{v \in V | (u, v) \in E\}$. Le *degré* du nœud u, noté $d^\circ(u)$, est le nombre de ses voisins, c'est-à-dire le cardinal de son voisinage : $d^\circ(u) = |N(u)|$.

Un *chemin* est une séquence $x_1, x_2, \ldots, x_k$ de nœuds de G, avec $k \geqslant 2$, telle que pour tout $i \in [\![1, k-1]\!]$, $(x_i, x_{i+1}) \in E$. La longueur du chemin $x_1, x_2, \ldots, x_k$ est $k-1$. Un chemin est *élémentaire* si tous ses nœuds sont distincts.

Un *cycle* est une séquence $x_1, x_2, \ldots, x_k$ de nœuds de G, avec $k \geqslant 3$, tel que $x_1, x_2, \ldots, x_k$ est un chemin dans G et $(x_k, x_1) \in E$. La longueur du cycle $x_1, x_2, \ldots, x_k$ est k. Si G ne possède aucun

cycle, on dit que G est *acyclique*.

On appelle *distance* $d(u, v)$ entre deux nœuds u et v de G la longueur d'un plus court chemin qui les relie, s'il en existe un (il peut en exister plusieurs). Sinon on fixe par convention $d(u, v) = \infty$.

Un graphe $H = (V_H, E_H)$ est un *sous-graphe* de $G = (V, E)$ si $V_H \subset V$ et $E_H \subset E$. $G = (V, E)$ est un *graphe complet* si $E = V \times V$. On désigne par K_n un graphe complet à n nœuds.

Une *composante connexe* d'un graphe est un ensemble C de nœuds maximal par rapport à l'inclusion tel que, pour toute paire de nœuds x, y dans C, il existe un chemin de x vers y. Un graphe est dit *connexe* s'il est composé d'une seule composante connexe.

Un *arbre* est un graphe non orienté connexe et acyclique. Dans un arbre, il existe un et un seul chemin entre toute paire de nœuds. Un graphe acyclique non connexe est appelé une *forêt*. Un *arbre enraciné* est un arbre dans lequel l'un des nœuds se distingue des autres. On appelle ce nœud particulier *racine*.

Soit u un nœud d'un arbre enraciné T de racine r. Un nœud x quelconque sur le chemin de r à u est appelé *ancêtre* de u. Si x est un ancêtre de u, alors u est un *descendant* de x. Si le dernier lien sur le chemin de la racine r vers le nœud u est (x, u), alors x est le *père* de u et u est un *fils* de x. La racine est le seul nœud qui n'a pas de père. Deux nœuds qui ont le même père sont appelés *frères*. Un nœud qui n'a pas de fils est appelé *feuille*.

La longueur du chemin entre la racine r et le nœud u est appelée *profondeur* de u dans l'arbre. La plus grande profondeur est la *hauteur* de l'arbre.

1.1.2 Statistiques

De nombreuses statistiques ont été introduites dans l'étude des graphes de terrain. Nous allons présenter quelques statistiques parmi les plus utilisées.

Nombre de nœuds, de liens, de triplets et de triangles

Ces statistiques consistent à dénombrer les structures de base de un à trois nœuds dans le graphe, voir figure 1.1. Il s'agit du nombre de nœuds, de liens, de triplets et de triangles qui interviennent également dans d'autres définitions statistiques plus complexes.

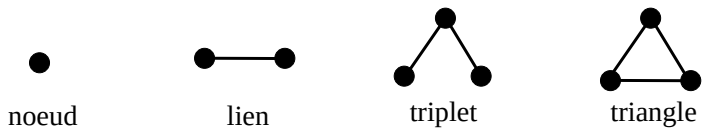


FIG. 1.1 – Structures élémentaires de trois nœuds au plus.

Le nombre de triplets est le nombre de paires de liens incidents à un même nœud. Le nombre de triangles est le nombre d'ensembles de trois nœuds tous reliés deux à deux.

Degré moyen, densité

Le *degré moyen* d'un graphe, noté d°, est la moyenne des degrés de tous ses nœuds :

$$d^\circ = \frac{1}{n}\sum_v d(v) = \frac{2m}{n}.$$

La *densité* du graphe G, notée δ, est le rapport entre son nombre de liens et le nombre maximum de liens qui peuvent exister entre les nœuds. En d'autres termes, il s'agit du nombre de liens du graphe divisé par le nombre de liens du graphe complet de même nombre de nœuds :

$$\delta = \frac{2m}{n(n-1)}.$$

Distance moyenne et diamètre

La distance moyenne d'un nœud u dans un graphe vers tous les autres nœuds est :

$$d(u) = \frac{1}{n-1}\sum_{v\neq u} d(u,v).$$

La *distance moyenne* du graphe est la moyenne des distances de tous ses nœuds :

$$D = \frac{1}{n}\sum_u d(u).$$

Notons que la définition ci-dessus n'a d'intérêt que si le graphe est connexe. En pratique, si le graphe n'est pas connexe (ce qui est souvent le cas) on considèrera la distance moyenne dans la plus grande composante connexe.

Le *diamètre* est la plus grande distance entre deux nœuds du graphe.

Coefficients de *clustering*

Le concept de *clustering* évalue la tendance générale observée dans les graphes de terrain à avoir des cliques dans le voisinage des nœuds. Le coefficient de *clustering* mesure la probabilité que si deux nœuds i et l sont connectés à un même nœud j, i et l soient aussi connectés.

Il existe dans la littérature deux définitions légèrement différentes pour calculer le coefficient de *clustering*, qu'on désigne en général par *clustering global* et *clustering local*.

Le coefficient de *clustering* global (gc) compte le nombre de triplets connectés qui sont aussi des triangles. Aussi appelé rapport de transitivité, il est exprimé de la façon suivante :

$$gc = \frac{3N_\triangledown}{N_\vee},$$

où $N_\triangledown$ désigne le nombre de triangles et $N_\vee$ désigne le nombre de triplets connectés dans le graphe.

Le coefficient de *clustering* local $lc(v)$ d'un nœud v est la probabilité qu'un lien existe entre deux de ses voisins. On l'exprime formellement comme suit :

$$lc(v) = \frac{2.|E_{N(v)}|}{d(v).(d(v)-1)},$$

où $E_{N(v)} = E \cap (N(v) \times N(v))$ est l'ensemble des liens qui existent entre les voisins du nœud v. Il n'est défini que pour les nœuds dont le degré est au moins égal à 2. Le coefficient de *clustering* local d'un graphe est la moyenne des coefficients de *clustering* locaux de ses nœuds :

$$lc = \frac{1}{|v \in V, d(v) \geqslant 2|} \sum_{v, d(v) \geqslant 2} lc(v).$$

Le *clustering* local peut être interprété comme la densité du voisinage d'un nœud et dans ce sens il mesure la densité locale.

Distribution des degrés

La distribution des degrés p_k d'un graphe est définie comme étant la probabilité qu'un nœud choisi de façon aléatoire dans le graphe ait un degré k.

La distribution des degrés peut être homogène (toutes les valeurs sont proches de la moyenne, comme dans une distribution de Poisson ou Gaussienne) ou hétérogène (il y a un écart important entre les valeurs, avec plusieurs ordres de grandeur entre elles : il y a beaucoup de nœuds de faible degré et très peu de nœuds de fort degré, et tous les comportements intermédiaires).

Quand une distribution est hétérogène, il y a un sens à mesurer l'hétérogénéité plutôt que la valeur moyenne, qui est peu représentative. Dans certain cas, cela est possible en faisant correspondre la distribution à une loi de puissance, c'est-à-dire une distribution de la forme $p_k \sim k^{-\alpha}$. Alors l'exposant α peut être considéré comme un indicateur du niveau d'hétérogénéité de la distribution.

1.1.3 Notions de réseau et de l'internet

Nous allons donner les définitions des termes et des notions de réseau informatique que nous utilisons dans ce mémoire. Certaines définitions ont souvent une équivalence dans la théorie des graphes mais la connotation réseau est nécessaire pour saisir le contexte.

Pour définir l'*internet*, on dit souvent que c'est un réseau de réseaux. Un *réseau* est un ensemble d'ordinateurs (qu'on appelle généralement hôtes) connectés de telle sorte que chacun d'eux puisse communiquer avec tous les autres. La connexion entre les hôtes est possible grâce à deux composantes : le matériel et le logiciel. Le matériel réfère aux éléments physiques du réseau, tels que les ordinateurs et les lignes de communication. La composante logicielle fait référence à l'ensemble des programmes qui se chargent de l'échange des données à travers les composantes physiques. Les logiciels qui définissent les opérations sur le réseau sont souvent appelés *protocoles* parce qu'ils définissent un ensemble de standards qui régissent la communication.

Le protocole TCP/IP de l'internet contient une famille de protocoles dans laquelle le protocole de contrôle de transmission (*Transmission Control Protocol* : TCP) et le protocole internet (*Internet Protocol* : IP) sont les plus importants. Le protocole IP définit un espace d'adresses uniques. Pour rejoindre l'internet, chaque hôte doit donc posséder une adresse unique qui est utilisée pour l'identifier afin de pouvoir communiquer avec lui. Une adresse IP est un nombre de 32 bit divisé en quatre champs. Chaque champ, séparé par un point, est spécifié par un octet (8 bit). Ainsi chaque adresse IP est constituée de quatre nombres pris entre 0 et 255, par exemple 45.249.45.8.

Les *routeurs* sont les éléments de base dans l'internet. Leur principal rôle est d'acheminer l'information de proche en proche depuis sa source jusqu'à sa destination. Un routeur a plusieurs *interfaces*, qui sont des ports d'entrée et de sortie lui permettant de communiquer avec d'autres routeurs. L'information transite donc de routeur en routeur depuis la source jusqu'à la destination.

Les routeurs jouent un rôle essentiel dans l'internet car ils assurent sa connexité. Chaque routeur maintient une *table de routage*, qui indique, pour chaque destination, auquel de ses voisins les paquets correspondants doivent être envoyés.

Le *routage* est le mécanisme par lequel les chemins sont choisis pour acheminer les données vers une destination. Il s'agit d'un processus décentralisé. Cette caractéristique a le grand avantage que tous les routeurs ont en principe une importance égale : la défaillance de l'un d'eux n'empêche pas le fonctionnement du réseau, puisque les routeurs peuvent décider en temps réel de faire suivre les paquets à travers un chemin différent.

Une *route* est un chemin suivi par des paquets dans le réseau. Sa *longueur* est le nombre de *sauts*, c'est à dire le nombre de passages d'une machine à la suivante, effectués par les paquets.

L'équilibrage de charge ou *load balancing* est un système de routage des paquets qui permet d'améliorer la fiabilité et d'optimiser l'utilisation des ressources. Il consiste à envoyer des paquets pour une même destination sur plusieurs routes différentes. Ainsi deux paquets successifs d'une communication peuvent à partir d'un même routeur emprunter deux chemins différents.

Il se fait le plus souvent à travers les protocoles de routage intra-domaine tels OSPF [44] et IS-IS [10] qui supportent les chemins multiples à égal coût.

Les principales stratégies d'acheminement pour l'équilibrage de charge sont :

- Le *load balancing* par-flot, qui consiste à maintenir les paquets d'un même flot vers une même sortie.
- Le *load balancing* par-paquet, qui ne prête aucune attention à garder les paquets d'un même flot ensemble sur la même sortie mais assure une charge bien répartie sur les différentes sorties.
- Le *load balancing* par-destination, qui peut être vu comme une forme basique du *load balancing* par-flot car il se base sur l'adresse de destination pour orienter les paquets.

L'internet est composé de domaines administrés de façon autonome, appelés *systèmes autonomes* (AS ou *autonomous system*), qui varient en taille, en étendue géographique et en fonctionnalités.

Chaque AS possède un numéro qui l'identifie de façon unique dans le réseau. En général, on peut répartir ces systèmes autonomes en deux types : AS de *transit* et AS de *bord*. Les AS de transit correspondent à l'épine dorsale (ou *backbone*) et assurent la connectivité nationale, intercontinentale, ... Leur but principal est d'assurer la connectivité entre les AS de bord, sans que ces derniers aient à se connecter directement.

Chaque AS gère les routeurs qui sont sous son autorité et définit comme il le souhaite le mécanisme de routage interne entre ses routeurs. On peut distinguer deux types de routeurs dans un AS : les routeurs internes et les *routeurs de bordure* qui font office de frontière de l'AS. Ces routeurs sont directement reliés à d'autres routeurs de même type appartenant à d'autres AS.

BGP (Border Gateway Protocol) est un protocole de routage (ou d'échange d'informations de routes) des systèmes autonomes. Sa première fonction est d'échanger les informations sur l'accessibilité réseau avec d'autres routeurs BGP.

Le programme `ping` est un outil de test basique pour évaluer les performances de l'internet. Il est basé sur le protocole internet de contrôle de message (Internet Control Message Protocol : ICMP). `ping` envoie des paquets à un hôte cible qui est censé répondre aux paquets reçus. Le programme mesure alors le RTT, *i.e.* le temps mis par chaque paquet pour faire un aller-retour entre l'hôte de départ et la destination. L'outil `ping` intervient dans la maintenance du réseau internet. En général, il sert à diagnostiquer les pannes et les congestions.

1.2 Contexte et état de l'art

Dans cette section, nous allons présenter le contexte dans lequel se situe cette thèse. Nous commencerons par une présentation générale du sujet des graphes de terrain dont fait partie le graphe de l'internet, puis nous présenterons les travaux effectués dans le domaine de la mesure et de la métrologie de l'internet, ainsi que de l'étude de sa dynamique. Nous allons aussi présenter les différents problèmes ouverts et le positionnement de cette thèse.

1.2.1 Graphes de terrain

L'étude des *graphes de terrain* ou *grands réseaux d'interactions* concerne les graphes ou réseaux issus de contextes réels. Les graphes de terrain regroupent une large variété d'objets. On peut citer :

- ceux issus de l'internet (relations entre systèmes autonomes, routeurs et liens entre eux, sauts au niveau IP entre interfaces, par exemple) ;
- les graphes du Web (ensembles de pages Web, et liens entre elles) ;
- les réseaux *overlays* (par exemple les réseaux pair-à-pair) ;
- les réseaux d'échanges (courriers électroniques, fichiers, ...) ;
- les réseaux sociaux (relations entre individus ou groupes d'individus) ;
- les réseaux biologiques (interactions protéiques, topologie du cerveau, réseau trophiques) ;
- les réseaux de mots (cooccurrence ou relation de synonymie) ;
- les réseaux ferroviaires, routiers ou de distribution d'électricité.

Bien que ces graphes soient issus de divers contextes, un résultat important montre qu'ils appartiennent à une même classe : la plupart des graphes de terrain ont des propriétés non-triviales en commun [69, 7].

Les propriétés communes des grands graphes de terrain communément admises sont :

- une très faible densité (proche de 0), ou de façon équivalente un degré moyen très petit devant n ; autrement dit, quand on choisit une paire de nœuds au hasard la probabilité qu'ils soient liés est très faible.
- une composante connexe géante, c'est-à-dire regroupant l'immense majorité des nœuds (alors que les autres, s'il y en a, sont très petites) ;
- une distance moyenne et un diamètre faibles dans cette composante ;
- une distribution de degrés hétérogène et souvent raisonnablement bien approximée par une loi de puissance : $p_k \sim k^{-\alpha}$ pour un α généralement entre 2 et 3 ;
- une densité locale, généralement mesurée par le coefficient de clustering, élevée (de plusieurs ordres de grandeur supérieure à la densité) ; autrement dit, on a significativement plus de chances que deux nœuds soient reliés si ce sont deux voisins d'un même nœud que s'ils sont choisis aléatoirement.

Ces propriétés [1] constituent aujourd'hui un ensemble de référence, considéré comme fondamental, souvent complété par diverses autres propriétés selon le cas étudié.

D'autre part de très nombreuses questions qui se posent sur ces divers graphes sont en fait très générales. Ces problématiques se répartissent naturellement en quatre grandes familles :

Mesure et métrologie. La plupart des grands graphes de terrain n'étant pas directement disponibles, la connaissance qu'on en a passe par une opération de mesure. Celle-ci peut s'avérer extrêmement complexe, et la mettre en œuvre est alors un défi en soi. De plus, elle fournit généralement

[1] Il est important de remarquer que ces propriétés ne sont que qualitatives.

une vision partielle et biaisée de l'objet réel ; il est alors nécessaire d'étudier ce biais, de tenter de le corriger, et de mener une réflexion plus approfondie sur les conclusions que l'on peut effectivement tirer de nos observations.

Analyse. Étant donné un grand graphe de terrain, une première étape naturelle est de tenter d'en décrire la forme, c'est à dire les propriétés principales, les caractéristiques. Ceci se fait par le biais de notions statistiques et/ou structurelles, visant à faire une synthèse des principales caractéristiques du graphe. La définition de telles propriétés est toutefois loin d'être triviale, ainsi que l'évaluation de leur pertinence. De même, l'interprétation des descriptions obtenues peut s'avérer délicate.

Modélisation. Afin d'expliquer la nature des observations, de pouvoir développer des résultats mathématiquement rigoureux, et de mener les simulations adéquates, il est important de capturer les propriétés observées en pratique dans des modèles de graphes de terrain. Ceci se fait généralement par le tirage aléatoire de graphes dans certaines classes ou par un processus explicite de construction de graphes. On obtient ainsi des graphes *artificiels*, représentatifs des propriétés choisies.

Algorithmique. Enfin, l'étude de très grands graphes a naturellement besoin de l'algorithmique. Tout d'abord, le contexte des graphes de terrain soulève des questions algorithmiques originales (comme la détection de communautés par exemple), *i.e.* qui ne se posaient pas précédemment. De plus, les solutions usuelles à des problèmes algorithmiques classiques, comme par exemple le calcul du diamètre, ne sont plus applicables du fait de la taille des graphes considérés. Par contre, les propriétés rencontrées en pratique peuvent être mises à profit pour concevoir des algorithmes efficaces sur les graphes de terrain.

Ces quatre grandes familles de problématiques ont un complément naturel : l'étude des phénomènes ayant lieu sur ces réseaux, comme la diffusion d'information ou de virus, la résistance aux pannes ou aux attaques, les comportements sociaux ou biologiques, le routage et les congestions, etc. L'étude de ces phénomènes soulève aussi des questions de mesure et métrologie, d'analyse, de modélisation et d'algorithmique, et joue naturellement un rôle important dans le domaine, notamment parce que cela motive l'étude des topologies et montre que les propriétés observées ont effectivement un impact fort sur les phénomènes qui s'y produisent.

Les travaux que nous présentons dans ce mémoire concernent principalement les deux premiers points de la chaîne, c'est-à-dire la mesure et l'analyse.

1.2.2 Mesure de l'internet

Caractériser comment les routeurs, ordinateurs et liens physiques sont interconnectés dans l'internet est un problème difficile pour plusieurs raisons. Premièrement, le réseau internet est un système auto-organisé dont l'évolution ne suit pas un plan préétabli. De plus, l'internet est intrinsèquement un système hétérogène composé de plusieurs réseaux avec une large variété technique et administrative. Les différents réseaux sont de tailles diverses, allant de petits réseaux locaux à de grands *backbone* transcontinentaux. Cette différence de taille est reflétée dans les différentes stratégies d'administration qui rend le routage sur internet difficilement prévisible avec des phénomènes hétérogènes. Un autre raison vient du fait que l'internet est un système qui évolue rapidement avec le temps. En effet, les routeurs et liens sont ajoutés ou supprimés en fonction de contraintes techniques et économiques, conduisant à une structure physique très complexe qui ne se conforme à aucun plan optimisé. La taille de l'internet qui est en croissance continuelle ne facilite pas non plus son étude. L'internet a eu une croissance exponentielle depuis sa naissance [52]. De nos jours, l'internet est un objet à grande échelle dont les propriétés en général ne peuvent être déduites des propriétés obtenues localement.

La topologie de l'internet, comme pour la plupart des graphes de terrain, n'est donc pas directe-

ment disponible : on n'a pas de carte de sa topologie. En lieu et place, on se base sur des méthodes qui consistent à sonder les machines et des heuristiques qui permettent d'établir un échantillon de la topologie. La mise en œuvre de ces méthodes passe par l'intermédiaire d'un outil de sonde, comme par exemple `traceroute`.

On peut s'intéresser à la topologie de l'internet vue à plusieurs niveaux : en particulier au niveau des adresses IP, des routeurs et des AS. Nous détaillons ci-dessous les principales méthodes de mesure dans ces trois cas.

Niveau IP

Pour obtenir les informations sur les liens entre routeurs, il est possible de contacter les routeurs pour savoir avec quels autres routeurs ils sont reliés, à partir de leur table de routage, mais très peu de routeurs l'autorisent. On peut également sonder le réseau internet à l'aide d'un outil (c'est la méthode la plus utilisée).

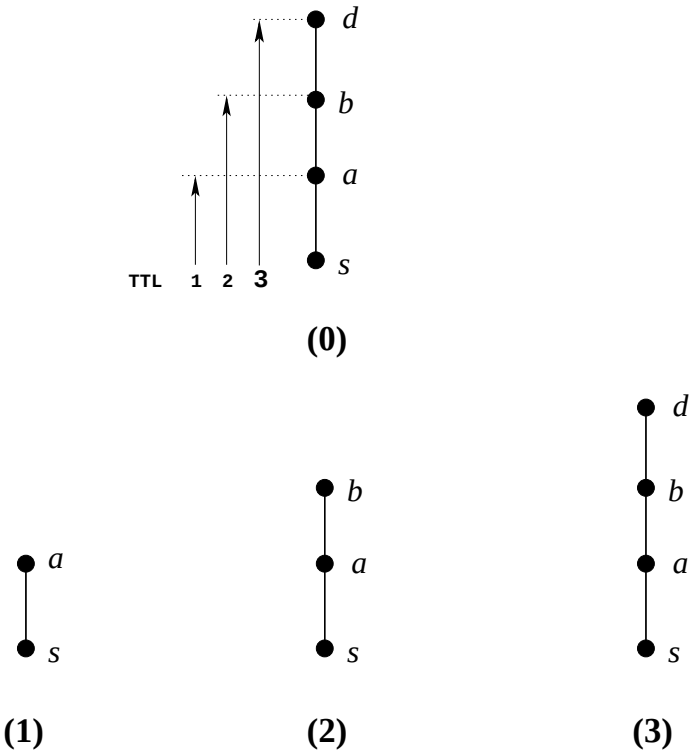


FIG. 1.2 – Principe de `traceroute`. (0) La topologie à mesurer : le nœud `s` est le moniteur et `d` est la destination. Elle est située à distance 3 du moniteur. Chaque flèche verticale représente un paquet avec le TTL correspondant, envoyé vers la destination `d`. (1) Un paquet de TTL 1 est envoyé à d. On découvre le nœud `a`. (2) Un paquet de TTL 2 est envoyé à d. On découvre le nœud `b`. (3) Un paquet de TTL 3 est envoyé à d et on la découvre.

Avec l'outil `traceroute` [31], il est possible de connaître une série d'interfaces entre le moniteur et la destination. Pour cela `traceroute` envoie des sondes pour découvrir les interfaces à chaque saut IP entre le moniteur et la destination. Le principe de `traceroute` est simple. Il consiste à augmenter progressivement la durée de vie des sondes en jouant sur la valeur du champ TTL (*Time To Live*).

Le champ TTL d'un paquet indique le nombre maximum de routeurs qu'il peut traverser. À chaque fois qu'un routeur transfère un paquet, il diminue de un la valeur du TTL. Si cette valeur atteint zéro, le routeur supprime le paquet, puis en informe l'expéditeur en lui envoyant un message d'erreur. Celui-ci peut donc en déduire la présence du routeur qui envoie le message d'erreur sur la route vers la destination. En envoyant un paquet de TTL 1 vers une destination donnée, on obtient l'adresse IP de l'interface qui est à un saut du moniteur. Envoyer un paquet de TTL 2 permet de découvrir l'interface qui est à deux sauts du moniteur, et ainsi de suite jusqu'à la destination. La figure 1.2 illustre ceci. On peut ainsi reconstituer le chemin suivi par les paquets entre le moniteur et la destination. En faisant de telles mesures depuis plusieurs moniteurs vers plusieurs destinations, on arrive à voir une partie non négligeable du réseau.

Des méthodes plus complexes utilisent le *source-routing* (ou LSRR : *loose source record route*) qui permet de trouver un chemin vers une destination en passant par un certain nombre de routeurs intermédiaires choisis [24]. Cette technique peut être utilisée pour détecter des anomalies dans le réseau ou éviter de surcharger certains liens, mais elle peut aussi servir pour le piratage informatique. Par conséquent elle n'est plus utilisable de nos jours, un grand nombre de routeurs détruisant les messages faisant appel à cette fonction.

Niveau routeur

Chacune des interfaces d'un routeur correspond à une connexion physique avec un autre routeur, et chaque interface est identifiée par une adresse IP différente. La méthode de mesure décrite plus haut permet de découvrir un chemin d'interfaces. Chaque paquet envoyé découvre une seule interface de chaque routeur traversé : c'est celle par laquelle le paquet arrive à ce routeur. Il est donc possible que les paquets venant de différents moniteurs arrivent sur différentes interfaces d'un même routeur. La figure 1.3 montre un exemple simple de la différence entre la connexion physique entre routeurs et les données obtenues avec `traceroute`. Dans cette configuration, on a quatre routeurs connectés entre eux à travers six interfaces différentes. En agrégeant les différentes vues obtenues par des mesures avec `traceroute`, on obtient un graphe dont les nœuds sont les interfaces des routeurs.

Pour obtenir la topologie au niveau des routeurs, il faut arriver à identifier les différentes interfaces du même routeur [24, 61] : ce procédé s'appelle *résolution d'alias* ou *anti-aliasing* car il s'agit de trouver tous les *alias* (interfaces) d'un même routeur. Cette résolution d'alias pose un problème complexe pour lequel il n'y a pas de méthode sûre, ce qui laisse place à plusieurs heuristiques. L'une des méthodes [24] consiste à utiliser le fait que les routeurs qui répondent à un message le font souvent en donnant l'adresse de l'interface par laquelle ils répondent. Ainsi, en envoyant un message vers une interface x d'un routeur, on peut obtenir une réponse de l'interface y, ce qui signifie que x et y appartiennent au même routeur. Ces opérations doivent être répétées souvent car les routes peuvent changer et donc l'interface utilisée par le routeur aussi. Il faut aussi noter que certains routeurs ne répondent pas à ces messages.

D'autres méthodes ont été introduites pour tenter de prédire les interfaces probables d'un même routeur. Le principe est de supposer que les adresses IP des interfaces d'un même routeur sont généralement proches. On peut également regarder les identifiants des paquets ou supposer que les temps mis pour atteindre deux interfaces d'un même routeur sont similaires [61]. Une fois les interfaces de tous les routeurs identifiées, on peut définir le réseau des routeurs en considérant que deux routeurs sont reliés si deux de leurs interfaces le sont.

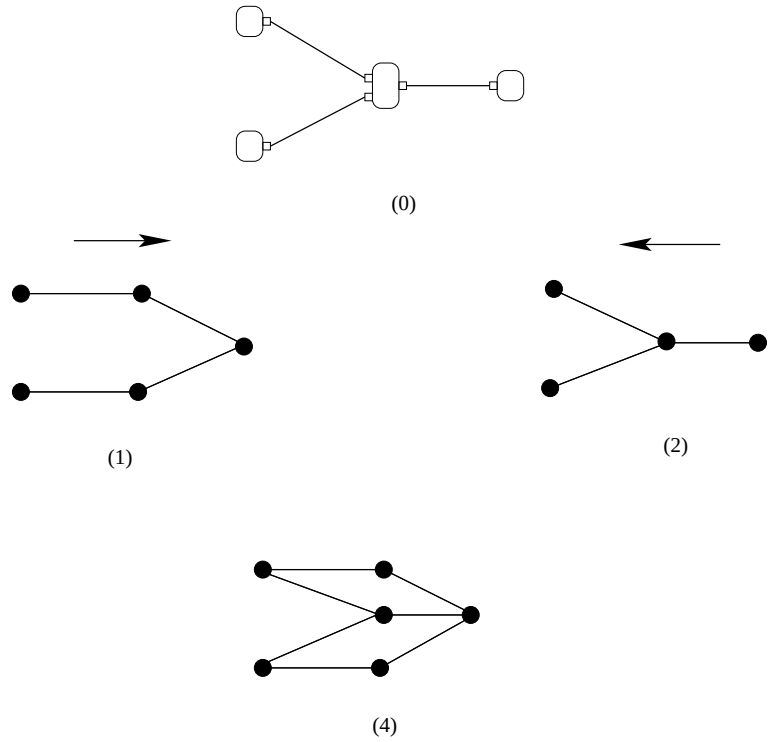


FIG. 1.3 – Différences entre la topologie réelle et les données obtenues avec `traceroute`, dues aux interfaces multiples d'un même routeur. (0) la topologie à mesurer. Le routeur du milieu possède trois interfaces correspondant à trois adresses IP. (1) Le graphe obtenu par des mesures effectuées depuis les moniteurs de gauche. Chaque interface vue est un nœud différent dans le graphe. (2) Le graphe obtenu par des mesures effectuées depuis les moniteurs de droite. On obtient une vue différente de la topologie. (4) En fusionnant (1) et (2) on obtient une vue de la topologie avec un nombre d'interfaces plus élevé que le nombre de routeurs.

Niveau AS

Le niveau physique décrit précédemment ne prend pas du tout en compte le fait que les routeurs appartiennent à des AS.

Il est possible d'obtenir les liens entre AS en faisant usage des tables de routage BGP des routeurs qui s'occupent des liaisons inter-domaines. Ces tables de routage permettent de savoir, étant donnée l'adresse du destinataire, à quel AS transférer un message, et aussi tout les AS à traverser pour atteindre celui de la destination. Cependant, il est difficile d'obtenir ces tables de routage.

En principe, une table de routage BGP devrait couvrir l'ensemble des adresses d'AS connues [2], mais ne fournit pas tous les liens qui existent entre les AS. C'est en agrégeant les informations de plusieurs tables de routage BGP qu'on parvient à obtenir une vue aussi complète que possible de la topologie au niveau AS.

On peut aussi obtenir une vue de la topologie au niveau AS en agrégeant les adresses IP ou routeurs appartenant aux AS correspondants. C'est cette méthode que le projet CAIDA utilise pour fournir une visualisation de la topologie de l'internet au niveau des AS.

Enfin, il est possible d'explorer le réseau d'AS avec un outil comme par exemple *Hermes* [11] de

[2] Ce n'est pas le cas. Quand un routeur BGP n'a pas de chemin d'AS pour une certaine adresse, il envoie le trafic de cette adresse à un routeur par défaut.

la même manière que `traceroute`.

1.2.3 Biais dans la mesure

Pour les méthodes de mesure décrites plus haut, il est montré qu'il est impossible de voir tous les routeurs et les liens, par conséquent les cartes sont incomplètes. Cependant il y a un problème plus grave encore : les échantillons issus de ces méthodes de mesure peuvent être biaisés. De nombreux travaux ont été effectués pour évaluer la précision des cartes obtenues de la topologie de l'internet, voir par exemple [1, 2, 8, 15, 17, 16, 26, 27, 28, 30, 34, 36, 54, 58, 60]. Toutes ces études ont montrés par des arguments solides que les cartes de l'internet actuellement disponibles sont incomplètes et biaisées.

Biais lié à la méthode

Comme décrit plus haut, la mesure de la topologie de l'internet consiste en général à collecter un grand nombre de chemins. Les chemins sont ensuite fusionnés pour obtenir un graphe qui représente la topologie. La question est de savoir quelle est la précision d'un échantillon obtenu avec cette méthode de mesure. Dans un cas typique de mesure conduite avec l'outil `traceroute`, on peut utiliser de nombreuses destinations qui sont passives, tandis que les moniteurs sont actifs et requièrent le déploiement d'une infrastructure dédiée de mesure, et sont par conséquent peu nombreux. Ainsi, quand on fait la mesure avec un nombre de moniteurs relativement petit vers un plus grand ensemble de destinations, les nœuds et les liens proches des moniteurs ont une probabilité plus élevée d'être vus que ceux qui sont plus éloignés. Il est montré que cela peut induire un biais dans l'échantillon.

La distribution des degrés est l'une des propriétés de l'internet pour laquelle le biais a été l'objet d'études plus approfondies [1, 2, 15, 16, 34, 54, 58]. L'un des résultats importants a été de montrer que le nombre de moniteurs et de destinations a une grande influence sur la distribution des degrés observée sur l'échantillon : on peut observer une distribution des degrés en loi de puissance alors que la topologie sous-jacente a une distribution des degrés qui suit une loi de Poisson. Ce résultat a été approfondi par [27, 28] qui ont montré en utilisant différents modèles de graphes que le biais dépend de la topologie de l'objet mesuré. Par exemple, si l'on considère un graphe généré selon le modèle aléatoire [3], et donc avec une distribution des degrés en loi de Poisson, l'échantillon obtenu en l'explorant avec un petit nombre de moniteurs peut donner, sous certaines conditions, une distribution des degrés en loi de puissance. Cependant si la topologie sous-jacente a une distribution des degrés en loi de puissance, même avec un petit nombre de moniteurs on obtient avec l'échantillon une distribution similaire à celle de que l'objet réel [34, 27].

Des travaux ont étudié explicitement l'influence du nombre de moniteurs sur le biais induit dans la mesure [8, 27, 28, 60]. Il existe d'autres biais dans la mesure de la topologie de l'internet, mais nous ne les détaillerons pas tous ici.

Notons par exemple qu'un moniteur peut observer des liens éloignés de lui tout en ne voyant pas des liens très proches (par exemple un lien entre deux de ses voisins immédiats ne sera quasiment jamais vu). Plus généralement, la vision obtenue avec un moniteur est proche d'un arbre, ce qui est plus lié à la méthode de mesure qu'à une réalité sous-jacente. De même, le graphe obtenu est la fusion d'un nombre relativement faible de graphes très arborescents, chacun étant la vision obtenue par un moniteur. Ceci peut avoir une influence importante sur les propriétés du graphe final.

[3] Graphe aléatoire d'Erdös et Rényi [21].

Biais lié à l'outil

En plus des biais liés intrinsèquement à la méthode de mesure, d'autres peuvent être induits par l'outil de mesure.

En général, on suppose que le routage reste relativement stable et qu'à l'échelle du temps d'exécution d'un `traceroute` la probabilité qu'il change est faible. Il est montré dans [5, 66] que cette hypothèse est loin de la réalité, surtout en raison de la présence de nombreux routeurs qui déploient le *load balancing*.

Là où il y a du *load balancing* il n'y a pas un chemin unique du moniteur à la destination. Avec l'outil `traceroute`, on n'est pas assuré d'obtenir un chemin existant entre le moniteur et la destination : des faux liens qui n'existent pas peuvent être mesurés. Cela est dû au principe même de `traceroute` qui découvre chaque nœud par un paquet envoyé dans le réseau, et les routeurs qui font du *load balancing* peuvent diriger les paquets, successifs d'un même `traceroute` sur des chemins différents. Par conséquent deux paquets successifs peuvent découvrir deux nœuds appartenant à deux chemins différents. Par exemple, dans le cas de la figure 1.4, L oriente le paquet de TTL 7 vers A et celui de TTL 8 vers B, ce qui conduit à inférer un faux lien entre A et D.

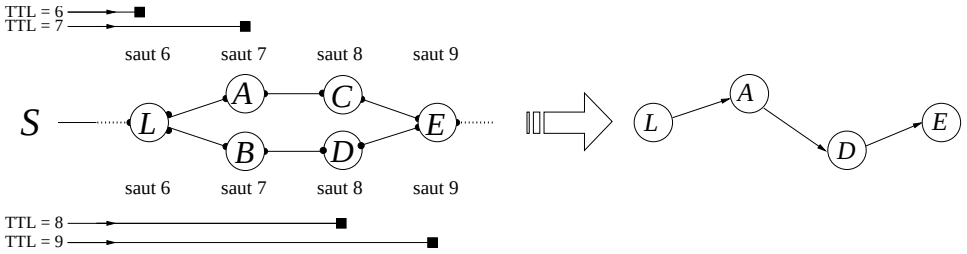


FIG. 1.4 – Cas typique de vision erronée de la topologie obtenue par traceroute due à la présence d'un routeur équilibrant la charge entre deux chemins : L envoie le premier paquet (TTL 7) sur le chemin du haut, et les deux suivants (TTL 8 et 9) sur celui du bas, ce qui induit la vision de droite.

Dans [66, 5] la présence de nombreux faux liens, ainsi que divers artéfacts topologiques tels que les boucles, cycles et diamants qui sont imputables à l'équilibrage de charge sont mis en évidence. Un outil appelé `paris-traceroute` a été alors proposé pour corriger certains biais. Nous le présentons plus en détail dans la section 1.2.4.

Bien d'autres problèmes existent, par exemple certains routeurs ne répondent pas aux sondes, et certains *firewalls* filtrent l'ICMP.

1.2.4 Améliorations de l'outil de mesure

Les problèmes de mesure liés à `traceroute` ont motivé des travaux dans le but de fournir des outils de mesure plus adaptés pour la mesure de la topologie de l'internet. Nous allons présenter trois de ces outils et les améliorations spécifique qu'ils apportent.

Tcptraceroute [62]

`tcptraceroute` est une implémentation de l'algorithme `traceroute` mais utilisant des paquets TCP pour envoyer ses messages. Cet outil a pour but de réduire le nombre de paquets sans réponse (le nombre d'étoiles) observés par rapport à la mesure avec `traceroute`.

En effet , les messages que `traceroute` envoie pour découvrir un chemin vers une machine sont des paquets UDP ou ICMP ECHO, et de nombreux pare-feu ou *firewalls* filtrent ces paquets, rendant impossible l'obtention d'un chemin complet à la destination dans certains cas. Cependant, dans la majeure partie des cas, ces *firewalls* laissent passer les paquets TCP. `tcptraceroute` arrive ainsi à passer un plus grand nombre de *firewalls*.

Cet apport dans l'amélioration de l'outil de mesure permet de découvrir plus de nœuds et de liens dans la mesure de la topologie de l'internet.

Paris-traceroute [49]

`paris-traceroute` est une nouvelle implémentation du principe de `traceroute` qui traite les problèmes causés par l'équilibrage de charge (*load balancing*). L'apport principal de cet outil est de réduire les faux liens entre les nœuds et constitue un pas important dans la résolution des problèmes de biais dans la mesure.

Les routeurs qui font de l'équilibrage de charge peuvent faire suivre leur trafic à travers plusieurs sorties en utilisant une stratégie par-flot, par-paquet ou par-destination. Dans le cas du *load-balancing* par-flot l'information contenue dans l'en-tête d'un paquet définit un flot et le routeur fait suivre les paquets d'un même flot à la même interface de sortie.

`paris-traceroute` utilise un principe qui consiste à maintenir identiques les champs qui définissent le flot dans l'en-tête des paquets qui sont envoyés vers une même destination, afin d'assurer que le *load-balancer* les orientera toujours vers la même interface. `paris-traceroute` ne peut pas non plus découvrir tous les chemins dans toutes les situations mais apporte une amélioration considérable par rapport à `traceroute`.

Notons cependant qu'il souffre des mêmes problèmes que `traceroute` en ce qui concerne le *load-balancing* par paquets.

DoubleTree [20]

DoubleTree est un outil conçu dans le but de réduire la redondance dans les mesures. L'idée principale pour réduire la redondance est de commencer à sonder loin du moniteur, de fonctionner en arrière, et d'exploiter la structure en arbre des routes issues d'un seul moniteur afin de ne pas sonder plusieurs fois les mêmes liens. Cet apport est très important dans notre contexte où nous étudions la dynamique. Cette idée est reprise dans nos travaux et nous avons conçu une variante de cet outil que nous avons appelé `tracetree`, qui rend possible une mesure périodique avec une fréquence élevée (section 2.1.2).

1.2.5 Les grandes campagnes de mesure

Plusieurs projets ont été lancés pour mesurer la topologie de l'internet. Ces projets exécutent des grandes mesures qui couvrent le monde entier, développent des outils de visualisation et de mesure, etc. Nous allons présenter trois grands projets parmi les plus connus dans le domaine.

Ces projets collectent les données de l'internet : nœuds (adresses IP, routeurs, AS) et les liens entre eux dans le but de construire un graphe représentant une grande partie de l'internet.

Dimes

Dimes [59, 18] est un projet de recherche scientifique qui a pour objectif d'étudier la structure et la topologie de l'internet avec l'aide d'une communauté de volontaires. Il serait très efficace pour établir une carte de l'internet d'avoir un nombre élevé de moniteurs bien repartis à travers le monde. C'est l'idée que les concepteurs du projet ont eu pour distribuer l'effort de cartographie de l'internet en invitant les gens à participer de partout dans le monde. Plus les machines qui y participent sont localisées dans des zones diversifiées, plus la carte obtenue sera précise.

L'application Dimes exécute les mesures sur internet comme `traceroute` avec un coût en bande passante d'environ 1Kb/s, ce qui peut être supportable pour une machine hôte de faible débit.

Les mesures ont débuté le 1 septembre 2004 avec l'aide de 8 328 volontaires provenant de 115 pays de tous les continents qui ont mis à la disposition du projet environ 20 000 hôtes à travers le monde. Environ 29 404 AS ont été vus, avec 204 204 liens entre eux.

Skitter

Skitter [9] est un outil de mesure de la topologie de l'internet développé par CAIDA (*Cooperative Association for Internet Data Analysis*). À partir de 1998, ce projet a fait des mesures et développé des outils de visualisation pour collecter et analyser la topologie de l'internet. Comme tous les autre outils similaires à `traceroute`, *Skitter* envoie des paquets en variant le TTL afin de collecter les chemins entre un moniteur et une liste d'adresses de destinations. Plusieurs moniteurs sont utilisés et divisés en groupes avec différentes listes selon des objectifs spécifiques d'analyse [29]. Chaque moniteur stocke les données obtenues qui sont ensuite centralisées. Les moniteurs sont stratégiquement placés dans le monde entier et les destinations sont sondées avec une fréquence qui dépend du lieu où est localisé le moniteur. *Skitter* est un projet qui a fourni des cartes de l'internet parmi les plus grandes existantes.

Skitter n'est plus en service depuis février 2008. Un nouvel outil appelé *Archipelago* (Ark) a été déployé depuis lors en remplacement de *Skitter*.

Route Views

Route views [64] est un projet de l'Université de l'Oregon visant à obtenir une vue globale de la topologie au niveau AS à l'aide des tables de routage BGP. Au départ le projet avait pour objectif de fournir un outil aux opérateurs de services internet leur permettant de déterminer comment les autres voient leurs préfixes réseaux à des fins de maintenance ou d'amélioration de leurs réseaux.

Route views dispose actuellement de 11 moniteurs dans le monde qui enregistrent l'historique des tables de routage BGP. Le format et la fréquence des données dépendent du logiciel de routage utilisé :

- dans le format du système CISCO, les données sont collectées toutes les deux heures ;
- dans le format du système ZEBRA, les données sont collectées à différents intervalles.

C'est en février 2003 que les moniteurs ont commencé le stockage de l'historique des tables de routage BGP. Il faut noter qu'il y a eu une première période de mesure effectuée par le *National Laboratory for Applied Network Research* (NLANR) [45], qui a débuté en novembre 1997 et a pris fin en mars 2001.

Planetlab

PlanetLab [55] est une infrastructure mondiale qui soutient le développement de nouvelles applications réseau. C'est une plateforme d'ordinateurs utilisés par la communauté pour développer de

nouvelle technologies, par exemple pour :

- la cartographie de l'internet ;
- le stockage distribué ;
- les systèmes pair-à-pair ;
- les tables de hachage distribuées ;
- etc.

Le projet PlanetLab a été créé en 2003, et de nos jours comporte 1 011 nœuds répartis à travers le monde sur 475 sites, ce qui en fait une ressource très intéressante pour les applications nécessitant des machines réparties.

1.2.6 Dynamique de la topologie de l'internet

L'étude de la dynamique de l'internet est un important sujet de recherche pour plusieurs raisons. D'une part, la dynamique a une influence sur les différents protocoles. D'autre part, caractériser la dynamique peut permettre de prévoir l'évolution de l'internet, ce qui est fondamental pour sa modélisation.

Pour étudier la dynamique, on a besoin à priori d'une série d'images correspondant à différentes vues de la topologie mesurée à différents moments adéquats, afin de capturer une dynamique significative. Mais une telle fréquence de mesure pose un problème complexe car obtenir une seule vue de la topologie est déjà un problème.

Il existe cependant des travaux qui ont porté sur l'évolution de la topologie de l'internet, en général sur la caractérisation et la modélisation de sa dynamique. La majeure partie de ces travaux a concerné le niveau AS de la topologie.

Parmi les travaux qui ont étudié la dynamique de la topologie de l'internet au niveau IP ou AS [14, 23, 32, 33, 46, 48, 50, 51, 53, 59, 67], certains ont tenté de caractériser l'évolution de l'internet à long terme par des propriétés élémentaires comme par exemple la taille, le degré ou le diamètre [14, 50, 46].

L'évolution permanente de la topologie est un facteur important à prendre en compte dans la mesure car elle peut induire un biais. Dans les travaux de [46], des méthodes ont été utilisées pour déterminer avec un certain degré de confiance le changement réel dans la dynamique observée de la topologie. Cela a apporté une caractérisation plus précise de la dynamique des AS.

Dynamique au niveau AS

Beaucoup d'efforts ont été fournis ces dernières années pour comprendre la topologie de l'internet au niveau AS et son évolution. Plusieurs travaux ont porté sur l'étude des propriétés du graphe de la topologie [22, 38, 40]. D'autres travaux ont été réalisés pour élaborer des modèles de l'évolution : c'est essentiellement l'ajout et la suppression des nœuds et liens dans le graphe qui sont pris en compte dans la modélisation [4, 12, 13, 68]. Cependant une caractérisation plus précise de l'évolution pour mieux comprendre la dynamique est nécessaire pour l'élaboration de modèles plus précis ou pour évaluer la pertinence des propriétés obtenues sur des vues de la topologie. Les travaux effectués par [46] ont apporté une grande contribution sur le sujet, et la dynamique de la topologie de l'internet au niveau AS est connue avec plus de précision. Par exemple, il est montré que les AS de bord et les AS de transit n'ont pas la même dynamique. Toutes ces informations sont à prendre en compte pour une modélisation plus réaliste de la topologie.

Dynamique au niveau IP

Contrairement à la dynamique de la topologie au niveaux des AS, la dynamique de la topologie au niveau IP est plus complexe à étudier.

Cependant il y a eu des efforts pour étudier la dynamique de l'internet au niveau IP voir par exemple [66, 5], mais principalement des travaux liés au routage des paquets tels que l'équilibrage de charges (*load-balacing*), la stabilité et la convergence des routages, etc.

1.2.7 Positionnement

L'étude de la dynamique de l'internet est une question clé. Cependant, il est déjà difficile d'obtenir une carte statique. Pourtant, pour avoir la dynamique il faudrait répéter la mesure de la topologie périodiquement, à une fréquence relativement élevée. Cette situation a beaucoup pesé sur les initiatives de recherche concernant la dynamique de l'internet.

Dans cette thèse, nous introduisons une autre façon d'étudier la dynamique de la topologie : nous nous focalisons sur ce qu'une machine voit de la dynamique de la topologie, qui est en soi un objet intéressant et mérite d'être étudié. Nous l'appelons *vue ego-centrée*. Notre objectif n'est donc pas d'obtenir une carte aussi complète et précise que possible de la topologie mais de tirer le maximum possible d'informations de ce que peut voir une machine.

Comme une vue ego-centrée est un objet relativement petit, il est possible d'effectuer sa mesure en un temps relativement court et à un coût réseau relativement faible. Ceci permet de répéter la mesure, et par conséquent de mesurer sa dynamique.

Pour mesurer cette dynamique, nous avons conçu un nouvel outil que nous avons appelé `traceTree`, pour effectuer des mesures ego-centrées de la topologie de manière efficace et rapide. Des mesures *radar* consistent alors en des mesures périodiques de vues ego-centrées avec cet outil.

Nous avons mené une campagne de mesure à partir de plusieurs moniteurs répartis dans le monde, pendant un durée de deux mois environ. Les données obtenues sont mises à la disposition de la communauté. Nous montrons que ces données rendent possible une étude de la dynamique à un niveau tel que la détection d'événements est possible. Ce travail est présenté au chapitre 2.

Nous avons entamé l'analyse de la dynamique sur les données obtenues. Nos observations sont basées sur des statistiques simples telle que l'évolution du nombre d'adresses d'IP par passe de mesure. Des faits surprenants, auxquels on ne s'attendait pas, sont observés. En particulier, on observe l'apparition de façon soutenue de nouvelle adresses IP dans les vues ego-centrées. Nous avons mené plusieurs expériences pour comprendre et apporter une explication à ce phénomène. Ceci fait l'objet du chapitre 3.

Le chapitre 4 est une étude qui concerne la métrologie de l'internet. Nous nous servons des données obtenues pour évaluer l'apport d'utiliser un grand nombre de moniteurs et de destinations dans une mesure distribuée de la topologie de l'internet. Ce travail étend les travaux de [27, 28], qui présentent des simulations de mesure sur des graphes obtenues par des modèles. Avec les données réelles, nous confirmons des résultats obtenus par simulation et apportons un nouvel éclairage sur cette question.

Chapitre 2

Un radar pour l'internet

Ce chapitre présente l'idée qui sous-tend cette thèse : étudier la dynamique de la topologie de l'internet vue par une machine. Nous commencerons par présenter la notion de vue ego-centrée de la topologie d'internet et les différents moyens de mesure pour obtenir une telle vue. Nous présenterons les inconvénients de l'outil `traceroute`, qui ont conduit à la conception d'un outil dédié appelé `tracetree`. Ensuite nous présenterons ce nouvel outil en détail, de l'algorithme (en montrant comment il résout les problèmes de la mesure avec `traceroute`) à la conception (les méthodes et choix techniques qui ont permis de le mettre en œuvre). Avec l'outil `tracetree`, nous allons définir ce qu'est une mesure radar.

L'outil `tracetree` comporte beaucoup de paramètres. Dans la section 2.2, nous allons décrire ces différents paramètres puis nous analyserons l'influence de chacun sur la mesure. Nous déterminerons au fur et à mesure les paramètres adéquats pour la mesure grâce à des mesures tests.

L'étude de l'influence des paramètres a permis de définir deux modes de mesure. Nous les présenterons à la section 2.3 ainsi que la campagne de mesure qui a été effectuée. Une description des données obtenues terminera cette section.

Nous présenterons dans la section 2.4 une comparaison entre `tracetree` et `traceroute`. En utilisant les données obtenues par la mesure, nous confirmons de manière pratique que `tracetree` convient mieux à la mesure ego-centrée que `traceroute`.

Dans la section 2.5 nous allons présenter un traitement des données qui consiste à appliquer un processus de filtrage permettant de supprimer de l'information inutile de la sortie de `tracetree`. Nous allons décrire le filtre dans un premier temps puis nous montrerons par une comparaison des données avant et après le filtre que l'intégrité des données est conservée.

Nous terminerons le chapitre en présentant dans la section 2.6 une analyse préliminaire de la dynamique observée. Nous allons montrer premièrement quelques caractéristiques d'une vue ego-centrée et de son évolution puis nous allons présenter deux approches d'analyse qui ont permis de détecter des événements dans la dynamique observée.

2.1 Méthodes de mesure

Nous allons présenter dans cette section ce qu'est une vue ego-centrée de la topologie, puis nous montrerons comment la mesurer efficacement.

2.1.1 Vue ego-centrée de la topologie

Nous appelons *vue ego-centrée* de la topologie de l'internet ce qu'une machine voit de la topologie autour d'elle. Plus précisément, étant donnés une machine (qu'on appellera moniteur) à partir de laquelle on effectue une mesure et un ensemble de machines cibles (qu'on appellera destinations), la vue ego-centrée est constituée des chemins entre le moniteur et chaque destination.

Vue ego-centrée mesurée avec `traceroute`

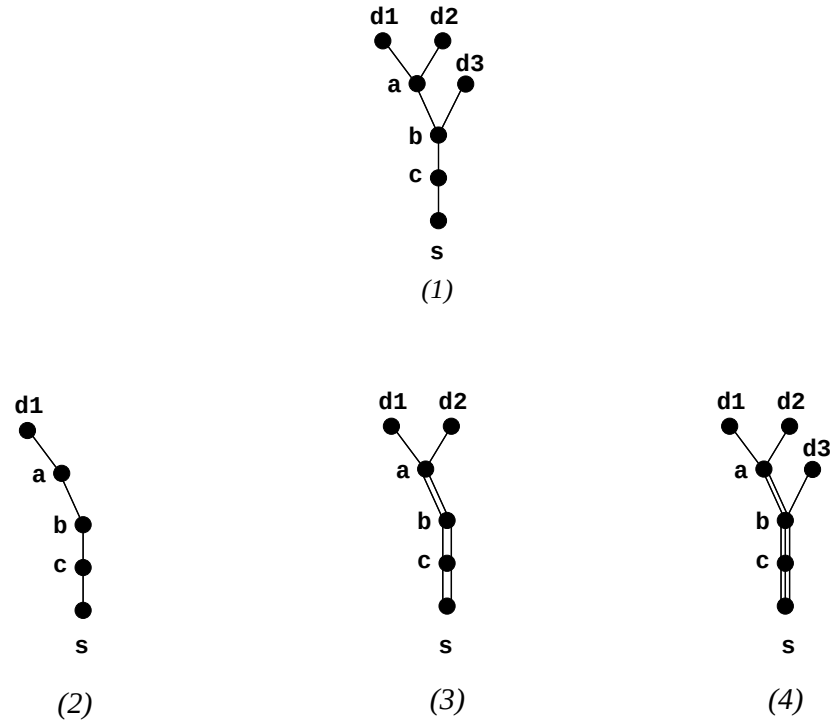


FIG. 2.1 – **Mesure ego-centrée avec l'outil `traceroute`. (1) - La topologie réelle à mesurer.** La mesure s'effectue du moniteur `s` vers les destinations `d1`, `d2` et `d3`. On collecte les routes en faisant les mesures de `s` vers `d1`, `d2` puis `d3`. Le nombre de fois qu'un lien est vu est représenté par son nombre de traits **(2) - Route vers `d1`. (3) - Route vers `d2`**. On découvre un nouveau lien (`a-d2`) et on voit des liens déjà découverts auparavant. **(4) - Route vers `d3`**. Vue ego-centrée de la topologie. Charges déséquilibrées sur les liens : ceux qui sont proches du moniteur sont vus plusieurs fois.

On peut utiliser l'outil `traceroute` pour mesurer une vue ego-centrée de la topologie. On peut en effet envisager d'effectuer la mesure en faisant une série de `traceroute` vers chaque destination, les unes après les autres. Toutefois cette méthode est très coûteuse en temps et ne convient donc pas pour bien capturer la dynamique. On souhaite en effet que la mesure d'une vue ego-centrée soit la plus rapide possible, pour pouvoir la répéter avec une grande fréquence. Une autre possibilité consiste à faire des `traceroute` simultanément vers toutes les destinations. Mais cette dernière approche induit une charge importante sur le réseau, en particulier pour les machines proches du moniteur (nous détaillerons ce point plus loin). Outre les différents inconvénients inhérents à ces approches, l'utilisation même de l'outil `traceroute` pour la mesure ego-centrée montre d'autres limites qui sont illustrées dans la figure 2.1. Il y a une importante redondance d'information dans les données obtenues : les liens proches du moniteur sont vus plusieurs fois. Ceci induit une charge inutile. De

plus, cela entraîne un manque d'homogénéité de l'information : les liens proches sont vus plus de fois que les liens éloignés, ce qui rend plus difficile une analyse rigoureuse. Enfin, bien qu'intuitivement la mesure ego-centrée doive fournir un arbre, ce n'est pas ce que l'on obtient en pratique car la vue obtenue avec `traceroute` diffère significativement d'un arbre. En effet, certains routeurs font ce qu'on appelle de l'équilibrage de charge (*load balacing*) [5, 66], et quand un tel routeur se trouve entre le moniteur et la destination, cela implique qu'il existe plusieurs chemins possibles entre eux. C'est pourquoi la vue ego-centrée avec l'outil `traceroute` ne donne pas un arbre. Notons cependant que s'il est montré qu'il peut exister plusieurs chemins entre moniteur et destination, les découvrir tous est un autre problème [6, 65], et `traceroute`ne voit pas non plus tous les chemins [66]. D'une manière générale obtenir une vue complète de la topologie, même réduite autour d'un moniteur, est un problème difficile qui en soi est un large sujet de recherche [27, 36, 19]. Nous y reviendrons plus en détail au chapitre 3.

Au final, l'approche `traceroute` pour la mesure ego-centrée présente trop d'inconvénients pour être utilisée dans notre contexte. Nous avons donc conçu un outil dédié à la mesure ego-centrée n'ayant aucun de ces inconvénients.

2.1.2 L'outil `tracetree`

Nous avons conçu un outil spécifique à la mesure ego-centrée nommé `tracetree`. Comme son nom l'évoque, il construit un arbre de chemins depuis un moniteur (la racine) vers un ensemble de destinations (les feuilles). Cet outil, que nous avons implémenté avec le langage de programmation `C`, est disponible publiquement [57].

Principe

On peut résoudre les différents problèmes soulevés par `traceroute` en faisant une mesure à l'envers partant des destinations et allant vers le moniteur comme l'illustre la figure 2.2. Étant donné un ensemble de destinations et un moniteur, pour chaque destination on découvre d'abord le dernier lien qui est sur le chemin du moniteur vers la destination (le plus proche de la destination), puis le lien précédent et ainsi de suite jusqu'à atteindre le moniteur. Quand deux ou plusieurs chemins se rencontrent sur un même nœud, on sélectionne une seule des destinations correspondantes pour continuer à sonder. On arrête donc d'envoyer des paquets vers les autres destinations car on suppose que la partie des chemins restant à découvrir est la même pour toutes ces destinations. Cette supposition n'est pas toujours vraie en pratique car il peut exister plusieurs chemins. Tous se rencontrent cependant à ce nœud, ce qui donne un sens à notre supposition : il existe dans le réseau un même chemin du moniteur vers le nœud de rencontre des destinations concernées.

On fixe un *timeout* (temps d'attente) pour la réponse à chaque paquet envoyé. Lorsque le *timeout* est expiré et qu'on n'a pas reçu une réponse, on représente le nœud correspondant par une étoile dans la vue ego-centrée. Chaque étoile est identifiée de manière unique et par conséquent on ne peut avoir deux chemins qui se rencontrent sur une étoile.

La mise en pratique de cet algorithme n'est pas évidente et soulève d'autres difficultés. Par exemple, il peut arriver qu'un même nœud apparaisse deux fois sur le chemin vers une destination, c'est-à-dire qu'on observe une boucle dans le chemin. Cela entraîne qu'on arrête de sonder vers cette destination car son chemin se rencontre lui-même. Il résulte de cette situation un graphe non connexe à la fin de la mesure.

Il y a plusieurs raisons qui sont à l'origine de ces boucles dans la mesure de la topologie. L'une des raisons est l'équilibrage de charge (*load-balancing*) [66, 5]. Un *load-balancer* est un routeur qui fait

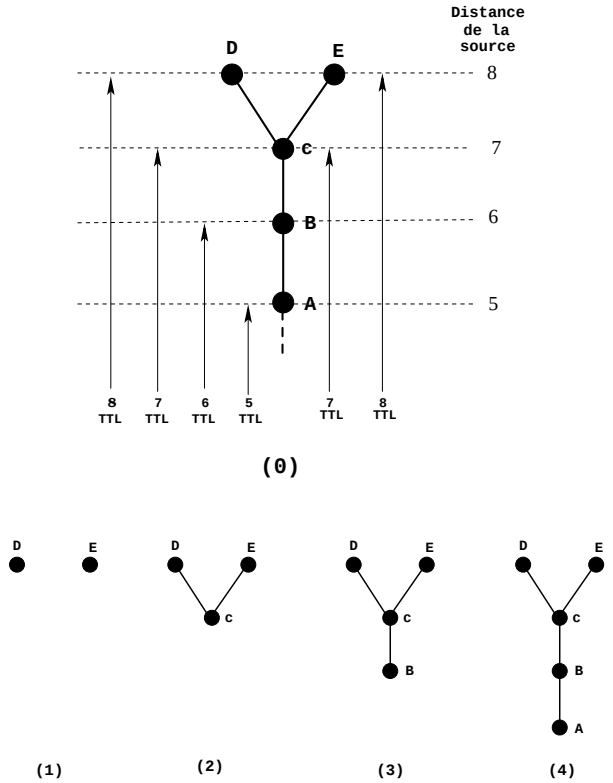


FIG. 2.2 – **Algorithme naïf de `tracetree`. (0) la topologie à mesurer** : Les nœuds `D` et `E` sont les destinations. Elles sont situées à distance 8 du moniteur. La distance de chaque nœud au moniteur est donnée par une ligne horizontale pointillée. Chaque trait vertical représente un paquet envoyé avec le TTL correspondant, vers la destination `D` à gauche ou vers la destination `E` à droite. **(1)** Un paquet de TTL 8 est envoyé vers chaque destination. On découvre deux nœuds différents : `D` et `E`. **(2)** Un paquet de TTL 7 est envoyé vers chaque destination. On découvre 2 fois le même nœud `C`. **(3)** Les deux chemins se rejoignent au nœud `C`. On sélectionne une seule destination (ici on choisit le nœud `D`) vers laquelle on envoie le paquet de TTL 6. On découvre le nœud `B`. **(4)** On envoie vers `D` un paquet de TTL 5 et on découvre le nœud `A`. Chaque lien est vu une et une seule fois.

suivre les paquets pour une même destination sur plusieurs sorties. La figure 2.3 donne un exemple de cas où le *load-balancing* faire apparaître une boucle. Le routeur e peut faire passer les paquets vers i ou vers h. Dans notre exemple nous avons les deux cas qui se produisent : il fait passer les paquets de TTL 8 et 7 vers h puis change de direction et fait passer le paquet de TTL 6 vers i. L'algorithme s'interrompt alors de façon prématurée.

Aspect algorithmique

Afin de résoudre tous ces problèmes et assurer que l'arbre soit connexe, `tracetree` utilise un second paramètre pour identifier les nœuds. Il s'agit du TTL du paquet envoyé, correspondant à la distance où se trouve le nœud. On considérera que deux nœuds sont identiques seulement s'ils ont la même adresse IP et le même TTL. En appliquant cette méthode on résout les problèmes et on obtient forcément un arbre. La figure 2.4 illustre ceci sur un exemple de topologie avec des cas problématiques. Notons que dans l'arbre, plusieurs nœuds peuvent avoir la même adresse IP. Nous avons conçu un traitement supplémentaire qui, entre autres, le ramène à un arbre dont les nœuds

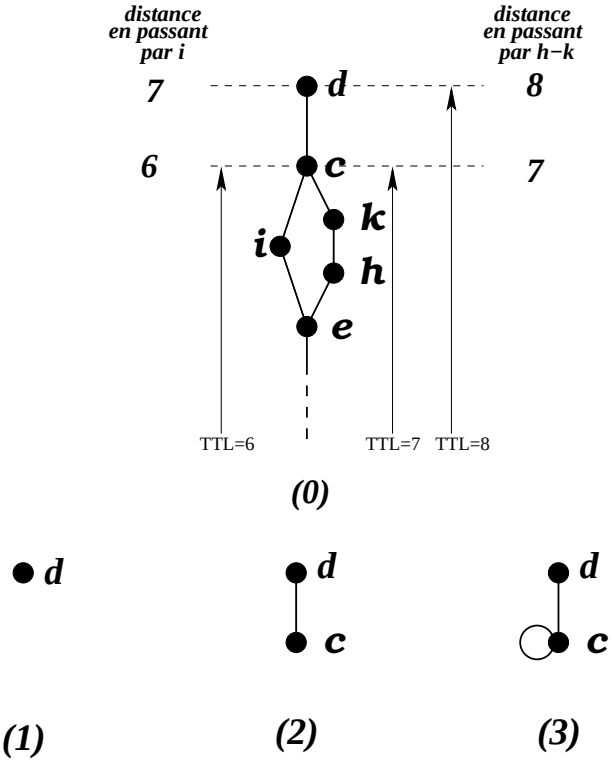


FIG. 2.3 – **Boucle causée par le *load-balancing*.** On applique la mesure `tracetree` pour découvrir la topologie (0). La destination d est à distance 8 ou à distance 7 du moniteur selon le chemin emprunté. Le nœud e fait du *load-balancing* par paquet, c'est à dire qu'il achemine les paquets vers i ou vers h. **(1)** Le moniteur envoie un paquet de TTL 8 et le nœud e l'achemine vers h pour atteindre d. On découvre le nœud d. **(2)** Le moniteur envoie le paquet suivant de TTL 7 qui est acheminé vers h et découvre c. **(3)** Le moniteur envoie un paquet de TTL 6 ; le nœud e le fait passer par i cette fois-ci et on voit le nœud c pour la deuxième fois, ce qui se traduit par un lien entre c et lui-même sur le chemin observé.

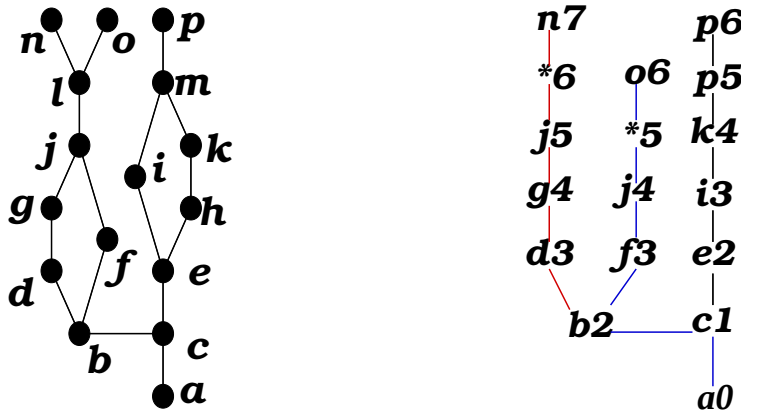
sont uniquement des adresses IP. Ce traitement est présenté dans la section 2.5.

Pour conclure, l'algorithme 1 présente les opérations effectuées par `tracetree`.

Le fait que `tracetree` construise l'arbre à partir des destinations requiert de connaître les distances auxquelles elles se trouvent. Dans la plupart des cas il n'est pas facile d'obtenir la distance exacte d'une machine sur internet, à cause de changements de routage et d'autres phénomènes [43, 66]. Pourtant, une bonne estimation de la distance est importante et renforce l'efficacité de la mesure ego-centrée. En effet une sous-estimation de la distance fera manquer les derniers liens du chemin vers la destination et à l'inverse une surestimation impliquera l'envoi de plusieurs paquets à la destination. D'une part cela constitue une charge inutile, et d'autre part la destination pourrait l'interpréter comme une attaque.

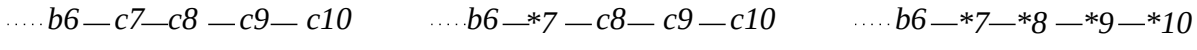
On suppose qu'on fournit une estimation de la distance de chaque destination à `tracetree` en entrée (nous expliquerons dans la section suivante pourquoi ceci est raisonnable dans notre cas). Nous avons intégré une heuristique dans `tracetree` pour gérer le cas où la destination n'est pas à cette distance.

S'il s'avère que la distance de la destination est surestimée, alors on recevra plusieurs réponses de cette même destination, correspondant à des TTL différents. `tracetree` prendra alors comme



(1) (2)

FIG. 2.4 – **Mesure `tracetree`. (1)-topologie réelle.** Le nœud a est le moniteur ; les destinations sont n, o et p. On suppose que : le nœud l ne répond pas aux sondes ; b fait du *load-balancing* par destination (c'est à dire qu'il fait suivre les paquets pour n à d et les paquets pour o à f) et e fait du *load-balancing* par paquet en faisant suivre alternativement les paquets à i ou à h. **(2)- vue ego-centrée.** Les nœud sont des couples formés d'adresse IP et de TTL avec une redondance dans les adresses IP. On obtient nécessairement un arbre connexe



(1) (2) (3)

FIG. 2.5 – **Estimation de la distance d'une destination**. La distance de la destination `c` est surestimée. On observe `c` avec plusieurs TTL. **Cas typique (1)** : de la distance 10 à la distance 7 on voit la destination `c` et juste après, à la distance 6, on observe une autre adresse, `b`. On en déduit que c est à distance 7 du moniteur. **Cas difficile (2)** : de la distance 10 à la distance 8 on voit la destination `c`, mais juste avant de voir une autre adresse on a une étoile à la distance 7. Dans ce cas on gardera 8 comme estimation de la distance de c au moniteur. **(3)** : on ne voit pas la destination `c` mais des étoiles jusqu'à la première adresse `b`. Dans ce, on gardera la valeur maximale (un paramètre de mesure, voir section 2.2) comme distance de `c` à défaut de pouvoir faire une estimation.

distance exacte de la destination le plus petit TTL des paquets pour lesquels il a reçu une réponse de cette destination. La figure 2.5 (1) montre un exemple de ce cas. On ne peut pas toujours faire cette déduction car certains cas peuvent être ambigus, par exemple le cas illustré à la figure 2.5 (2). Il y a juste après la dernière réponse de la destination une étoile. On ne sait donc pas si cette étoile est la machine juste avant la destination qui n'a pas répondu ou bien si c'est la destination qui n'a pas répondu au dernier paquet qui lui a été envoyé. On gardera néanmoins le TTL du dernier paquet à laquelle la destination a répondu comme estimation de sa distance au moniteur.

Dans le cas où la distance de la destination est sous-estimée, `tracetree` recevra au premier paquet envoyé une réponse correspondante d'une machine autre que la destination (cette machine est située sur le chemin entre le moniteur et la destination). Dans ce cas, `tracetree` augmentera

alors automatiquement le TTL du prochain paquet à la valeur maximale (un paramètre de mesure, voir section 2.2) dans le but de surestimer la distance pour retomber dans le premier cas. Il faut noter qu'il n'est pas assuré qu'avec la valeur maximale on atteigne la destination car elle peut se trouver à une distance supérieure. C'est la responsabilité de l'utilisateur de choisir la valeur adéquate et les destinations adaptées. Nous verrons en section 2.6 qu'en général la valeur 30 est suffisante.

La figure 2.5 (3) montre un exemple de cas où la destination n'est pas vue. Dans ce cas, on considère que l'étoile ne correspond pas à la destination et on augmente le TTL à la valeur maximale comme ci-dessus. Il peut arriver qu'une destination ne réponde à aucun paquet. Dans ce cas on observe des étoiles jusqu'à la première adresse IP. On ne peut alors obtenir une estimation de la distance de cette destination. On gardera la distance maximale par défaut jusqu'à ce qu'on la voie de nouveau.

Il existe plusieurs heuristiques qui ont été proposées pour estimer la distance entre deux machines sur internet [43]. Mais nous avons choisi cette heuristique car elle est la plus simple à mettre en œuvre et convient à notre cas.

Algorithm 1: `tracetree`.

Input: D ensemble de destinations, avec i_d la distance de $d \in D$.
$to\_probe \leftarrow$ file vide, $to\_receive \leftarrow \varnothing$, $seen \leftarrow \varnothing$
foreach $d \in D$ **do** ajouter (d, i_d) à to_probe
while $to\_probe \neq \varnothing$ **or** $to\_receive \neq \varnothing$ **do**
α **if** $to\_probe \neq \varnothing$ **then**
enlever (d, ttl) de to_probe et envoyer une sonde à d
ajouter $(d, ttl, current\_time())$ à $to\_receive$
// ici nécessairement $to\_receive \neq \varnothing$
β **if** *une réponse p à la sonde* (d, ttl) *est arrivée* **then**
// p envoyé par $p.moniteur$, réponse à la sonde de (d, ttl)
if $(d, ttl, \_) \in to\_receive$ **then** // else timeout
output $p.moniteur$ ttl d
if $(p.moniteur, ttl) \notin seen$ **then**
ajouter $(p.moniteur, ttl)$ à $seen$
ajouter $(d, ttl - 1)$ à to_probe **if** $ttl > 1$
for $(d, ttl, t) \in to\_receive$ **if** *timeout exceeded* **do**
enlever (d, ttl, t) de $to\_receive$
output * ttl d
ajouter $(d, ttl - 1)$ à to_probe **if** $ttl > 1$

Aspects techniques

Nous détaillons dans cette partie certains aspects techniques de `tracetree`. Comme nous l'avons décrit dans les sections précédentes, `tracetree` envoie et reçoit constamment des paquets sur le réseau et cela demande une parfaite maîtrise de deux points importants qui sont : le protocole de communication et la gestion des paquets en attente.

La version actuelle de `tracetree` fonctionne sous IPv4 et utilise le protocole ICMP, mais le programme est modulaire, afin de permettre son adaptation à d'autres protocoles (UDP et TCP).

La figure 2.6 montre le format de l'en-tête du paquet ICMP `Echo` que `tracetree` envoie aux

1 octet	1 octet	1 octet	1 octet
8 (Type)	0 (Code)	Checksum	
Identifier		Sequence Number	
Data			

FIG. 2.6 – En-tête du paquet ICMP Echo généré par `tracetree`.

destinations. Le champ *Sequence number* contient la valeur du TTL lors de l'envoi. `tracetree` traite les réponses dans deux cas de figure :

- le paquet a atteint la machine destination. Elle va renvoyer au moniteur un paquet du type ICMP `Echo-Reply` (en réponse à l'écho). Ce paquet est le même que le paquet envoyé par le moniteur : la destination va juste modifier le type du paquet en mettant 0 dans le champ correspondant de l'entête ICMP.
- le TTL du paquet envoyé a atteint zéro sans que le paquet arrive à destination. Le premier routeur qui constate que le TTL est nul arrête la course du paquet et prévient le moniteur de cela en lui envoyant un paquet ICMP `Time Exceeded` qui encapsule l'entête du paquet ICMP `Echo` reçu (ces données sont nécessaires à l'identification du paquet réponse).

Généralement le moniteur reçoit plusieurs paquets ICMP destinés à différents processus en train de s'exécuter sur la machine. Chaque processus voit en effet tous les paquets ICMP qui arrivent au moniteur. Pour reconnaître les siens, `tracetree` utilise le champ *identifier* de l'entête ICMP `Echo` dans lequel il inscrit son numéro de processus (*pid*). Le TTL du champ *Sequence number* et l'adresse de destination permettent de faire correspondre chaque réponse à un paquet envoyé.

2.1.3 Radar

Maintenant que l'outil `tracetree` est opérationnel, nous sommes en mesure de conduire nos mesures. Étant donnés un moniteur et un ensemble de destinations, une mesure radar est une répétition périodique de mesures ego-centrées. Le moniteur va donc exécuter régulièrement une mesure avec l'outil `tracetree` afin d'obtenir à chaque fois une vue ego-centrée. Bien que nous ayons tous les outils nécessaires, la mesure requiert néanmoins des paramètres adéquats. Si on considère par exemple la fréquence de la mesure (inverse du délai entre deux mesures ego-centrées consécutives), elle doit être suffisamment grande pour capturer une dynamique significative mais aussi suffisamment petite pour que la charge induite sur le réseau soit raisonnable. Il n'est pas trivial de trouver une fréquence conciliant ces deux contraintes, et on ne peut la choisir de manière arbitraire. Nous présenterons en détail dans la section suivante l'approche utilisée pour choisir des paramètres convenables.

Ainsi qu'on l'a expliqué dans la section 2.1.2, l'outil `tracetree` utilise une estimation de la distance entre le moniteur et les destinations. À cause des changements de routage et autres phénomènes sur le réseau internet la distance d'une destination peut varier avec le temps. Cependant on peut supposer que la distance du moniteur à une quelconque destination est relativement stable entre deux passes consécutives de `tracetree`. Cela signifie que l'estimation de la distance qu'on utilise dans une passe est celle observée à la passe précédente.

Pour la première passe, on affecte à chaque destination une valeur initiale par défaut. La première passe donnera de nouvelles estimations des distances aux destinations si la valeur initiale était incorrecte. Les distances des destinations sont alors mises à jour avec ces nouvelles estimations qu'on utilisera pour la passe suivante, et ainsi de suite.

2.2 Paramètres de mesure

Nous allons présenter dans cette section nos méthodes concernant les choix des paramètres de mesure. Il faut noter que les différents paramètres de l'outil `tracetree` peuvent chacun avoir un impact sur la qualité de la mesure. Cependant pour trouver des paramètres convenables il nous est impossible de tester toutes les combinaisons possibles des paramètres. Il s'agit de trouver :

- combien de destinations sonder et lesquelles ;
- quel *timeout* utiliser, c'est à dire pendant combien de temps on attend une réponse ;
- le délai entre deux mesures consécutives de `tracetree` ;
- jusqu'à quelle distance du moniteur doit-on sonder une destination ;
- la fréquence des paquets envoyés et la fréquence des paquets reçus. Différentes stratégies combinant les fréquences d'envoi et de réception sont possibles. Par exemple, envoyer 1 paquet et recevoir 1 paquet ; tout envoyer d'abord puis tout recevoir ; envoyer 1 et tout recevoir ou vice-versa...

Dans le but de trouver de bons paramètres de mesure, nous proposons une approche simple et rigoureuse pour estimer l'influence de chaque paramètre.
Elle consiste à choisir des paramètres qui semblent a priori raisonnables et prudents, que nous appellerons paramètres de base. Ces paramètres sont les suivants :

- 3 000 destinations choisies aléatoirement (tirage d'adresses aléatoires puis sélection de celles qui répondent à un `ping`) ;
- 2 secondes de *timeout* ;
- 10 minutes de délai entre passes ;
- TTL maximal de 30 ;
- 1 envoi de paquet pour 1 réception de paquet.

Ensuite nous conduisons des mesures de test en parallèle sur plusieurs moniteurs. Les moniteurs sont séparés en 2 groupes :

- les *moniteurs de contrôle* qui exécutent la mesure avec les paramètres de base pendant toute la durée de la mesure ;
- les *moniteurs de test* qui alternent des périodes de mesure avec les paramètres de base et de mesure où l'on modifie en général un seul des paramètres de base.

Une étude comparative entre les différentes phases de mesures (paramètres de base et paramètres modifiés) permet d'observer l'impact du changement et donc de comprendre l'influence du paramètre changé. Les moniteurs de contrôle permettent d'assurer que les changements observés sont bien dus aux changements des paramètres et non à un événement sur le réseau.

2.2.1 Nombre de destinations

La taille de l'ensemble de destinations est un paramètre très influent sur la mesure. La figure 2.7 illustre ceci. Comme on pouvait s'y attendre, augmenter le nombre de destinations implique une augmentation du nombre d'adresses IP observées. Toutefois notons que ce n'est pas une relation linéaire ; le nombre d'adresses IP observées n'est pas proportionnel au nombre de destinations. La structure arborescente de la vue ego-centrée explique cette disproportion. Par exemple, si l'on considère un certain nombre de destinations appartenant à un même sous-réseau, la première destination fera découvrir le chemin (ensemble d'adresses IP) séparant le moniteur de ce sous-réseau. Les autres destinations auront en général (à peu près) le même debut de chemin que la première, donc elles ne feront voir que très peu de nouvelles adresses IP. Ce sont de tels cas qui peuvent faire que l'augmen-

tation des destinations ne soit pas proportionnelle à celle des adresses IP observées. Cette remarque nous montre que ce n'est pas en augmentant systématiquement le nombre de destinations qu'on verra mieux la topologie. Mais on devra tenir compte du choix des destinations, par exemple leur emplacement sur le réseau.

Dans la figure 2.7, à partir de la mesure avec l'ensemble de 10 000 destinations, on simule ce qu'on aurait vu avec les ensembles de 3 000 et 1 000 destinations de la façon suivante : on ne garde que les chemins vers des destinations appartenant à l'ensemble de 3 000 ou 1 000 destinations (selon l'ensemble considéré). On constate que le nombre d'adresses IP de la mesure simulée est plus petit que celui de la mesure directe avec ces mêmes destinations. Le moniteur de contrôle montre que cela n'est pas dû à un changement de la topologie ou à un problème sur le réseau. Il n'y a non plus aucun effet inhérent à la simulation parce qu'on n'observe pas de différence entre la mesure directe de l'ensemble de 1 000 destinations et sa mesure simulée sur la mesure directe de 3 000 destinations. L'une des raisons possibles de cette baisse du nombre d'adresses IP vient du fait que certains routeurs limitent leur taux de réponse aux paquets ICMP, donc les surcharger en dépassant le taux limite fait qu'ils ne répondent pas et apparaissent comme des étoiles dans la mesure `tracetree`.

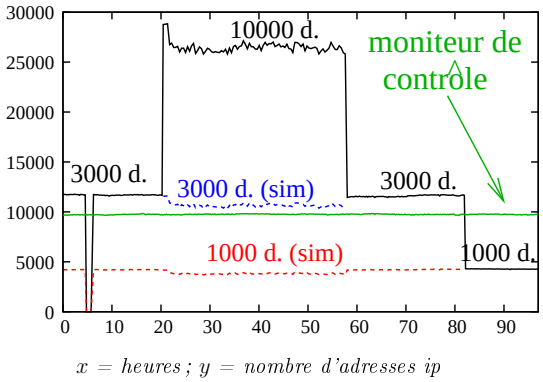


FIG. 2.7 – **Impact du nombre de destinations.** Nombre d'adresses IP vues à chaque passe en fonction du temps (en heures) écoulé depuis le début de la mesure. La courbe proche de $y = 10\,000$ correspond au moniteur de contrôle mesurant avec les paramètres de base. L'autre courbe (en trait plein) correspond à un moniteur qui commence la mesure avec les paramètres de base, avec un ensemble de destinations D de taille 3 000, change 20 heures plus tard à D' de taille 10 000 (avec l'ajout de 7 000 destinations à D) puis retourne à D après 40 heures. La mesure se termine environ 20 heures plus tard avec D'' un sous ensemble de taille 1 000 de D. Les courbes en pointillés correspondent à des simulations de ce qu'on aurait vu avec l'ensemble D durant la période de mesure utilisant l'ensemble D' et de ce qu'on aurait vu avec D'', pendant la période de mesure utilisant l'ensemble utilisant D ou D'.

Dans notre cas, nous voyons que l'augmentation du nombre de destinations peut entraîner une baisse de l'efficacité de la mesure. La simulation a montré que l'utilisation de 3 000 destinations n'implique pas de surcharge par rapport à 1 000 destinations. Cela permet finalement de conclure qu'utiliser 3 000 destinations est adapté.

2.2.2 *Timeout* des réponses

Le paramètre *timeout* limite le temps d'attente des réponses. En effet on peut ne jamais recevoir de réponse à un paquet envoyé car celui-ci ou le paquet réponse peuvent se perdre dans le réseau. De plus la machine concernée peut ne pas répondre. On doit donc arrêter d'attendre une réponse après

un délai donné.
Le problème est que si une réponse arrive, mais après le *timeout*, elle sera néanmoins rejetée. Il existe donc naturellement une relation entre le *timeout* et le nombre de réponses jetées. La figure 2.8 (droite) montre que plus on diminue le *timeout*, plus on rejette de réponses. À l'inverse, on voit sur la figure 2.8 (gauche) que plus on diminue le *timeout*, plus la durée de la passe est petite. Une bonne valeur de *timeout* est alors un compromis entre les deux : jeter peu de réponses et avoir une durée de passe relativement courte.

Nous observons que, en ce qui concerne la durée, la différence entre un *timeout* de 2 s et un *timeout* de 1 s est faible, contrairement à la différence entre un *timeout* de 4 s et un *timeout* de 2 s, qui est plus importante.

Au final, le *timeout* de 2 s est donc approprié pour obtenir un compromis : il n'augmente pas beaucoup la durée par rapport au *timeout* de 1 s et le nombre de réponses jetées est raisonnable.

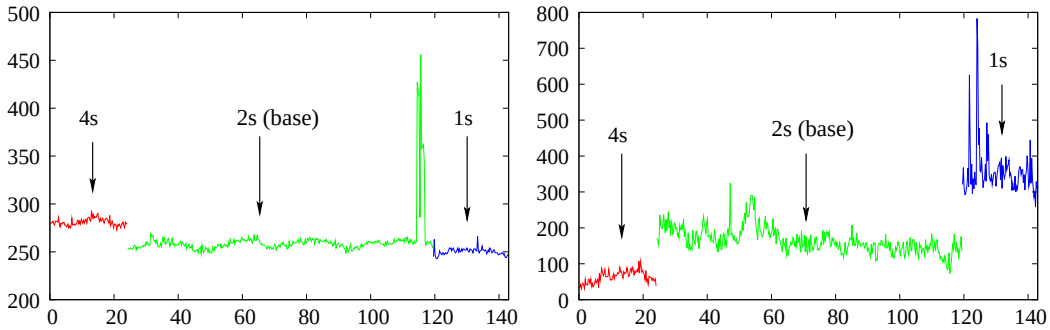


FIG. 2.8 – **Impact du *timeout***. Le moniteur commence la mesure avec un *timeout* de 4 s puis le diminue à 2 s environ 21h plus tard. Il termine avec 1 s de *timeout* après 120 heures. **Gauche** : durée de la passe de mesure en fonction du temps. **Droite** : nombre de réponses jetées par passe en fonction du temps.

2.2.3 Délai entre les passes

Une mesure radar consiste en une série de passes de mesure `tracetree`. Un bon choix du délai inter-passes n'est pas trivial car il doit être suffisamment court pour bien capturer la dynamique, mais aussi ne pas surcharger le réseau. La figure 2.9 montre que changer ce délai n'a pas d'impact significatif sur les observations. On voit que réduire le délai entre passes jusqu'à 1 minute semble être supportable par le réseau. En particulier, les variations du nombre d'adresses IP vues par passe sont similaires. On observe un important pic vers le bas dans la courbe (aux environs de 60 heures sur l'axe des abscisses) qui montre qu'un événement a eu lieu à ce moment. On constate que le moniteur de contrôle a aussi capturé cet événement car à la même période on voit aussi un pic identique. Donc le pic n'est pas dû à la réduction du *timeout*. De plus le moniteur de contrôle montre que le délai de base est adéquat, puisque le réduire ne révèle pas une dynamique significative.

2.2.4 Envoi et réception des paquets

Il ressort dans la description de `tracetree` donnée par l'algorithme 1 qu'il consiste principalement à traiter des envois (ligne α) et réceptions (ligne β) de paquets. Nous avons choisi de faire un traitement équitable entre la réception et l'envoi des paquets (1 fois α et 1 fois β à chaque tour)

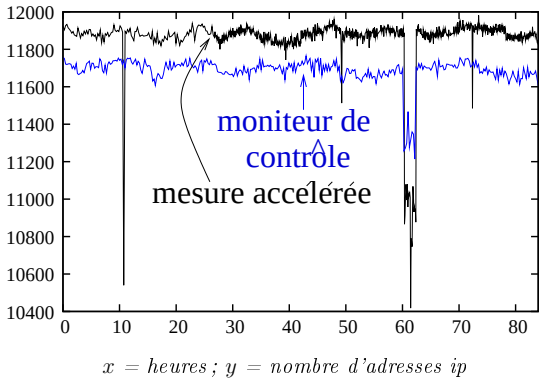


FIG. 2.9 – **Impact du délai inter-passes.** Nombre d'adresses IP vues à chaque passe en fonction du temps (en heures). La courbe du bas correspond au moniteur de contrôle avec les paramètres de base (délai entre passes de 10 minutes). La courbe du haut correspond à un moniteur dont la mesure commence avec les paramètres de base puis pour lequel après 27 heures le délai inter-passe est réduit à 1 minute.

comme paramètre de base. Il existe cependant d'autres stratégies. Nous analysons et comparons plusieurs stratégies afin de déterminer les inconvénients et avantages de chacune d'elles.

La stratégie `send_all` consiste à donner la priorité à l'envoi des paquets (remplacer le `if-then` de la ligne α par un `while-do`) : tant qu'il y a un paquet à envoyer, on l'envoie, peu importe le nombre de paquets en attente de réception. La stratégie `read_all` donne la priorité à la réception des paquets réponses (remplacer le `if-then` de la ligne β par un `while-do`) : tant qu'il a un paquet réponse, on le traite peu importe le nombre de paquets en attente d'envoi. La stratégie `send_all_read_all` réuni les deux méthodes (on a simultanément le `while-do` dans les lignes α et β). La priorité est accordée tour à tour à l'envoi et à la réception. La priorité passe de l'envoi à la réception quand la liste des paquets à envoyer devient vide et vice-versa.

La figure 2.10 (gauche) montre que le nombre d'adresses IP vues avec la mesure `send_all` est nettement inférieur à celui de la mesure avec la stratégie de base. On observe aussi une baisse, moins importante, du nombre d'adresses IP vues avec la mesure `send_all_read_all`. Comme nous l'avons expliqué dans la section 2.2.1, une baisse de ce genre du nombre d'adresses traduit une surcharge sur le réseau. La durée des passes avec ces stratégies est très courte (environ 1 minute typiquement) par rapport à celle de la mesure de base (environ 4 minutes). Mais la charge induite sur le réseau par ces stratégies entraîne une baisse de la qualité des mesures.

La mesure `read_all` est beaucoup plus proche de la mesure avec la stratégie de base que les deux autres car elle observe autant d'adresses IP que cette dernière. Malgré le fait qu'elle donne une priorité au traitement des paquets reçus, elle n'observe pas plus d'adresses IP que la mesure avec paramètres de base. Cela montre qu'avec les paramètres de base on observe en général le maximum possible d'adresses IP. La différence entre les deux stratégies se trouve au niveau de la durée de la passe : une passe de mesure avec `read_all` dure deux fois plus longtemps qu'avec la stratégie de base.

Cette analyse montre que la stratégie qui consiste à traiter en alternance la réception et l'envoi un à un présente le plus d'avantages. Elle sied mieux à nos objectifs que les autres stratégies.

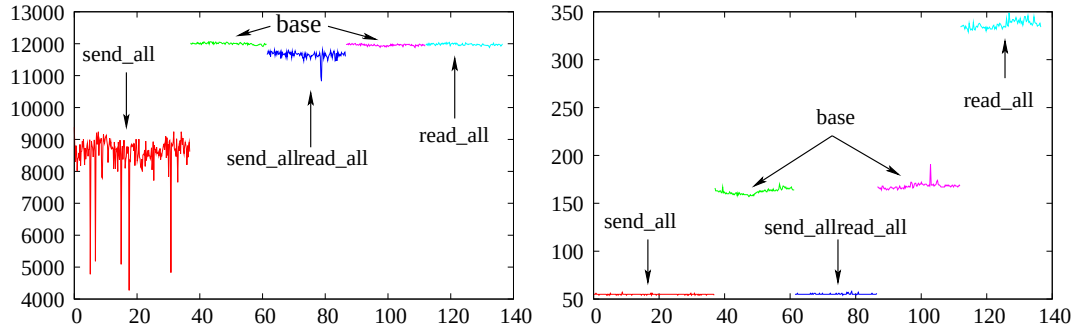


FIG. 2.10 – **Impact de la stratégie d'envoi et de réception des paquets.** On a cinq phases successives de mesure du même moniteur dont deux (la deuxième et la quatrième) avec les paramètres de base. La première phase est celle de la stratégie `send_all`. Quelques heures plus tard on effectue une phase de mesure avec les paramètres de base, avant de passer à une phase de mesure avec la stratégie `send_all_read_all`. La phase suivante est la seconde mesure avec paramètres de base, puis la dernière phase utilise la stratégie `read_all`. **Gauche** : nombre d'adresses IP vues à chaque passe en fonction du temps. **Droite** : durée de chaque passe (en seconde).

2.3 Mesures

Nous avons au final conduit plusieurs types de mesure. Les différentes mesures tests effectuées avec les paramètres de base et des paramètres modifiés ont permis de définir deux régimes de mesure pertinents : une mesure normale et une mesure rapide.

Mesure normale

L'étude de l'influence des paramètres a montré que les paramètres de base peuvent convenir pour une mesure de vue ego-centrée. Nous appelons mesure *normale* la mesure qui utilise les paramètres de base.

Mesure rapide

La mesure rapide est une mesure à haute fréquence. Pour cela le délai entre les passes est réduit. On réduit aussi le nombre de destinations et le TTL maximal pour ne pas surcharger le réseau. Les paramètres utilisés sont les suivants :

- 1 000 destinations ;
- 2 secondes de *timeout* ;
- 1 minute de délai entre les passes ;
- TTL maximal de 15 ;
- 1 envoi de paquet pour 1 réception de paquet.

2.3.1 Choix des moniteurs et destinations

Il y a différents choix possibles pour les destinations. Une des possibilités consiste à choisir les destinations de sorte que chaque classe d'adresses IP soit représentée. Le choix des destinations pourrait aussi dépendre du type de mesure qui est visé. Par exemple si l'objectif de la mesure est de

surveiller un réseau ou une zone géographique (un pays par exemple) les destinations seront choisies dans la plage d'adresses IP de ce réseau ou de cette zone (ou de façon à la traverser).

Dans notre cas les destinations ont été choisies de façon aléatoire sans aucun objectif d'appartenance à un groupe donné. Pour constituer l'ensemble de destinations nous avons donc choisi de manière aléatoire uniforme des adresses IP valides[1]. Ensuite ces adresses sont testées avec `ping`. À la fin celles qui n'ont pas répondu au `ping` sont abandonnées car certaines adresses non utilisées peuvent être à l'origine de boucles infinies (c'est à dire que les paquets qui leur sont envoyés tournent en rond) dans le réseau [70]. De plus, elles feraient augmenter le nombre d'étoiles dans la mesure.

Contrairement aux destinations, il existe des contraintes pour le choix des moniteurs. En effet il faut non seulement avoir un accès direct au moniteur (un compte) mais il faut aussi avoir certains droits d'utilisateur privilégié sur la machine pour lancer la mesure : le programme `tracetree` fait appel à des fonctionnalités que seuls les administrateurs du système peuvent utiliser.

Nous avons obtenu l'autorisation d'accéder à plus d'une centaine de machines à travers le monde. Une grande partie de ces machines proviennent de PlanetLab [55]. Les autres structures dans lesquelles nous avons obtenu des moniteurs sont :

- des universités (Europe, Afrique et Asie) ;
- des petites entreprises (France) ;
- des connexions ADSL à domicile (France).

Tous ces moniteurs sont bien répartis géographiquement dans le monde. Cependant, il y a beaucoup plus de moniteurs aux États-Unis où se trouve une grande partie de l'internet qu'en Afrique où l'infrastructure est plus réduite.

2.3.2 Données

Nous avons conduit plusieurs semaines de mesures en continu. Notons qu'il y a eu des moments d'interruption au niveau de certains moniteurs dus à des pannes locales du réseau ou directement liés au moniteur de mesure. Tous les moniteurs n'ont donc pas eu la même durée de mesure. Si la plupart des moniteurs (notamment ceux de PlanetLab) ont eu une durée de mesure de quelques semaines (environ 8), la mesure a été prolongée sur quelques moniteurs pendant plusieurs mois. Les données obtenues lors de cette campagne de mesure sont disponibles publiquement [57].

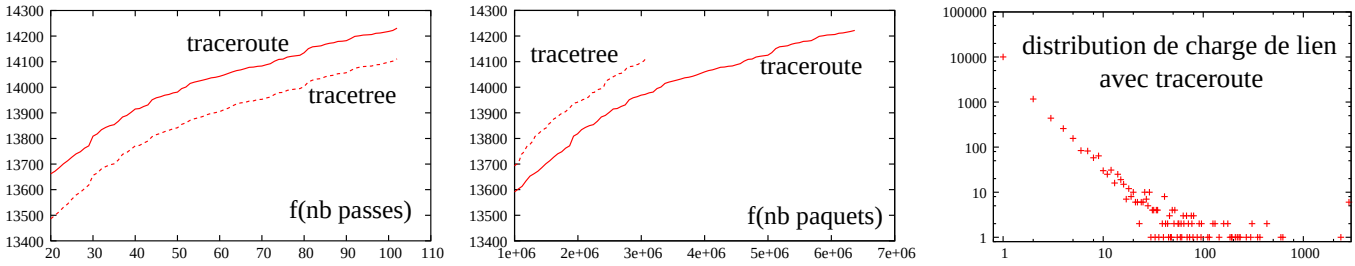
2.4 Comparaison de `tracetree` et `traceroute`

Nous présentons ici une étude comparative entre `tracetree` et `traceroute`. Comme nous l'avons déjà expliqué, les motivations qui ont conduit à la conception de `tracetree` viennent du fait que `traceroute` n'est pas efficace pour la mesure ego-centrée. Nous montrons de manière pratique la différence entre les deux outils.

Nous avons effectué une mesure à la `traceroute` en paramétrant l'outil `tracetree`. Pour cela il suffit de le modifier pour qu'il n'arrête pas de sonder vers une destination même quand son chemin en rencontre un autre. On obtient ensuite une mesure simulée de `tracetree` à partir de cette mesure en appliquant l'algorithme `tracetree` sur les données obtenues. On garantit de cette façon que les données obtenues sont bien comparables.

Afin d'évaluer leurs différences, nous comparons `traceroute` et `tracetree` selon deux aspects : l'information obtenue et la charge induite sur le réseau, voir la figure 2.11 (gauche et centre). On

[1] Dans la version IPv4, il y 2^{32} adresses possibles, desquelles il faut exclure toutes les classes d'adresses privées et réservées [56].



x = # passes ; y = # d'adresses ip *x = # paquets ; y = # IP* *x = # nb de fois vu ; y = # liens*

FIG. 2.11 – **Comparaison entre `traceroute` et `tracetree`. Gauche et centre :** nombre d'adresses IP distinctes vues depuis le début de la mesure avec `traceroute` (lignes continues) et une simulation de mesure `tracetree` (lignes pointillés) à partir de la mesure `traceroute`. Gauche : en fonction du nombre de passes. Centre : en fonction du nombre de paquets envoyés. Pour améliorer la lecture, nous ne présentons pas la partie des courbes correspondant aux 20 premières passes et aux 10^6 premiers paquets respectivement. **Droite** : distribution de la charge des liens d'une passe typique de mesure ego-centrée avec `traceroute`. À chaque valeur x sur l'abscisse, on fait correspondre le nombre de liens (y sur l'ordonnée) vus exactement x fois durant la mesure.

remarque que la courbe du nombre d'adresses IP vues en fonction du nombre de passes de `traceroute` est au-dessus de celle de `tracetree`. Ceci veut dire qu'une passe de `traceroute` recueille un peu plus d'information qu'une passe de `tracetree` (moins de 1%, dans ce cas-ci). Il est toutefois plus intéressant de les comparer en termes du nombre de paquets envoyés. Cela reflète la charge sur le réseau et donc la possibilité d'accroître la fréquence de mesure (réduire le temps entre deux passes). Les courbes montrent que de ce point de vue, `tracetree` est beaucoup plus efficace que `traceroute`. Dans notre cas, si nous considérons la barre des 14 100 adresses IP, `tracetree` l'atteint avec 3 millions de paquets environ tandis que `traceroute` a besoin de 4, 5 millions de paquets pour y arriver. À information égale, `traceroute` charge donc beaucoup plus le réseau, avec 50% de paquets de plus que `tracetree` dans ce cas, qui est représentatif.

Il faut de plus noter que la mesure ego-centrée avec `traceroute` génère une charge déséquilibrée sur le réseau, comme le montre la figure 2.11 (droite). Les liens ne subissent pas la même charge : certains sont sondés beaucoup plus de fois que d'autres. Ce phénomène de déséquilibre n'existe pas du tout avec la mesure `tracetree` : la charge est uniformément répartie sur tous les liens du réseau. En effet, `tracetree` n'envoie qu'un paquet par lien à découvrir, ce qui est optimal.

Au final, outre qu'elle a l'avantage de fournir une vue ego-centrée en arbre et homogène, la mesure `tracetree` se montre plus efficace que celle de `traceroute` en termes de nombre de paquets envoyés. Cela rend possible le fait de la répéter à fréquence élevée à un coût réseau raisonnable, pour effectuer une mesure radar.

2.5 Filtre

L'outil `tracetree` décrit en section 2.1 fournit en sortie un arbre dont les nœuds ne sont pas seulement des adresses IP mais des couples (IP, TTL) correspondant à des adresses et à leur distance du moniteur. De plus, cet arbre contient certaines informations n'ayant à priori pas d'intérêt (par exemple de longues séries d'étoiles finales lorsque la destination ne répond pas). Nous avons donc choisi de convertir la sortie de `tracetree` en un arbre nettoyé dont les nœuds sont des adresses IP. Il aurait été possible d'intégrer l'étape du filtre dans `tracetree`, mais le souci de rendre l'outil

aussi simple et modulaire que possible a prévalu et nous avons donc conçu séparément le filtre et `tracetree`.

La procédure de filtrage consiste à :

- fusionner tous les nœuds de l'arbre ayant la même adresse IP et supprimer toutes les boucles (lien dont un même nœud est à la fois l'origine et l'extrémité) ; à cette étape l'arbre de couples devient un graphe dont les nœuds sont des adresses IP ou des étoiles ;
- supprimer de manière itérative toutes les feuilles qui sont des étoiles ;
- fusionner en une seule étoile toutes les étoiles qui sont successeurs d'un même nœud. Cette étape élimine un grand nombre d'étoiles susceptibles de représenter la même machine. Le nœud l sur la figure 2.12 (1) est un exemple type.
- construire un arbre de parcours en largeur dans le graphe obtenu. L'arbre obtenu contient de nouvelles feuilles qui n'en étaient pas au début, créées par le parcours en largeur.
- supprimer itérativement toute feuille qui n'est pas dans la liste des derniers nœuds rencontrés sur les chemins allant aux destinations (le dernier nœud n'est la destination elle-même que si elle est atteinte). On applique cette étape pour supprimer les feuilles créées par le parcours en largeur.

La figure 2.12 illustre ce processus.

Un point important à noter est le fait que l'arbre obtenu est un arbre possible de routage IP du moniteur vers les destinations (semblable à un arbre de broadcast). Il contient une information similaire à l'arbre sorti de `tracetree`, ce que montre la section suivante.

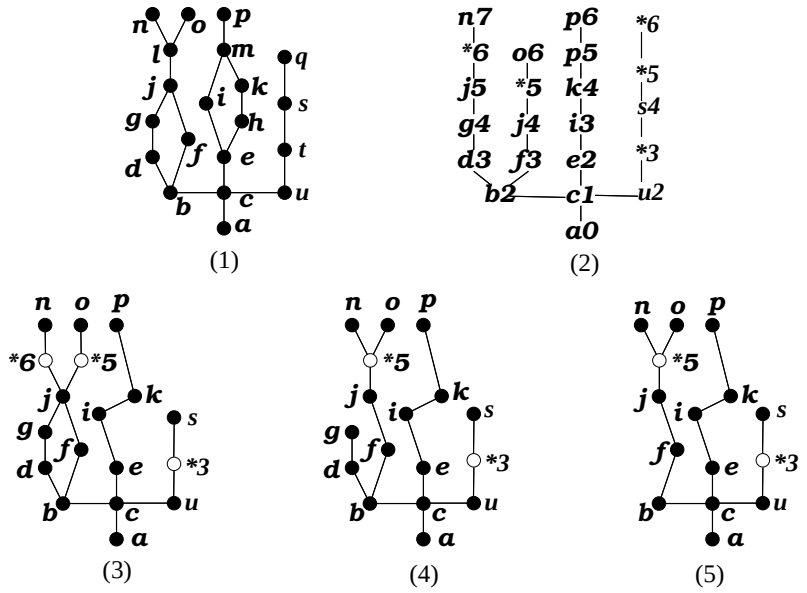


FIG. 2.12 – **Principales étapes du processus de filtrage**. **(1) -** Topologie à mesurer. **(2) -** Sortie de la mesure de `tracetree`. La destination p est vue 2 fois ; la destination q n'a pas répondu, on n'a que des étoiles à partir de s ; le nœud j est vu à différents TTL sur différents chemin, et juste après on a une étoile à chaque fois. **(3)** - Fusion des couples ayant la même adresse IP, suppression des boucles et des feuilles qui sont des étoiles. **(4) -** Fusion des étoiles ayant un même prédécesseur ; parcours en largeur. **(5) -** Suppression itérative de toute feuille qui n'est pas le dernier nœud rencontré sur un chemin vers une destination.

2.5.1 Filtrage et intégrité des données

L'étape du filtre est très importante : si elle n'est pas bien maîtrisée elle peut violer l'intégrité des données et par conséquent fausser la dynamique capturée. Nous allons montrer que le processus du filtre n'a pas d'impact notable sur les observations et donc que l'intégrité des données est conservée.

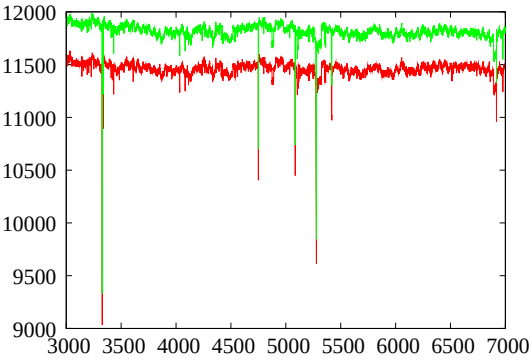


FIG. 2.13 – **Impact du filtre sur le nombre d'adresses IP.** Nombre d'adresses IP distinctes vues à chaque passe en fonction du numéro de la passe. **Haut :** données non filtrées ; **bas :** données filtrées.

La figure 2.13 montre le nombre d'adresses IP vues par passe pour les données filtrées et non filtrées. On observe une légère baisse du nombre d'adresses, de l'ordre de 3%, due à l'application du filtre. Cependant l'allure des deux courbes est quasiment identique. Cela montre que les variations du nombre d'adresses (même petites) n'ont pas été altérées significativement par le filtre. Cette remarque signifie que les dynamiques observées sur les données avant et après le filtre sont équivalentes.

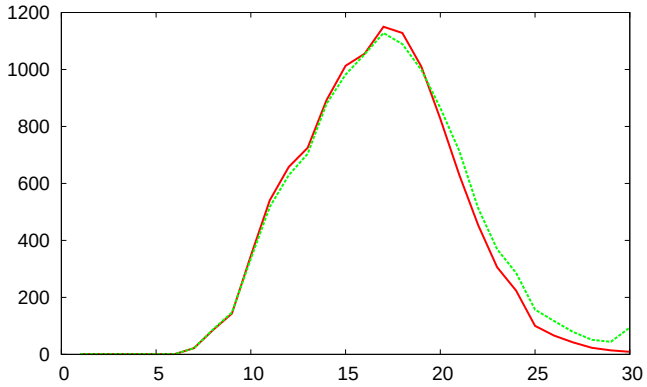


FIG. 2.14 – **Impact du filtre sur la structure de l'arbre de vue ego-centrée**. Nombre d'adresses IP observées en fonction de la distance au moniteur. Ligne pointillée : données non filtrées ; ligne continue : données filtrées.

Regardons maintenant de plus près la structure de l'arbre obtenu avant et après le filtrage. Il s'agit de vérifier que l'arbre a gardé la même structure. Pour cela, nous allons comparer le nombre d'adresses IP à chaque distance de la racine pour les données filtrées et non filtrées. La figure 2.14 montre la répartition des adresses IP vues à chaque distance du moniteur (racine de l'arbre) pour une passe donnée. Nous remarquons que le nombre d'adresses IP à distance 30 a sensiblement baissé après le filtrage. Ces adresses IP n'ont pas été supprimées de l'arbre, comme cela peut le laisser penser, mais

se trouvent à des distances inférieures. En effet, une même adresse peut être observée à différents TTL (c'est par exemple le cas de la destination p de la figure 2.12 (1) d'illustration du filtre), donc celles qui ne sont plus à distance 30 après le filtrage ont été aussi vues à des TTL inférieurs.

On n'observe aucune autre différence significative entre les deux répartitions ce qui montre que la structure générale de l'arbre est essentiellement conservée après le filtrage.

Soulignons pour terminer que nous avons présenté ici une tranche de passes de mesure d'un seul moniteur pour illustrer nos observations. Mais ce que nous avons montré est représentatif de ce que nous avons observé sur plusieurs moniteurs.

2.6 Analyses préliminaires

Cette section est dédiée à la présentation de nos premières analyses des données collectées. Nous présentons quelques propriétés statistiques qui caractérisent une vue ego-centrée, puis des méthodes qui nous ont permis de comprendre certaines caractéristiques de la dynamique de la topologie de l'internet. Sauf indication contraire, les données utilisées sont filtrées et proviennent des mesures avec les paramètres de base.

2.6.1 Caractéristiques d'une vue ego-centrée

Il s'agit ici d'établir des propriétés statistiques de base qui caractérisent la mesure ego-centrée. Dans l'absolu toutefois, on ne peut pas parler de caractérisation de la vue ego-centrée : comme on l'a déjà montré dans la section 2.2, les propriétés observées dépendent significativement des paramètres de mesure, sans parler du moniteur choisi et même du moment de la mesure. Cependant nous allons montrer que certains comportements généraux sont observables. On montrera aussi comment identifier et analyser de tels comportements et comment en distinguer de particuliers.

En général, les vues ego-centrées ont une répartition des nœuds en fonction de la distance du moniteur en forme de cloche. La figure 2.15 montre le nombre de nœuds (tous genres de nœuds confondus : étoiles, adresses IP et destinations) en fonction de la distance à laquelle ils ont été vus par le moniteur. On observe dans cette figure qu'il y a une série de 6 nœuds à traverser avant de rencontrer un nœud qui a plusieurs successeurs. On constate que la plupart des adresses IP vues sont concentrées entre les distances 15 et 20 du moniteur. Les destinations sont aussi concentrées dans ce même intervalle mais beaucoup plus vers la droite, comme on s'y attendait.

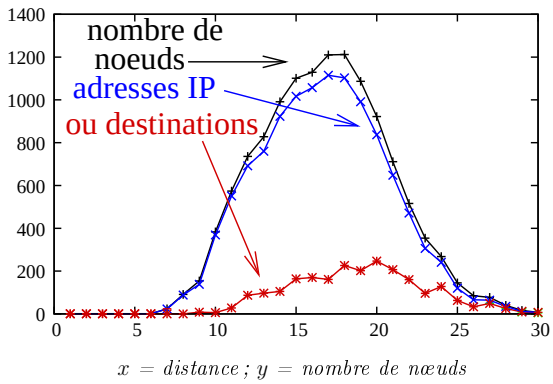


FIG. 2.15 – **Répartition du nombre de nœuds en fonction de la distance au moniteur.** De haut en bas : nombre total de nœuds, nombre d'adresses IP et nombre de destinations.

La figure 2.16 montre une représentation graphique d'une vue ego-centrée typique. On y voit que la structure des ramifications n'est pas triviale : on observe des sous-arbres de différents types. On a par exemple des formes bien équilibrées, mais aussi des structures filiformes. Les degrés sont très hétérogènes. Il faut aussi noter la faible proportion d'étoiles parmi les nœuds. La distance entre la racine de l'arbre (le moniteur) et les feuilles varie significativement (on observe une plus grande variation lorsque le TTL est plus grand). Bien que les destinations aient été choisies aléatoirement, on peut remarquer qu'un nombre significatif est proche du moniteur. En particulier on observe sur la gauche de la figure, à distance 9 du centre, un groupe d'une douzaine de destinations qui appartiennent probablement à un même sous-réseau ayant une grande plage d'adressage.

2.6.2 Évolution de la vue ego-centrée

Maintenant que nous avons une vue générale de ce qu'on observe à une passe de mesure, nous allons étudier comment évolue une vue ego-centrée avec le temps, afin de dégager des comportements généraux. Pour cela nous avons choisi de présenter pour chaque propriété statistique les données venant de trois moniteurs différents dont les périodes de mesure ne sont pas synchronisées.
Cette partie constitue un préliminaire à l'analyse de la dynamique.

La figure 2.17 montre que le nombre d'adresses IP vues à chaque passe est stable en général. Notons cependant qu'il y a quelques passes pour lesquelles le nombre d'adresses IP est très faible. On le voit par exemple dans la figure 2.17 (centre) où on a des baisses très importante atteignant parfois le zéro. Ce genre de cas indique une perte de connexion qui peut être due à une panne réseau proche du moniteur, ou à des RTT très élevés (ce qui entraine une arrivée tardive des paquets réponses et donc leur rejet) ou encore à d'autres types d'événements similaires.

Par contre, on ne constate aucune augmentation significative du nombre d'adresses IP au-delà des valeurs habituelles. La figure 2.18 confirme ces observations : la plupart des valeurs prises par le nombre d'adresses IP d'une passe de mesure sont proches d'une valeur typique. On observe quelques valeurs très petites, mais aucune valeur n'excède significativement la valeur typique. Le fait qu'il n'y ait pas de grande valeur n'est pourtant pas évident : un tel effet pourrait très bien être induit par un changement dans la vue ego-centrée.

Cependant les observations ci-dessus ne signifient pas que les adresses IP observées à chaque passe sont toujours les mêmes ou que la vue ego-centrée ne change pas. Une grande différence peut exister entre deux passes consécutives, même si elles ont presque le même nombre de nœuds. Nous étudions ceci dans la section 2.6.4.

La figure 2.19 montre qu'en général une grande partie des 3000 destinations est atteinte à chaque passe. De même que pour le nombre d'adresses IP, on observe une certaine stabilité du nombre de destinations atteintes à chaque passe. Par contre on constate que la courbe du nombre de destinations atteintes décroît légèrement au fil du temps. Les destinations (choisies aléatoirement) sont des adresses IP de différents types de machines sur internet (routeurs, ordinateurs, etc). Par conséquent le fait que certaines destinations soient des adresses dynamiques ou de simples postes de travail, temporairement connectés, peut entraîner une baisse du nombre de destinations présentes sur le réseau et donc des destinations atteintes, au fil du temps. Il faut remarquer que les courbes de la figure 2.19 ont une forme sinusoïdale plus ou moins nette selon le moniteur.

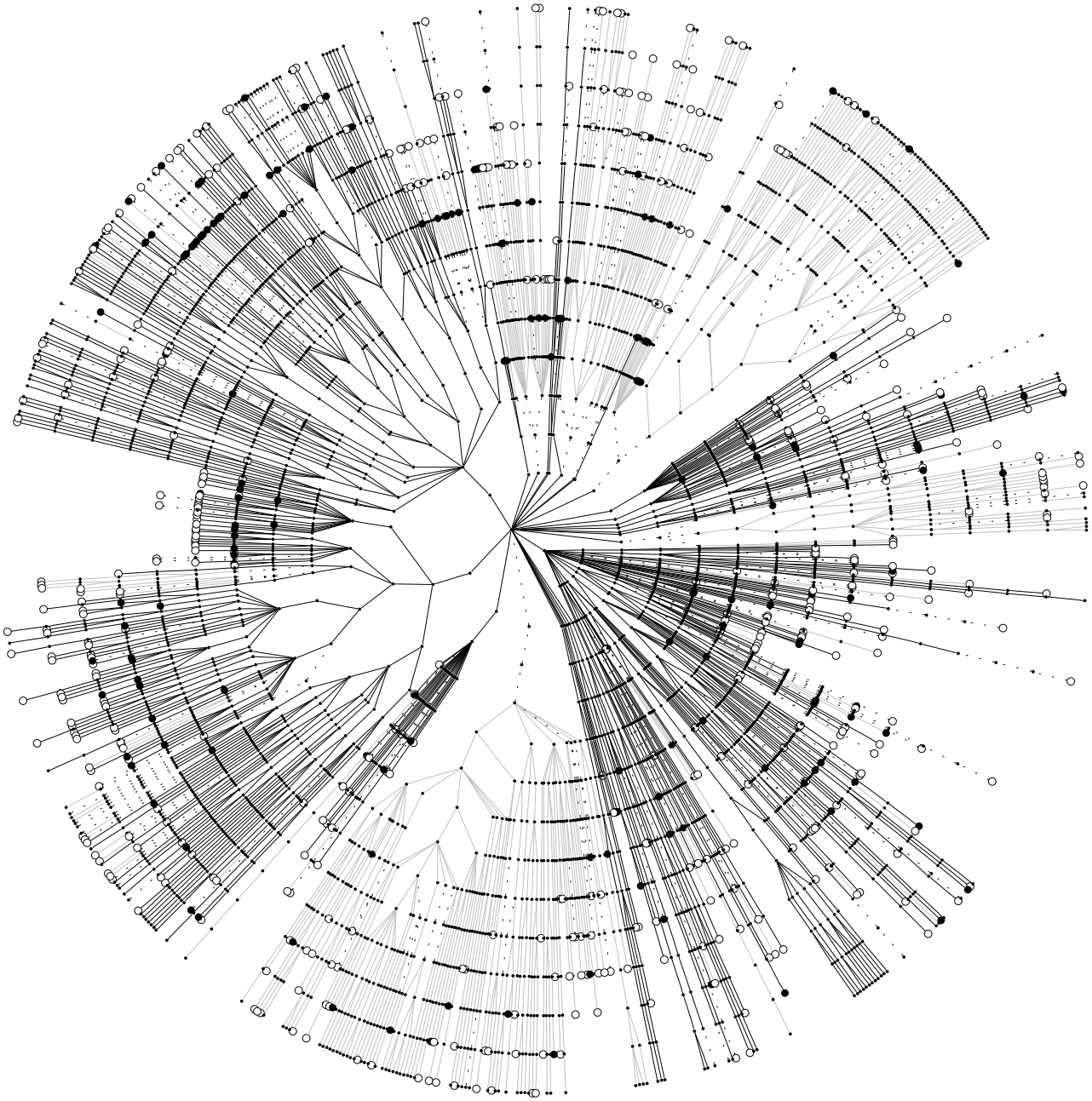


FIG. 2.16 – **Représentation graphique d'une vue ego-centrée.** Pour faciliter la lecture, on a utilisé un ensemble de 1 000 destinations seulement et le TTL maximal a été ramené à 18. Pour la même raison, nous avons enlevé les 6 premiers sauts à partir du moniteur qui n'ont aucune ramification. Le nœud qui est au centre est le premier nœud à partir du moniteur ayant une ramification. Les autres nœuds sont placés sur des cercles concentriques en fonction de leur distance au moniteur. Les liens en pointillés indiquent que l'une des extrémités au moins est une étoile. Un lien gris indique qu'il se trouve sur un chemin qui contient au moins une étoile. Les feuilles qui sont des destinations sont représentées par un cercle blanc. Toutes les autres feuilles (qui sont à distance maximale 18) sont représentées par un disque noir de taille plus petite que le cercle représentant les destinations.

2.6.3 Effet jour-nuit

Nous allons présenter ici des observations sur un effet périodique lié au jour et à la nuit, et aussi lié à la mesure ego-centrée.

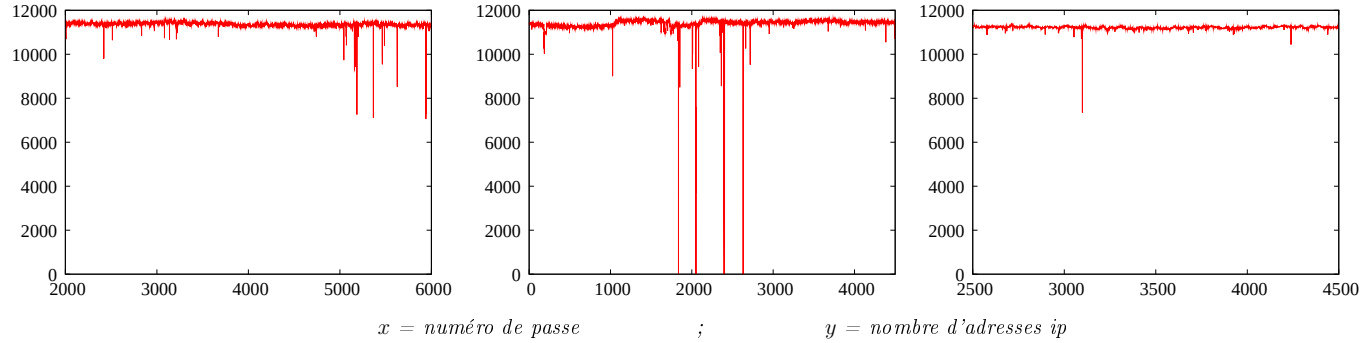


FIG. 2.17 – **Nombre d'adresses IP distinctes vues à chaque passe**. Chaque courbe représente le nombre d'adresses IP vues en fonction du numéro de passe pour un moniteur. De gauche à droite : moniteurs localisés respectivement en Allemagne, aux États-Unis et au Japon, avec des moments de mesure différents. Les deux derniers moniteurs utilisent le même ensemble de destinations, disjoint de celui du premier moniteur.

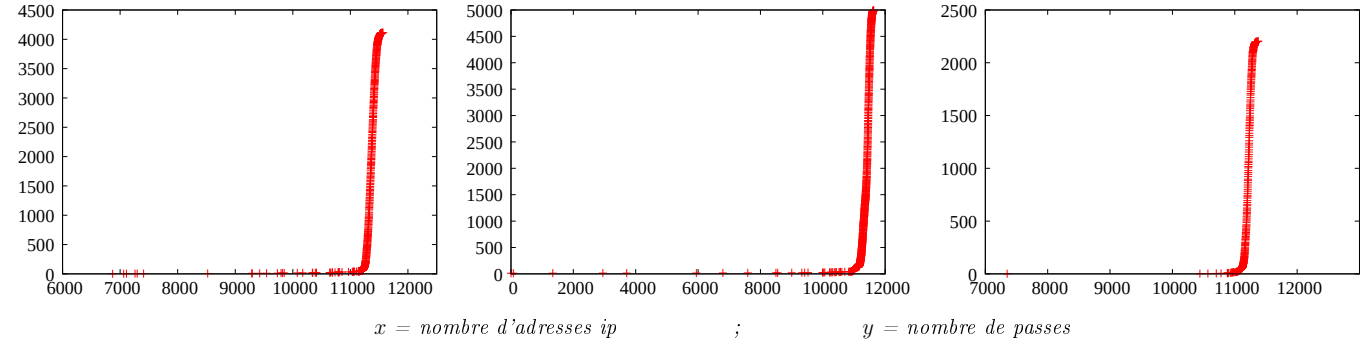


FIG. 2.18 – **Distribution cumulative du nombre d'adresses IP vues par passe.** Chaque courbe représente la distribution cumulative du nombre d'adresses vues par passe, pour les mêmes moniteurs que la figure 2.17. Pour chaque valeur x de l'abscisse, la valeur y correspondant sur l'ordonnée est le nombre de passes pendant lesquelles le nombre d'adresses IP vues est inférieur ou égal à x. La pente quasi verticale que l'on peut facilement voir sur chaque courbe signifie qu'un grand nombre d'occurrences se trouvent dans l'intervalle horizontal correspondant. Une pente horizontale signifie au contraire qu'il y a très peu d'occurrences.

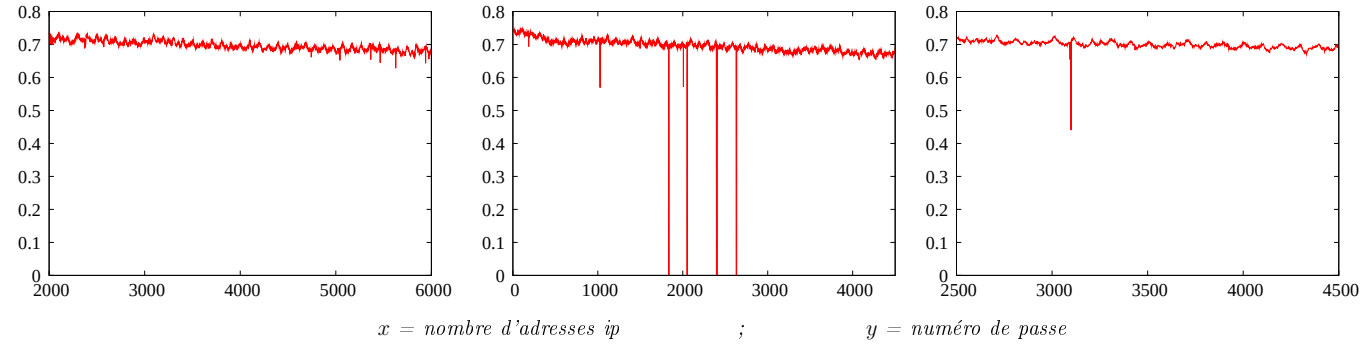


FIG. 2.19 – **Proportion du nombre de destinations atteintes par passe**. Chaque courbe représente la proportion atteinte sur les 3 000 destinations en fonction du numéro de passe, pour les mêmes moniteurs que la figure 2.17.

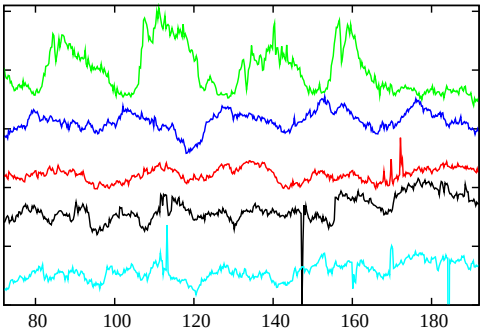


FIG. 2.20 – **Effet jour-nuit.** Nombre d'étoiles vues à chaque passe en fonction du temps (en heures). Les courbes sont décalées verticalement pour éviter qu'elles se superposent. On a en général 12 000 étoiles par passe non filtrée. Les mesures sont conduites au même moment en parallèle depuis plusieurs moniteurs répartis dans le monde. De haut en bas, les moniteurs sont localisés à Taiwan, en France, au Japon, et à l'Est et à l'Ouest des États-Unis.

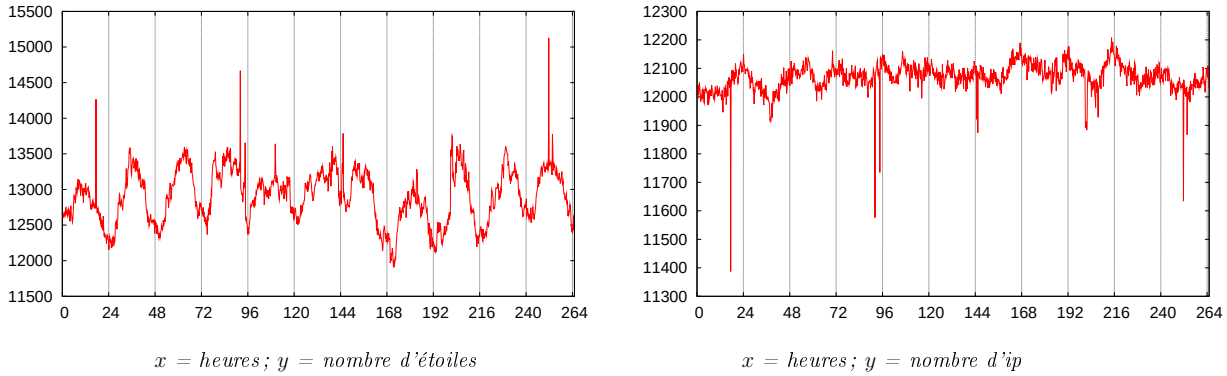
On pouvait remarquer sur la figure 2.19 que le nombre de destinations vues à chaque passe suit une certaine oscillation dont la périodicité n'apparait pas nettement. Mais on observe plus clairement sur la figure 2.20 une oscillation qui suit approximativement une période de 24 heures, soit une journée.

Ces oscillations traduisent un fonctionnement périodique de certaines machines (en tous cas de leurs réponses aux sondes ICMP) sur l'internet. À une certaine période de la journée on observe un peu plus de machines, voir par exemple figure 2.21 (droite), et cela est observable de la plupart des moniteurs. Ces observations soulèvent beaucoup de questions car les machines qu'on observe sont quasiment toutes des routeurs censés fonctionner en permanence et non périodiquement. Nous constatons au contraire que le temps auquel la mesure a été faite a une influence sur la vue qu'on obtient.

Il est important de remarquer que l'effet est local car les oscillations ne sont pas synchronisées entre les fuseaux horaires (l'abscisse de la figure 2.20 donne le temps de la période de mesure). Étant donné que la plupart des adresses IP de l'internet sont localisées aux États-Unis, la plupart de nos destinations le sont probablement aussi. On pouvait donc s'attendre à une synchronisation de l'effet jour-nuit due aux destinations (si elles arrêtent de fonctionner pendant la nuit). Le fait que les oscillations ne soient pas synchronisées montre plutôt que les destinations ne constituent pas la principale cause de l'effet jour-nuit.

Une analyse plus approfondie et focalisée sur un moniteur révèle beaucoup plus d'informations sur ces oscillations. La figure 2.21 montre que l'oscillation de la courbe du nombre d'étoiles et celle de la courbe du nombre d'adresses IP sont opposées. Pendant la période de baisse du nombre d'étoiles, on a une hausse du nombre d'adresses IP. Notons cependant que ces variations n'ont pas la même proportion : l'amplitude de l'oscillation est plus grande pour les étoiles que pour les adresses IP.

La baisse des étoiles se passe dans la période de 17h à 19h (généralement on a une baisse de l'activité humaine à cette période) ce qui traduit une baisse de trafic sur le réseau (plus exactement dans la zone où se trouve le moniteur car l'effet est local). Cette baisse du trafic pourrait entraîner le fait que certaines machines qui ne répondaient pas à nos sondes (à cause de la quantité élevée de paquets qu'elles avaient à gérer à ce moment-là) deviennent visibles, d'où une augmentation sensible (mais moins importante que la baisse des étoiles) du nombre d'adresses IP pendant la période correspondante. Il est important de noter qu'il y a quelques fois des hausses du nombre d'adresses IP qui ne correspondent à aucune baisse du nombre d'étoiles et vice-versa. Cela laisse penser qu'il y



x = heures ; y = nombre d'étoiles *x = heures ; y = nombre d'ip*

FIG. 2.21 – **Dualité nombre d'IP - nombre d'étoiles.** Variation du nombre d'étoiles et celle du nombre d'adresses IP correspondant pour un même moniteur. L'axe des abscisses (représentant le temps) est divisé en périodes de 24h (l'heure du début est 18h environ). **Gauche** : nombre d'étoiles vues à chaque passe. **Droite** : nombre d'adresses IP vues à chaque passe.

a d'autres événements et que l'oscillation qu'on observe est une résultante de plusieurs événements d'origines diverses. La section suivante est dédiée à ce point.

Les courbes qu'on présente ici proviennent des moniteurs où ces formes sinusoïdales apparaissent le plus nettement. Soulignons qu'on n'observe pas ces oscillations sur tous les moniteurs. De plus les tentatives de formaliser la détection de ces formes sinusoïdales n'ont pas apporté de résultats concluants. Les observations faites dans cette section sont donc à prendre avec prudence. Nous laissons cette direction de recherche pour des travaux futurs.

2.6.4 Dynamique et détection d'événements

On affirme souvent que dans l'internet il y a constamment des nœuds qui apparaissent et disparaissent, des liens qui sont créés et disparaissent. On sait toutefois très peu de choses sur ces changements et leurs causes. Comprendre les informations sur la dynamique de l'internet est en effet très complexe et constitue en soi un vaste sujet de recherche. Nous présentons ici quelques analyses préliminaires en ce sens, basées sur l'approche radar.

Il y a *a priori* plusieurs sortes de dynamiques sur le réseau : d'une dynamique que nous supposons normale (due à la mise en service de nouvelles adresses, à des adresses que l'on cesse d'utiliser, ou au *load-balancing* par exemple) à une dynamique anormale, liée à un événement (fait inhabituel comme une panne sur le réseau ou un changement important du routage).

La détection d'événements dans la dynamique est une question très importante pour nous éclairer sur la dynamique observée, ainsi que pour ses applications. Nous présentons ci-dessous deux approches que nous proposons pour les détecter.

Une première idée naturelle pour chercher à détecter des événements sur la topologie est d'observer la variation du nombre d'adresses IP vues à chaque passe, comme montré dans la figure 2.22 (courbe du bas).

On y distingue nettement des événements sous la forme de pics vers le bas. Cependant ces pics fournissent peu d'informations : ils peuvent signifier une perte partielle ou totale de connectivité du moniteur ou d'une machine très proche de celui-ci (par exemple une des machines se trouvant entre le moniteur et la première ayant plusieurs successeurs) mais ce n'est pas un événement majeur du point de vue global de l'internet. La vue ego-centrée est ainsi très sensible aux événements locaux, et les pics vers le bas ne permettent donc pas de détecter des événements significatifs.

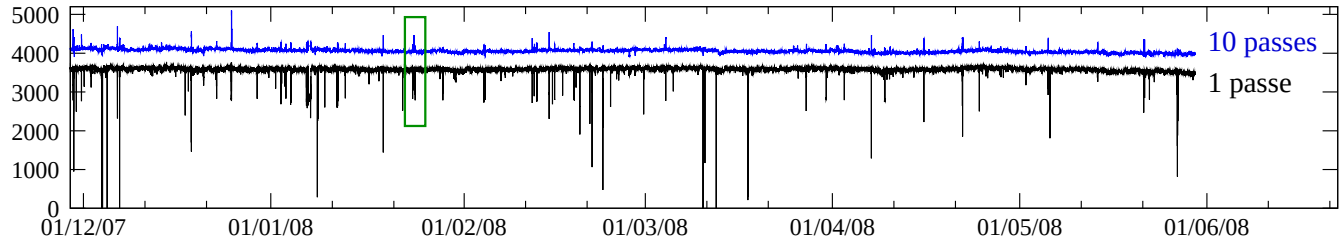


FIG. 2.22 – Bas : nombre d'adresses IP distinctes observées à chaque passe de mesure. Haut : nombre d'adresses IP distinctes observées dans 10 passes consécutives.

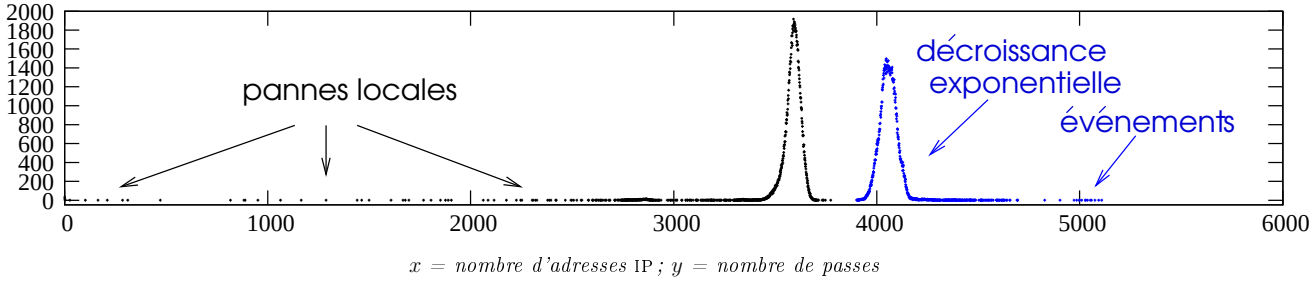


FIG. 2.23 – Distributions des valeurs des deux courbes (nombre d'IP par passe et nombre d'IP par union de 10 passes) de la figure 2.22.

Par ailleurs, on observe que la courbe du nombre d'IP par passe ne comporte pas de pic vers le haut, ce qui est confirmé par la distribution donnée en figure 2.23. De tels pics indiqueraient des événements, mais leur absence ne permet pas de conclure.

Au final, la courbe du nombre d'adresses IP vues par passe fournit très peu d'informations sur des événements significatifs sur l'internet. Ce peut être amélioré en observant le nombre d'adresses IP observées dans l'union de plusieurs passes consécutives.

Nous présentons ici les résultats obtenus pour l'union de 10 passes pour illustration. On observe que la courbe du nombre d'adresses IP distinctes observées dans l'union de 10 passes consécutives (figure 2.22 en haut) est également très stable. Mais concernant les pics, elle est très différente de celle pour une passe : il y apparait des pics vers le haut. Quand on observe la distribution de valeurs prises (figure 2.23), on constate effectivement que quelques valeurs s'écartent significativement de l'ensemble. Ces valeurs sont bien statistiquement significatives.

Ces pics vers le haut reflètent un important changement du nombre d'adresses IP distinctes dans les 10 passes consécutives et traduisent ainsi un important changement de la vue ego-centrée. Bien que le nombre d'adresses IP avant et après ces événements reste approximativement le même, la vue ego-centrée a donc changé.

Nous présentons dans la figure 2.24 un dessin du graphe obtenu en faisant l'union des vues ego-centrées avant et après un tel événement. Il illustre bien le changement détecté. On observe clairement un nombre important de nouveaux liens et nœuds qui ont fait leur apparition au moment de l'événement, et sont localisés dans une même zone. Il s'agit donc bien d'un événement qui s'est

produit dans une région particulière de l'internet.

Ces premiers résultats montrent que des nœuds apparaissent parfois subitement et de façon statistiquement anormale. Ceci nous amène à considérer, pour tout i, une période de mesure de la passe i à la passe $i+j$ que nous comparerons à une période précédente de référence qui va de la passe $i-k$ à la passe i. Plus précisément nous allons étudier les adresses IP qui sont apparues pendant la période d'observation et qui n'existaient pas pendant la période de référence. Nous appellerons ces adresses IP *nouvelles adresses*. Comme nous l'avons dit plus haut, à cause de la dynamique de l'internet il est normal d'observer constamment de telles adresses à divers endroits du réseau. Mais intuitivement, on pourrait s'attendre à ce qu'il y ait des événements localisés et donc de nouvelles adresses proches les unes des autres. Cela nous amène donc à nous intéresser aux composantes connexes qui peuvent exister parmi ces adresses comme un moyen permettant d'observer ces événements.

La figure 2.25 montre un exemple de composantes connexes parmi les nouveaux nœuds et leurs voisins observées dans un cas réel. On constate que ces composantes connexes peuvent avoir une structure relativement complexe. Elles révèlent que des événements se sont produits à différents endroits sur internet.

La figure 2.26 montre qu'il y a très peu de grandes composantes connexes. La majorité est constituée de nouveaux nœuds isolés (plus de 65% des composantes connexes). Il apparait tout de même de grandes composantes : la plus grande a 17 nœuds et une quinzaine de composantes ont au moins 10 nœuds.

Le choix de la taille de la fenêtre d'observation est un problème délicat. Il s'agit plus formellement de déterminer la longueur pertinente pour la période d'observation. Le choix de ce temps doit tenir compte du nombre de passes qu'il faut pour découvrir tous les nœuds d'une composante connexe. En effet un petit intervalle indique que tous les nouveaux nœuds concernés apparaissent probablement suite à un même événement réseau. Par contre, un grand intervalle montre que plusieurs événements se sont produits à proximité les uns des autres (voir la figure 2.25 où les deux cas se sont produits), mais on accroît le risque qu'une grande quantité d'apparitions due à la dynamique normale finissent par faire de grandes composantes qu'on prendrait pour des événements.

Conclusion

Nous avons proposé dans ce chapitre une nouvelle approche simple pour étudier la dynamique de la topologie de l'internet au niveau IP. Il s'agit de se concentrer sur une vision *ego-centrée* de la topologie. L'intérêt principal de cette approche est la possibilité de mesurer une vue ego-centrée en un temps relativement court, et de répéter cette mesure. On peut ainsi capturer une dynamique significative de la topologie de l'internet.

Nous avons montré que l'outil classique `traceroute` n'est pas efficace pour la mesure ego-centrée et qu'il était nécessaire de concevoir un nouvel outil. Nous avons conçu cet outil, nommé `tracetree`, et nous avons présenté en détail son principe de fonctionnement en montrant ses apports. Nous avons aussi présenté une étude comparative entre notre outil et `traceroute`, ce qui a confirmé l'efficacité de `tracetree`.

Nous avons présenté les différents paramètres de l'outil `tracetree` en montrant en quoi le choix de ces paramètres est important. Pour étudier l'influence de chaque paramètre, nous avons proposé une approche qui consiste à choisir des paramètres raisonnables afin d'éviter de considérer toutes les combinaisons. Ces paramètres raisonnables servent de paramètres de référence pour étudier l'impact dû aux variations. Cela nous a permis de définir de manière pratique deux types de mesure : une mesure normale et une mesure rapide.

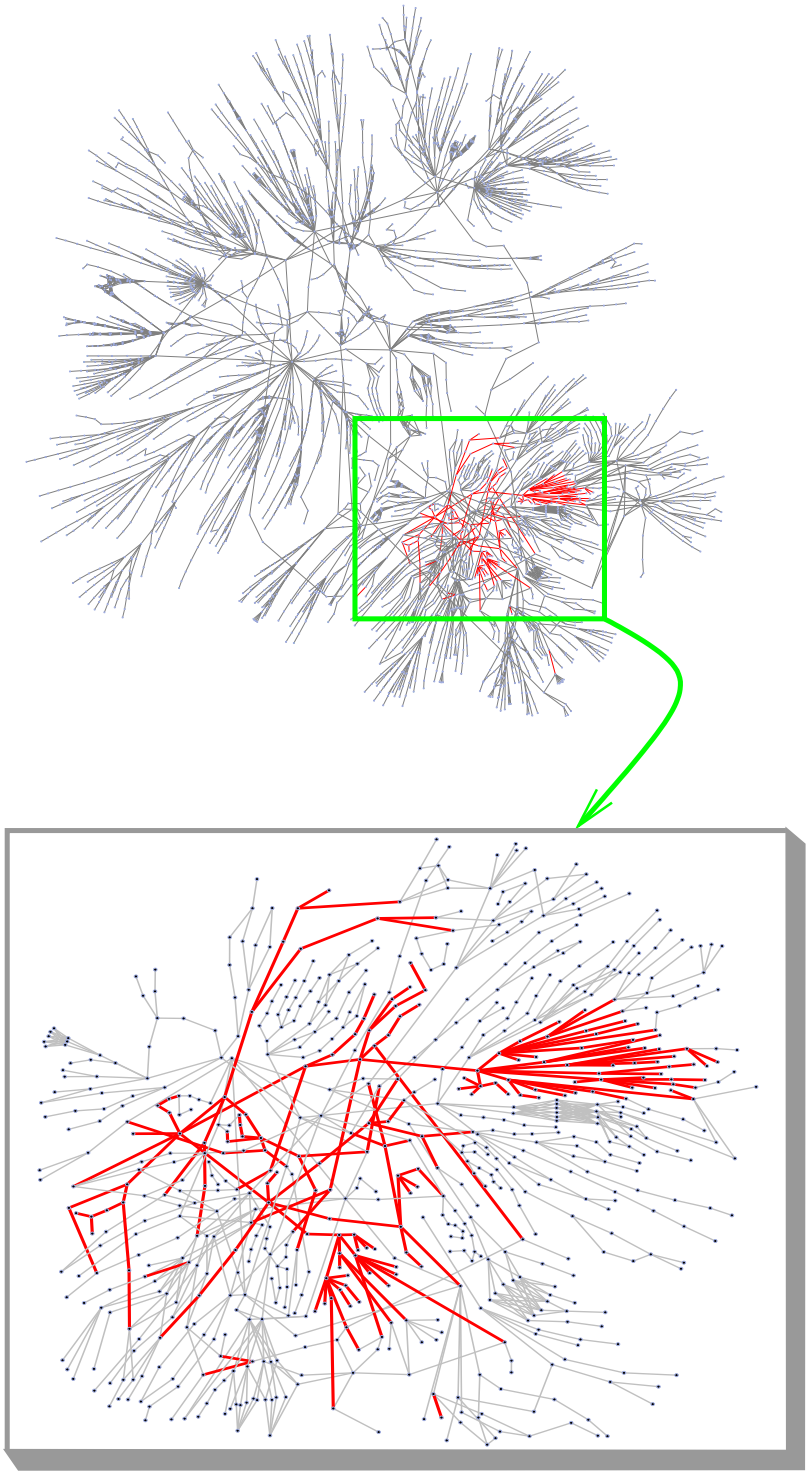


FIG. 2.24 – Événement qui a eu lieu dans le mois de janvier 2008 et signalé sur la figure 2.22 par un rectangle. Le graphe est obtenu en fusionnant les 100 passes avant l'événement et une passe après l'événement. Les liens épais sont les nouveaux liens apparus seulement après l'événement : ils n'existaient pas dans les 100 passes précédentes. **En haut** : le graphe complet. **En bas** : zoom sur la région où l'on observe beaucoup de nouveaux liens.

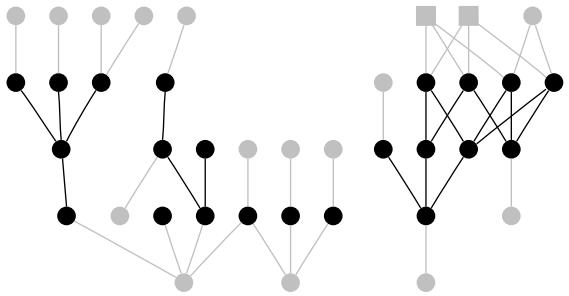
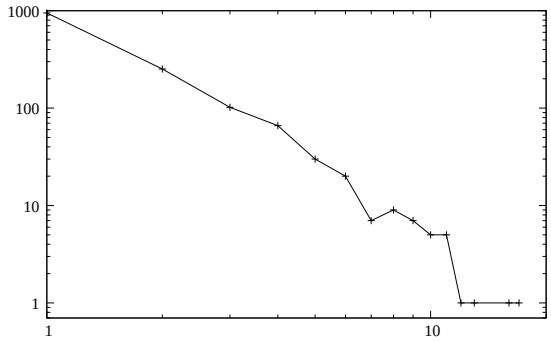


FIG. 2.25 – ***Groupes* typiques de nœuds qui apparaissent.** Chaque nœud est une adresse IP ; les nœuds noirs sont ceux apparus uniquement pendant la période d'observation, qui correspond à la deuxième moitié du temps de mesure ; les autres étaient déjà présent depuis la première moitié du temps de mesure qui correspond à la période de référence. Les nœuds de forme carrée sont vus à *toutes* les (2 200) passes, soit pendant tout le temps de mesure. Les liens sont dirigés du bas vers le haut, c'est à dire du moniteur (en bas) vers les destinations (en haut). Les 13 nouveaux nœuds situés à gauche du dessin ont tous été découverts dans un intervalle de 669 passes, par contre seulement 2 passes ont suffit pour découvrir les 9 situés à droite. Remarquons que 7 composantes connexes sont montrées dans cette figure : 4 de taille 1, 1 de taille 4, 1 de taille 5 et 1 de taille 9.



x = taille de la composante connexe ; y = nombre de composantes connexes

FIG. 2.26 – **Distribution de taille de composante des nouveaux nœuds**. Les périodes de référence et d'observation sont celles de la figure 2.25. Pour chaque taille possible x sur l'abscisse, on donne le nombre de composantes connexes de taille x.

Nous avons ensuite mené une mesure massive d'une durée de plusieurs semaines à partir d'une centaine de machines. La grande majorité des moniteurs sont des machines déployées sur la plateforme PlanetLab. Les résultats de cette mesure sont publiquement disponibles [57].

Dans la dernière section du chapitre, nous avons étudié certaines caractéristiques d'une vue ego-centrée et comment elles évoluent avec le temps. On a montré qu'en général le nombre d'adresses IP observées dans une vue ego-centrée est stable. On a montré qu'un tel fait n'était pas trivial du point de vue topologique et méritait plus d'investigation. Le chapitre suivant explore cette direction.

Un autre phénomène que nous avons présenté est l'effet jour-nuit : les variations de certaines statistiques semblent être périodiques et la période correspond à une journée. Mais cet effet n'est pas observé sur tous les moniteurs avec la même clarté, et une tentative plus formelle pour mettre en évidence cet effet n'a pas été concluante. Cela nous a conduit à conclure avec beaucoup de réserves sur l'observation de ce phénomène, ouvrant ainsi une perspective à des analyses plus approfondies.

Nous avons clos le chapitre en présentant une analyse de la dynamique visant à y détecter des

événements. Deux méthodes ont été exposées, et nous avons montré que les événements observés n'avaient pas la même importance. Certains événements sont locaux tandis que d'autres révèlent des changements importants de la topologie.

Ces premiers travaux ouvrent des perspectives prometteuses sur la détection d'événements sur l'internet.

L'approche ego-centrée pour l'analyse de la topologie de l'internet et de sa dynamique n'est pas parfaite. Notamment, un événement peut être observé avec une taille qui n'est pas proportionnelle à son importance, à cause de la vision ego-centrée.

Une perspective importante est par conséquent de considérer les vues ego-centrées d'un même événement à partir de plus d'un moniteur, c'est-à-dire un même événement observé par différents moniteurs au même moment. Ces différentes vues ego-centrées apporteront d'importantes informations sur la manière ego-centrée d'observer un événement. De plus, les pertes locales de connectivité seront éliminées. Dans la même optique, pour mieux comprendre ces questions, nous envisageons de simuler la mesure radar sur un graphe sur lequel on simule une dynamique de sa topologie (par ajouts et suppressions de nœuds et de liens).

Une autre direction intéressante est la modélisation de l'arbre fourni par la vue ego-centrée qui possède des propriétés non triviales. Une application importante serait la modélisation d'arbres de routage, qui peut être très utile pour la conception de nouveaux protocoles de routage. De plus, à partir du modèle d'arbre, on peut envisager une nouvelle approche de modélisation de la topologie de l'internet par une agrégation d'arbres.

Au delà de la détection d'événements, la caractérisation de la dynamique ouvre des perspectives intéressantes sur la modélisation de la dynamique de l'internet. Des modèles dynamiques qui simulent l'apparition et la disparition des AS et liens entre eux ont été proposés [4, 12]. Une description empirique qui caractérise l'évolution de la topologie de l'internet est nécessaire pour valider ces modèles et aussi permettre l'élaboration de nouveaux modèles plus précis.

Chapitre 3

Stabilité et instabilité des vues ego-centrées

Nous allons présenter dans ce chapitre une étude préliminaire de la dynamique des vues ego-centrées, se basant sur les données décrites dans le chapitre 2. À partir d'une vue ego-centrée de la topologie, on observe une dynamique des adresses IP très différente de ce à quoi l'on pouvait s'attendre : on voit en général un nombre d'adresses IP à peu près constant à chaque passe, mais derrière cette stabilité se cache un changement permanent des adresses IP observées. On découvre de nouvelles adresses IP à un rythme soutenu pendant toute la durée de la mesure. Ces apparitions d'adresses sont accompagnées de disparitions d'adresses IP déjà vues. L'objectif central de ce chapitre est de comprendre ces observations.

Nous présentons dans la section 3.1 différentes observations sur la dynamique des adresses IP d'une vue ego-centrée. Puis nous présentons quelques approches permettant de mieux décrire cette dynamique.

Dans la section 3.2 nous portons l'étude au niveau des systèmes autonomes (AS). Nous montrons qu'on obtient les mêmes observations qu'au niveau IP. En utilisant les archives de tables de routage BGP, nous parvenons à conclure que la dynamique de routage est une cause importante de la découverte permanente de nouveaux AS.

Cependant, cela n'explique pas complètement les observations faites au niveau IP. Dans la section 3.3, nous présentons une étude de l'incidence des changements de routage sur la dynamique observée, qui indique que la dynamique de routage constitue la principale cause de la dynamique des adresses IP que l'on observe.

Dans ce chapitre, nous allons utiliser principalement les données d'un seul moniteur pour illustrer nos observations sur la dynamique des vues ego-centrées. Nous avons fait les mêmes observations sur les données de chaque moniteur que nous avons analysé : certains détails dépendent cependant du moniteur, mais en général les observations sont qualitativement les mêmes pour tous les moniteurs.

Nous avons choisi un moniteur localisé au Japon qui a effectué une mesure radar normale (voir section 2.3) pendant deux mois (de juin à juillet 2007). Ces deux mois de mesure représentent 5 891 passes, soit approximativement 100 passes par jour, pendant lesquelles on a observé 29 100 adresses IP distinctes.

Nous avons eu recours à d'autres types de données dans certains contextes et utilisé des données provenant d'autres moniteurs. Nous signalerons explicitement les endroits où nous utiliserons d'autres données que celles décrites ci-dessus.

3.1 Dynamique des adresses IP vues par un moniteur

Dans cette section, nous allons présenter des observations générales sur la dynamique des adresses IP vues par un moniteur. Nous allons montrer que ces observations ne sont pas dues à des artéfacts de mesure. Ensuite nous étudions la stabilité des adresses IP observées dans nos mesures, c'est-à-dire à quel point on observe une adresse donnée dans un grand nombre de passes.

3.1.1 Nouvelles adresses observées par passe

Comme décrit dans le chapitre 2, une première idée pour étudier la dynamique des adresses IP observées est d'étudier le nombre d'adresses IP vues par passe, voir la figure 3.1.

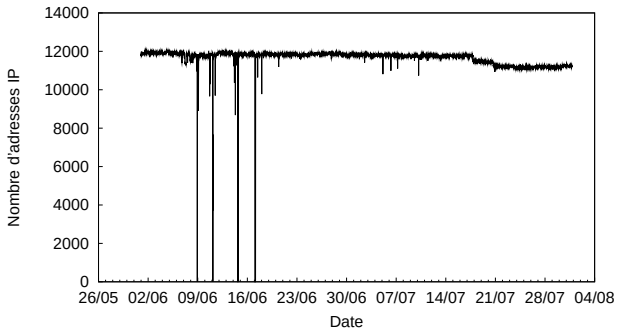


FIG. 3.1 – Nombre d'adresses IP observées à chaque passe de mesure en fonction du temps.

Nous voyons que les valeurs se stabilisent autour de 12 000 adresses, à l'exception de quelques valeurs qui s'en écartent significativement vers le bas (elles peuvent atteindre zéro) [1]. Cette figure ne nous renseigne pas beaucoup sur la dynamique. Elle peut laisser penser que chaque passe de mesure voit plus ou moins les adresses IP d'un même ensemble d'environ 12 000 adresses.

Pour exploiter la validité de cette intuition, nous présentons dans la figure 3.2 (gauche) le nombre d'adresses IP distinctes observées depuis le début de la mesure. On y voit un fait remarquable : on découvre continuellement de nouvelles adresses IP, même après une longue période de mesure [2]. On pouvait s'attendre à voir de nouvelles adresses du fait de phénomènes comme les changements de routage (qui sont des moments où l'on s'attend à découvrir beaucoup d'adresses), mais pas à un taux si élevé : pendant le dernier mois de mesure, environ 150 nouvelles adresses sont découvertes chaque jour. Pendant ce temps le nombre d'adresses IP vues par passe demeure stable. Cela semble indiquer que les nouvelles adresses qui apparaissent compensent des disparitions d'adresses déjà observées dans des passes antérieures. Pour vérifier cela, nous avons calculé le nombre d'adresses IP qu'on ne voit plus après un certain temps t, mais qu'on a observées avant ce temps t. Il est présenté dans la figure 3.2 (droite). On voit clairement que la courbe de disparition d'adresses est symétrique à celle des adresses apparues [3].

[1] On constate une baisse du nombre d'adresses IP vers la semaine du 14 au 21 juillet. On observe en pratique que le nombre d'IP par passe est stable pendant de longues périodes consécutives.

[2] Les mêmes observations ont été faites sur des mesures d'une durée de 6 mois.

[3] Les courbes de la figure 3.2 ont une forme similaire à une rotation de 180° degrés près.

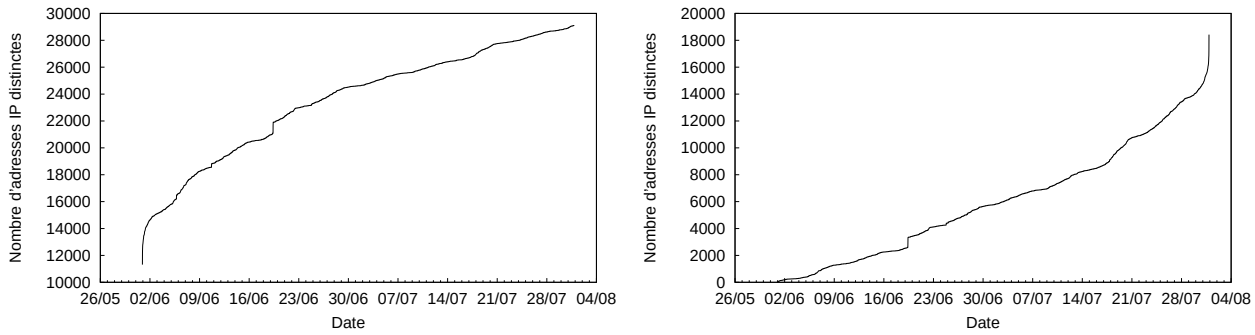


FIG. 3.2 – **Gauche** : nombre d'adresses IP observées depuis le début de la mesure en fonction du temps. **Droite** : nombre d'adresses IP en fonction du temps t, qui sont observées avant le temps t et qui ne sont plus observées après t.

Causes envisageables

La première question que l'on peut se poser est de savoir si cette étonnante observation est un artéfact de mesure.

Les destinations sont des adresses IP choisies aléatoirement parmi celles qui répondent au `ping`. Certaines de ces destinations peuvent être des adresses dynamiques, donc être allouées tour à tour à plusieurs machines. Ces machines peuvent être à différents endroits sur le réseau (et géographiquement). Quand un tel changement se produit, le changement de position de la destination dans le réseau peut faire voir de nouvelles adresses IP si l'adresse est allouée à une machine qui se trouve dans une zone qui n'a pas été vue avec les autres destinations.

Dans le but de mettre en évidence l'impact de ceci sur les nouvelles adresses IP qu'on observe, nous avons sélectionné les destinations qui sont restées stables durant toute la mesure. Cependant, déterminer exactement une destination stable n'est pas trivial, et est même très délicat. Pour cela, nous avons utilisé une heuristique très restrictive, inspirée de ce qui est fait dans le domaine de la géolocalisation [47]. Elle repose sur le fait de considérer qu'une destination est stable si le nœud immédiatement avant elle dans la vue ego-centrée (en partant du moniteur) est toujours la même adresse IP, et de plus n'est jamais une étoile. Notons que cela ne suffit pas à justifier que la destination n'est pas une adresse dynamique, mais cela prouve au moins que la destination n'a pas bougé pendant la mesure. Ceci est suffisant dans notre contexte. Seulement 35 de nos 3 000 destinations ont rempli ces conditions. Nous avons simulé une mesure restreinte à ces destinations : pour cela il suffit de supprimer les chemins vers les destinations qui ont été supprimées. La figure 3.3 montre le nombre d'adresses IP vues depuis le début de cette mesure ; même dans ce cas où une grande partie des destinations sont abandonnées (environ 98%), on observe toujours une croissance nette du nombre d'adresses IP observées. Cela montre que les adresses dynamiques ne constituent pas la cause de l'observation constante de nouvelles adresses IP.

Un autre artéfact de mesure qui pourrait être la cause de cette croissance concerne les routeurs qui utilisent plusieurs adresses IP pour répondre. On verra dans la section suivante que de tels routeurs ne sont pas la cause de nos observations.

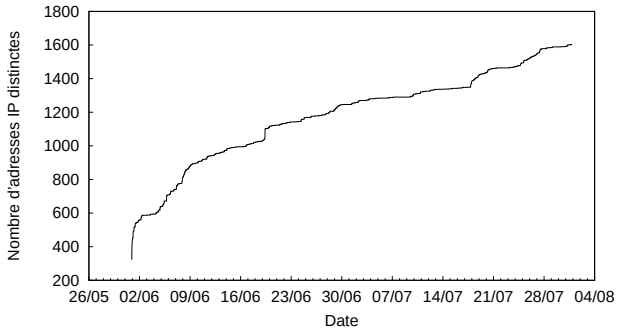


FIG. 3.3 – Nombre d'adresses IP distinctes observées dans une mesure restreinte aux destinations stables.

3.1.2 Pérennité des adresses IP

Les observations faites dans la section précédente conduisent naturellement à des questions sur la pérennité des adresses IP observées. Ces interrogations nous ont conduit à calculer diverses statistiques sur la dynamique des adresses IP observées par un moniteur, qui nous ont permis de mieux comprendre certains comportements.

Pour chaque adresse IP vue au moins une fois durant la mesure, nous nous intéressons notamment à deux grandeurs : le nombre de passes dans lesquelles l'adresse est observée et son nombre d'apparitions. On considère qu'il y a une apparition quand l'adresse est vue dans une passe mais pas dans la passe qui précède. Par exemple, une adresse IP qui est vue dans les passes 1, 5, 6, 7 et 10 a été vue dans 5 passes et est apparue 3 fois (aux passes 1, 5 et 10).

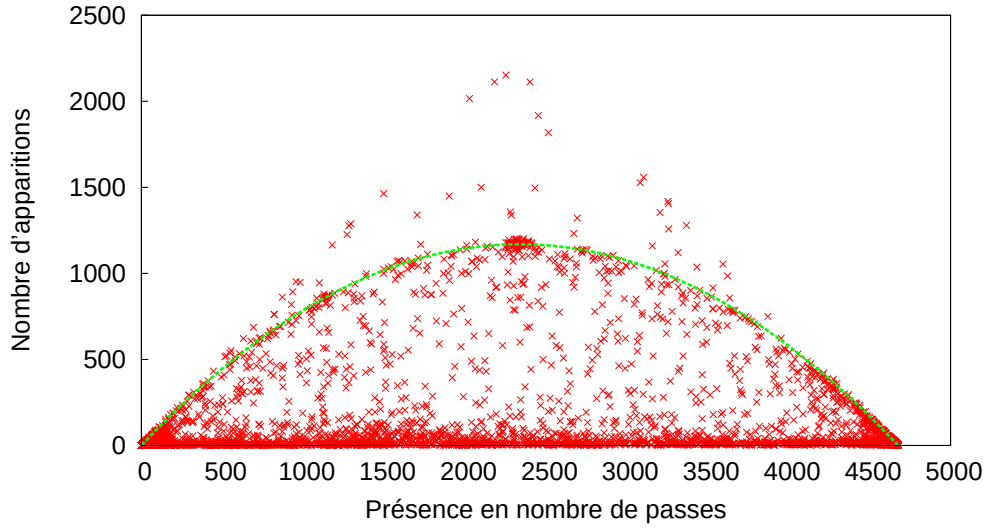


FIG. 3.4 – Nombre d'apparitions des adresses IP par rapport au nombre de passes dans lesquelles on les découvre. Chaque point correspond à une adresse IP. La coordonnée du point sur l'abscisse est le nombre de passes dans lesquelles l'adresse IP est vue par le moniteur et sa coordonnée sur l'ordonnée est son nombre d'apparitions.

La figure 3.4 montre ces valeurs (nombre de passes, nombre d'apparitions) pour toutes les adresses

IP observées par un moniteur. Il s'agit d'un moniteur [4] qui a effectué une mesure radar normale (voir section 2.3) d'une durée de 4676 passes. La courbe présente une forme géométrique claire : on peut voir un triangle dans lequel il apparaît une forme d'arc de cercle. On observe la même forme géométrique sur tous les moniteurs que nous avons étudiés. Ces formes sont surprenantes, mais peuvent cependant être expliquées.

Par définition de la courbe, aucun point ne peut apparaître en dehors du triangle : aucune adresse IP ne peut apparaître plus de fois que le nombre de passes dans lesquelles elle a été observée (par conséquent on ne peut avoir $y > x$). Inversement, aucune adresse IP ne peut apparaître un nombre de fois qui est supérieur au nombre de passes dans lesquelles elle n'est pas observée. En effet, une apparition est définie comme étant une passe où l'adresse IP n'est pas vue suivie d'une passe où elle est vue, donc on ne peut pas avoir $y > 4676 - x$, 4 676 étant le nombre total de passes de toute la mesure. Ces deux contraintes définissent les bords du triangle.

La forme en arc de cercle est en fait une parabole. Considérons une adresse IP observée dans exactement x passes distinctes durant toute la mesure. Si nous supposons que les passes où cette adresse est observée sont choisies de manière aléatoire parmi les 4676 passes de la mesure, nous pouvons alors calculer le nombre d'apparitions de cette adresse. Une passe donnée correspond à une apparition avec la probabilité que l'adresse soit vue dans cette passe, multipliée par la probabilité qu'elle ne soit pas vue dans la passe précédente, ce qui donne $(x/4676) * ((4676 - x)/4676)$. Pour obtenir le nombre d'apparitions, il suffit simplement de multiplier cette probabilité par le nombre total de passes, ce qui donne l'équation de la parabole (courbe en pointillés).

Le fait que la parabole apparaisse clairement signifie qu'un grand nombre d'adresses IP semblent avoir un comportement aléatoire dans nos observations. Les points qui sont au dessus de la parabole correspondent à des adresses IP qui ont tendance à clignoter (apparaître puis disparaître) plus vite que ce à quoi on s'attend. Enfin, la densité élevée des points au-dessous de la parabole signifie qu'un nombre important d'adresses IP ont tendance à être plus stables que la moyenne : quand elles apparaissent elles restent visibles pendant un grand nombre de passes consécutives avant de disparaître. Particulièrement au niveaux des deux bouts de la parabole, on observe une forte densité des points correspondant aux adresses IP les plus stables et les plus volatiles. Nous allons voir dans le paragraphe suivant que les adresses IP volatiles sont très nombreuses.

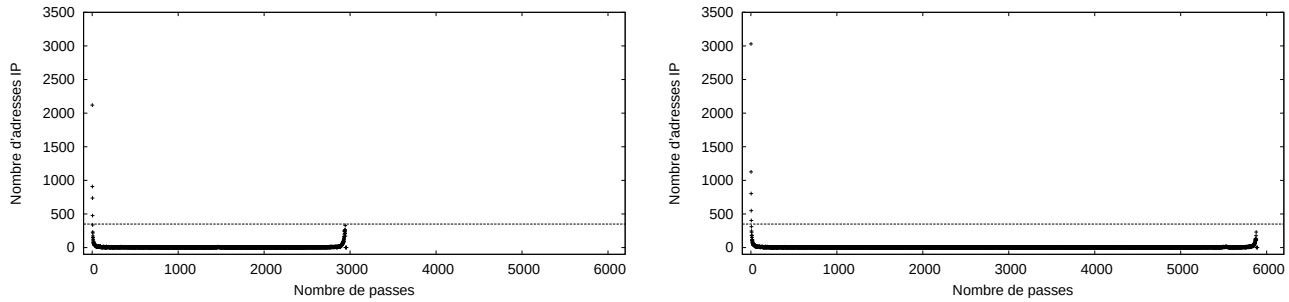


FIG. 3.5 – Distributions des nombres de passes dans lesquelles chaque adresse IP est observée. La ligne horizontale (en pointillé) correspond à la droite $y = 350$; elle facilite la comparaison du nombre d'adresses IP stables des deux courbes. **Gauche** : distribution pour les 3 000 premières passes de la mesure. **Droite** : distribution pour toute la mesure.

Nous nous intéressons ici à la distribution des nombres de passes dans lesquelles chaque adresse

[4] Le moniteur choisi est `planet1.zib.de` de la plate-forme planetLab.

IP est observée. La figure 3.5 (droite) présente pour chaque nombre de passes x le nombre d'adresses IP vues dans exactement x passes durant la mesure.

On observe qu'il y a beaucoup d'adresses IP volatiles : 3 030 IP sont observées seulement une fois durant les deux mois de mesure. Notons que les adresses IP observées une seule fois sont plus nombreuses que les adresses IP observées pour tout autre nombre de fois, voir figure 3.5 (droite). En revanche un nombre significatif d'adresses IP montrent une certaine stabilité : elles sont vues dans presque toutes les passes de mesure.

Dans le but d'étudier l'influence de la durée de mesure, on considère également la distribution correspondant aux 3 000 premières passes de la mesure pour obtenir la distribution de la figure 3.5 (gauche). On constate que la stabilité est biaisée par la durée de la mesure. Comme on peut le voir, cette distribution affiche plus d'adresses IP stables que celle sur toute la mesure. Cela signifie qu'une partie de ces adresses IP sont moins stables si la mesure a une plus longue durée. Par conséquent, la stabilité observée de certaines adresses IP sur toute la mesure est fortement liée à la durée de la mesure. Cela soulève la question de savoir comment cette distribution évolue quand la durée de la mesure continue à augmenter, voire tend vers l'infini. Cette question dépasse cependant le cadre de cette analyse.

Le nombre élevé d'adresses IP volatiles entraîne d'autres questions relatives à leur rôle dans la découverte continuelle d'adresses IP. Pour répondre à cette question, nous nous sommes intéressés à la découverte d'adresses IP stables. La figure 3.6 présente le nombre d'adresses IP distinctes observées depuis le début de la mesure, en considérant uniquement les adresses vues dans au moins 2, 10, 50, 200 ou 1 000 passes différentes. Bien que les pentes de ces courbes soient plus petites que celle de l'ensemble des adresses IP (voir figure 3.2), on continue d'observer une croissance pour tous les niveaux de stabilité des adresses. En particulier, si on considère seulement les adresses IP observées dans au moins 1 000 passes différentes (sur un total de 5 891), on observe une croissance non négligeable du nombre d'adresses IP observées. Cela montre bien que les nouvelles adresses observées ne sont pas uniquement des adresses volatiles. On ne peut donc pas imputer la croissance du nombre d'adresses IP observées uniquement aux adresses volatiles.

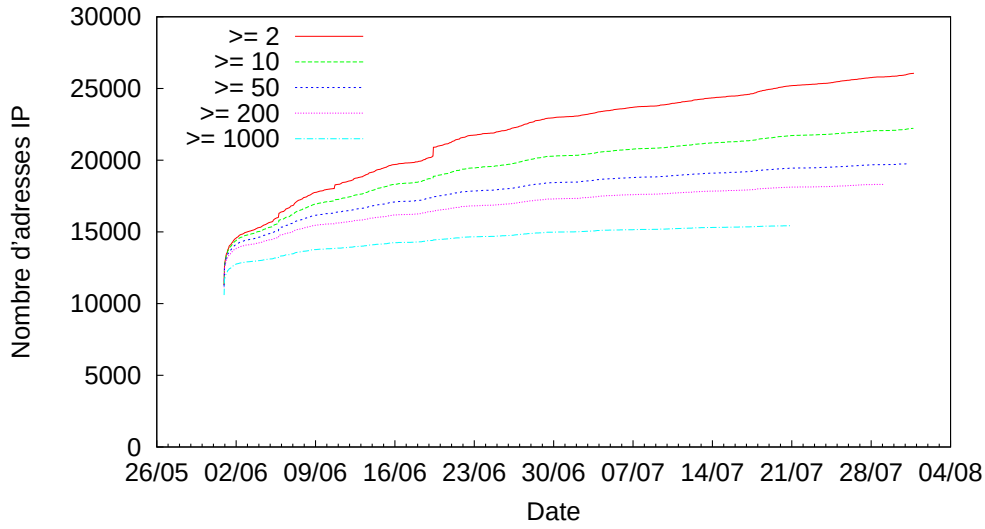


FIG. 3.6 – Nombre d'adresses IP observées depuis le début de la mesure, dans au moins 2, 10, 50, 200 ou 1 000 passes différentes (de haut en bas).

D'autre part, cela montre que les routeurs répondant avec différentes adresses ne sont pas non plus la cause du nombre croissant d'adresses IP observées. En effet, si un routeur répond avec des adresses différentes à chaque fois, celles-ci seront très volatiles : on s'attend à ne les observer qu'une fois ou un très petit nombre de fois. Comme les adresses volatiles ne sont pas la cause de nos observations, les routeurs qui répondraient avec un très grand nombre d'adresses IP différentes ne peuvent pas non plus en être la cause.

3.2 Dynamique des systèmes autonomes (AS)

Nous nous intéressons ici à ce que l'on observe au niveau des AS. Nous cherchons à savoir si l'on observe les mêmes types de comportements quand on considère les AS plutôt que les adresses IP.

Pour cela nous avons associé chaque IP observée pendant la mesure à l'AS auquel elle appartient, et nous avons étudié le nombre d'AS observés au fil du temps.

3.2.1 Méthodologie et données

Nous avons converti les données sur les adresses IP observées en des données concernant les AS. Il est possible en principe de connaître à quel AS appartient une adresse IP à l'aide du préfixe de l'adresse IP. Dans un premier temps, nous avons pour cela fait usage des services de *Team Cymru* [63]. Le site *Team Cymru* dispose d'un serveur de base de données qui fournit des informations sur l'AS correspondant à chaque adresse IP qui lui est envoyée [5].

On peut ensuite se servir de la correspondance obtenue entre adresses IP et AS pour avoir pour chaque passe de mesure radar, l'ensemble des AS observés.

Dans un second temps, nous avons également estimé le nombre d'AS qu'une mesure radar est susceptible de voir. Pour cela, nous avons utilisé les données du projet *Route Views* [64]. Ce projet fournit publiquement l'historique des tables de routage BGP de plusieurs routeurs participant au projet.

Ces données nous ont permis de simuler la mesure du point de vue AS. Cette simulation consiste à choisir un moniteur de *Route Views* situé près de notre moniteur [6], puis de sélectionner toutes les tables de routage correspondant à la période de notre mesure.

Dans chaque table de routage, nous avons récupéré l'ensemble de tous les AS que l'on peut observer sur les chemins d'AS vers les destinations : pour chaque destination, on a considéré le préfixe de son adresse IP et on a extrait dans la table de routage tous les chemins possibles d'AS pour atteindre ce préfixe. Prenons l'exemple suivant pour illustrer la méthode : la table 3.1 montre la correspondance entre les préfixes et les chemins d'AS que donne une table de routage BGP.

Si nous prenons la destination `4.23.113.42`, son préfixe `4.23.113.0/24` montre qu'elle appartient à l'AS `21889` et il y a trois chemins d'AS possibles pour l'atteindre. L'ensemble des AS que l'on peut observer sur le chemin vers cette destination est l'ensemble des AS appartenant à ces trois chemins.

[5] Pour cela, il suffit de lui envoyer un fichier contenant les adresses IP entre un `begin` et un `end` qui constitue le format de base du protocole de communication avec le serveur. On n'est pas assuré d'obtenir toujours une réponse du serveur pour une adresse IP ou bien il peut renvoyer plusieurs AS pour la même adresse. Dans ce cas, on ignore ces adresses IP.

[6] C'est la machine `Route-views.wide.outeviews.org` située dans le même AS que notre moniteur.

Préfixes	Chemins d'AS possibles
4.23.112.0/24	4777 2516 215 17132
	7500 2497 215 17132
4.23.113.0/24	4777 2516 324 21889
	7500 2497 324 21889
	2497 174 21889
4.23.114.0/24	4812 3015 174 5313
	7500 2497 174 5313

TAB. 3.1 – Illustration de la correspondance préfixe chemins d'AS d'une table BGP.

3.2.2 Observations

On observe dans la figure 3.7 que le nombre d'AS vus par passe est stable (et naturellement beaucoup plus petit que le nombre d'adresses IP). On voit approximativement 900 AS à chaque passe de mesure, sauf dans quelques cas où le nombre d'AS vus est nettement inférieur à 900 et peut même atteindre zéro (les passes où on observe une baisse du nombre d'AS correspondent à celles où on observe une baisse du nombre d'adresses IP, voir la figure 3.1). La même question se pose également ici : observe-t-on plus ou moins les mêmes 900 AS à chaque passe ?

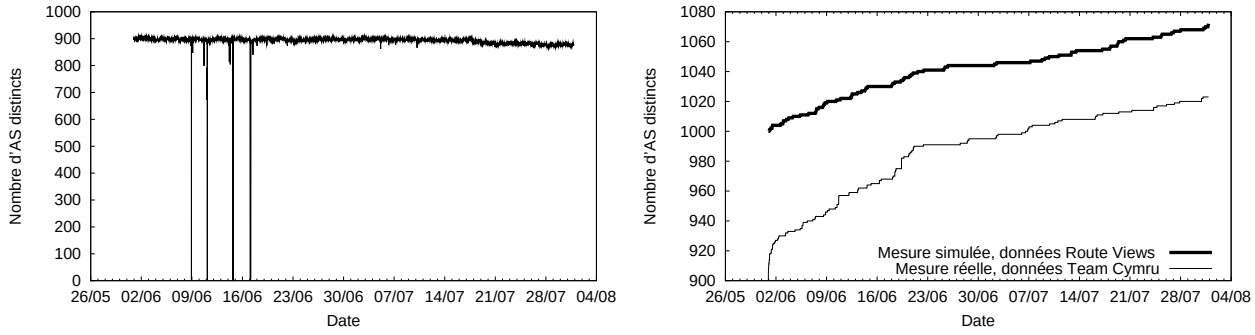


FIG. 3.7 – **Gauche** : nombre d'AS distincts observés à chaque passe de mesure en fonction du temps. **Droite** : nombre d'AS distincts observés depuis le début de la mesure : nombre réel obtenu avec les données de *Team Cymru* (courbe du bas, ligne fine), et nombre estimé avec une simulation sur les données de *Route Views* (courbe du haut, ligne épaisse).

La figure 3.7 (droite, courbe du bas) présente le nombre d'AS distincts observés depuis le début de la mesure et répond à cette question. On observe le même comportement qu'au niveau IP : bien que chaque passe voie plus ou moins un nombre constant d'AS, on découvre continuellement de nouveaux AS pendant toute la mesure.

La découverte de nouveaux AS peut-être considérée comme une explication partielle de ce qu'on observe au niveau IP : si l'on découvre un nouvel AS, on découvrira naturellement des nouvelles adresses IP appartenant à cet AS.

Pour approfondir cette question, nous nous intéressons maintenant à la taille des AS. Il s'agit ici de savoir si les AS observés sont équivalents. La figure 3.8 montre la distribution des tailles des AS observés avec *Team Cymru* : pour chaque AS vu, nous calculons le nombre d'adresses IP différentes observées dans cet AS, puis nous déterminons le nombre d'AS de chaque taille. Nous pouvons voir

que la distribution est très hétérogène : dans plus de 100 AS (sur un total de 1 023), nous observons une seule adresse IP, alors qu'il y a 3 AS contenant chacun plus de 1 000 adresses IP.

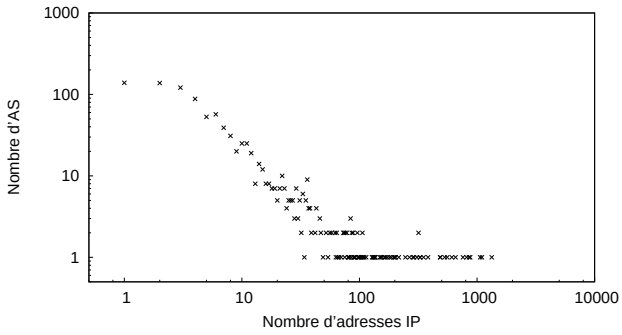


FIG. 3.8 – Distribution des tailles des AS observés.

La présence d'AS de grande taille dans la mesure nous amène à nous demander si on observe aussi une croissance soutenue des adresses IP observées dans chaque AS pris séparément. La figure 3.9 montre le nombre d'adresses IP observées depuis le début de la mesure dans le plus grand AS qu'on a observé [7]. Nous constatons encore le même type de comportement : on découvre continuellement de nouvelles adresses à l'intérieur de cet AS.

Nous pouvons dire à partir des observations faites ci-dessus que les nouvelles adresses sont observées à deux niveaux :

- on découvre des AS et par conséquent de nouvelles adresses IP ;
- dans les AS déjà vus, on découvre de nouvelles adresses IP.

On peut ainsi attribuer la croissance du nombre d'adresses observées à deux niveaux : au niveau AS et au niveau IP (à l'intérieur des AS). Cela ne donne cependant pas les raisons qui sont à l'origine de ce phénomène.

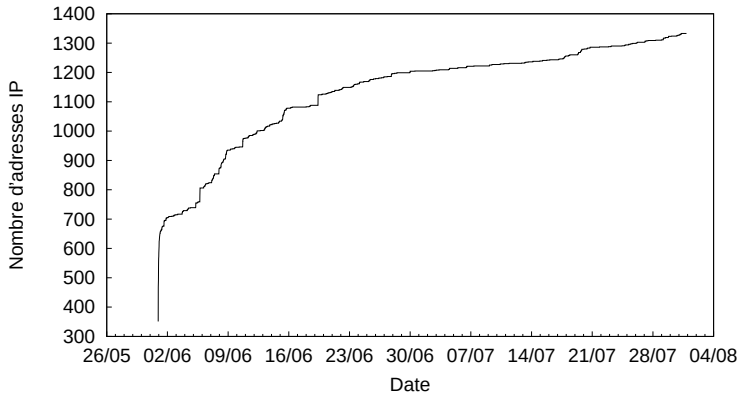


FIG. 3.9 – Nombre d'adresses IP observées dans le plus grand AS en fonction du temps.

Nous allons maintenant explorer les causes probables de ce phénomène. La question que l'on se pose est de savoir si les nouveaux AS observés sont effectivement nouveaux [8] (c'est-à-dire créés au

[7] Il s'agit de l'AS 3356 de *level* 3 *communication.*

[8] Pendant l'année 2007, environ 250 AS ont été créés chaque mois, voir http ://www.cidr-report.org/as.0/.

moment où on les découvre ou peu de temps auparavant) ou s'ils existaient avant d'être découverts et sont seulement devenus visibles à la mesure à cause de la dynamique de routage BGP. La mesure simulée sur les données de *Route Views* apporte des éclaircissements à cette question.

Comme *Team Cymru*, cette simulation montre que le nombre d'AS observables à chaque passe est stable, proche de 1 000, voir figure 3.10.

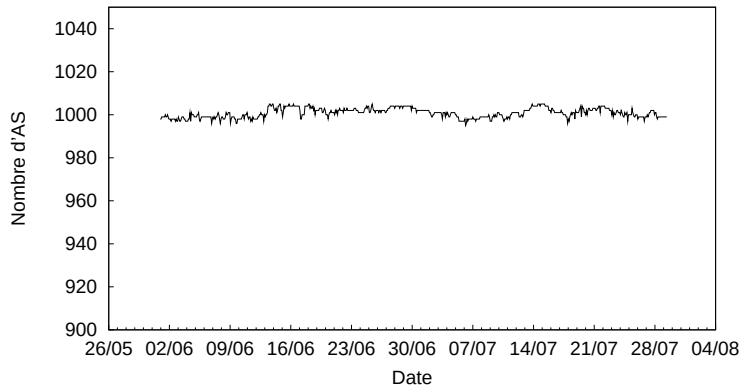


FIG. 3.10 – Nombre d'AS observés à chaque passe de mesure simulée sur les données de *Route Views*.

Nous présentons dans la figure 3.7 (droite, courbe du haut) le nombre d'AS distincts observables depuis le début de la mesure, obtenus avec la simulation sur *Route Views*. Nous constatons que cette courbe a une pente similaire à celle du nombre d'AS observés, obtenue avec les données de *Team Cymru*. Remarquons cependant que le nombre d'AS obtenu avec les données de *Route Views* est plus grand que celui obtenu avec celles de *Team Cymru*. Cela est dû au fait que les données de *Route Views* fournissent à chaque fois l'ensemble de *tous* les chemins d'AS possibles permettant d'atteindre les destinations ; les données de *Team Cymru* sont directement extraites de notre mesure, et par conséquent fournissent seulement un chemin pour chaque destination.

L'utilisation des données de *Route Views* nous permet d'aller plus loin dans la recherche des causes de la croissance du nombre d'AS observés. Nous observons 1 072 AS au total, dont 72 ont été découverts après le début de la mesure. Sur ces 72 AS observés, 70 étaient présents dans la première table de routage, mais ne faisaient partie d'aucun chemin vers nos destinations. Cela indique que les 70 AS existaient depuis le commencement de la mesure et sont devenus visibles uniquement à cause des changements de routage BGP.

Finalement, nous sommes en mesure de conclure qu'au niveau AS la croissance du nombre d'AS vus au fil du temps est principalement causée par la dynamique du routage BGP. Les AS qu'on découvre sont en effet pour la grande majorité des AS qui existaient dès le début de la mesure.

3.3 Rôle des changements de routage

Nous avons montré dans la section précédente que les nouveaux AS observés sont en réalité des AS qui existaient pour la plupart avant le commencement de la mesure. On se pose donc la même question ici, à savoir si les apparitions d'adresses IP dans nos mesures sont également causées par des changements de routage.

Sans pouvoir conclure de manière ferme, nous allons présenter des arguments importants dans ce sens.

3.3.1 Découverte d'adresses IP

Nous cherchons une estimation de la quantité d'adresses découvertes qui existaient avant le début de la mesure [9]. À la différence des AS, on ne dispose pas de base de données permettant de savoir à quels moments une adresse IP était allouée. Il est donc très difficile (voire impossible) de connaître exactement le nombre d'adresses IP créées pendant la période de mesure.

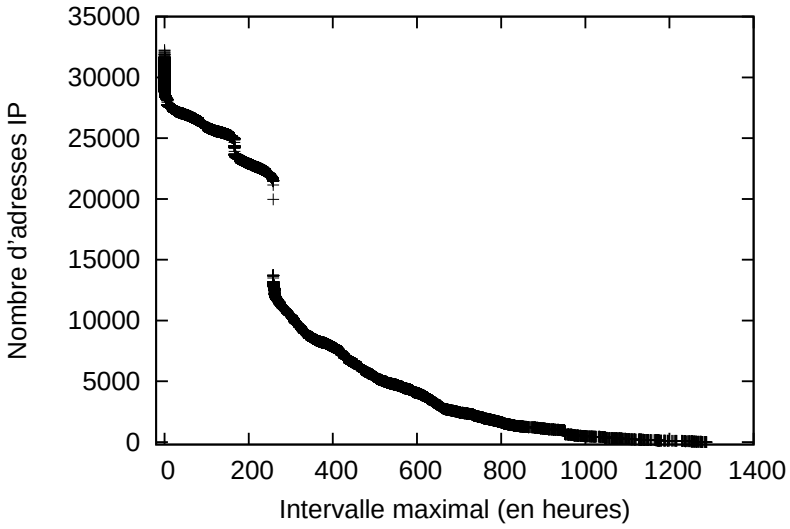


FIG. 3.11 – Distribution cumulative inverse du temps écoulé entre la première et la dernière découverte des adresses IP sur tous nos moniteurs.

Nous allons maintenant utiliser d'autres données radar que celles décrites au début de ce chapitre. Nous avons considéré les mesures provenant de plusieurs moniteurs dans l'objectif de savoir s'ils découvrent les adresses IP au même moment. Nous nous intéressons donc à l'écart qui peut exister entre le moment de la découverte d'une adresse IP par le premier moniteur et le moment où le dernier la découvre. Nous avons choisi 11 moniteurs qui ont effectué la mesure à la même période et qui utilisent le même ensemble de 3 000 destinations, dans le but de tenter de maximiser le nombre d'adresses observées par plus d'un moniteur. Nous nous intéressons uniquement aux adresses IP observées par au moins 2 moniteurs. Pour chaque adresse IP, nous notons la date à laquelle elle est découverte par chaque moniteur, puis nous calculons les intervalles entre ces moments de découverte. Par exemple une adresse qui est découverte par trois moniteurs dans la même journée, le premier à 5 heures, le deuxième à 8 heures et le dernier à 11 heures, aura un intervalle de découverte de 6 heures.

La figure 3.11 montre la distribution cumulative inverse des longueurs de ces intervalles. Nous observons qu'il y a un grand nombre d'adresses IP découvertes par un moniteur donné qui en réalité avaient été vues par d'autres moniteurs un temps significatif avant. Si nous considérons les 32 228 adresses IP vues par deux moniteurs au moins (sur un total de 40 076), 22 897 ont un écart minimum de 200 heures, c'est dire que ces adresses sont découvertes par un premier moniteur puis découvertes par un autre moniteur plus de 200 heures plus tard, ce qui montre que ces adresses existaient longtemps avant qu'on ne les découvre. Il faut noter que cela ne nous dit pas si les autres adresses existaient

[9] La question de ce que signifie exactement pour une adresse IP d'exister ou non est une question délicate, à laquelle nous ne prétendons pas répondre de manière définitive ici. En pratique nous nous intéressons au fait de savoir si ces adresses IP sont allouées ou non, c'est-à-dire si elles correspondent à des machines ou non.

avant d'être découvertes ou non. Mais ces observations montrent qu'une grande partie des adresses IP découvertes par un moniteur donné existaient bien avant que ce dernier ne les découvre, et donc que ce sont des changements de routage sur le réseau qui les rendent visibles.

3.3.2 Disparition d'adresses IP

Dans la section 3.1 nous avons montré que la découverte de nouvelles adresses IP était accompagnée d'une disparition similaire d'adresses IP (figure 3.2). Il est donc possible que les disparitions d'adresses IP aient les mêmes causes que les apparitions, et qu'expliquer les premières fournisse également une explication pour les deuxièmes.

Notre approche pour étudier cette question consiste à vérifier régulièrement l'existence de toutes les adresses IP pendant la mesure. Nous avons donc lancé une nouvelle mesure qui consiste à :

- faire une mesure radar normale avec 480 destinations (choisies aléatoirement parmi des adresses répondant au `ping`, voir section 2.3).
- faire régulièrement un `ping` vers toutes les adresses IP vues depuis le début de la mesure. Il faut noter que cet ensemble grandit avec le temps. Nous effectuons cet ensemble de `ping` toutes les 12 heures.

Les mesures ont été effectuées à partir d'un moniteur qui est situé au LIP6 (à Paris) et ont duré six semaines environ. Les données obtenues de ces mesures correspondent à 5 350 passes de mesure radar et 83 passes de test avec ping.

La figure 3.12 montre que le nombre d'adresses IP vues par passe (gauche) est stable (autour de 1 900 IP) et qu'on découvre continuellement de nouvelles adresses IP jusqu'à la fin de la mesure (droite) comme on l'a observé pour les autres moniteurs.

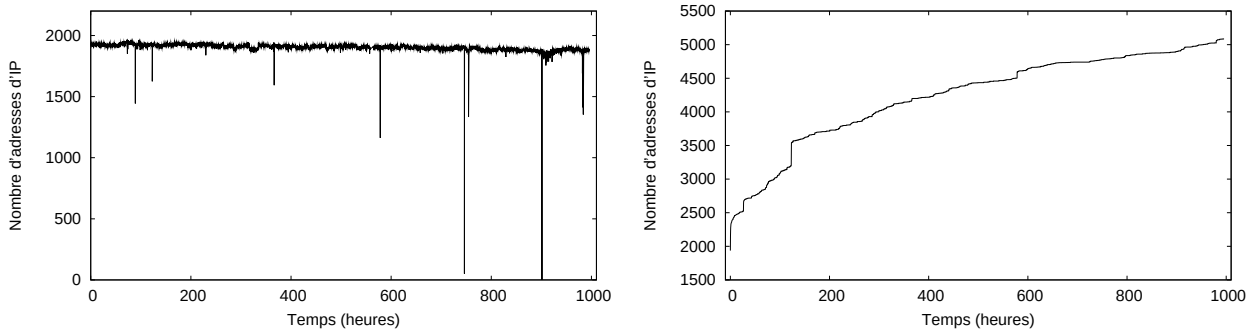


FIG. 3.12 – **Gauche** : nombre d'adresses IP observées par passe en fonction du temps en heures. **Droite** : nombre d'adresses IP observées depuis le début de la mesure en fonction du temps en heures.

La figure 3.13 (gauche) montre le nombre d'adresses IP vues qui ne sont plus observées après un temps t (ligne continue), et le nombre d'adresses parmi celles-ci qui ont répondu au dernier `ping`, effectué à la fin de la mesure (ligne pointillée). Ces deux courbes ont le même rythme de croissance, avec un écart relativement faible. La figure 3.13 (droite) présente le rapport entre ces deux courbes. On constate que 80% environ des adresses IP qui disparaissent de nos mesures existent toujours à la fin de la mesure puisqu'elles répondent au `ping`. Ce résultat montre que les changements de routage sont la cause principale de la disparition des adresses IP dans nos mesures.

Ces deux expériences montrent que la dynamique du routage entre les adresses IP existantes joue un rôle fondamental dans nos observations.

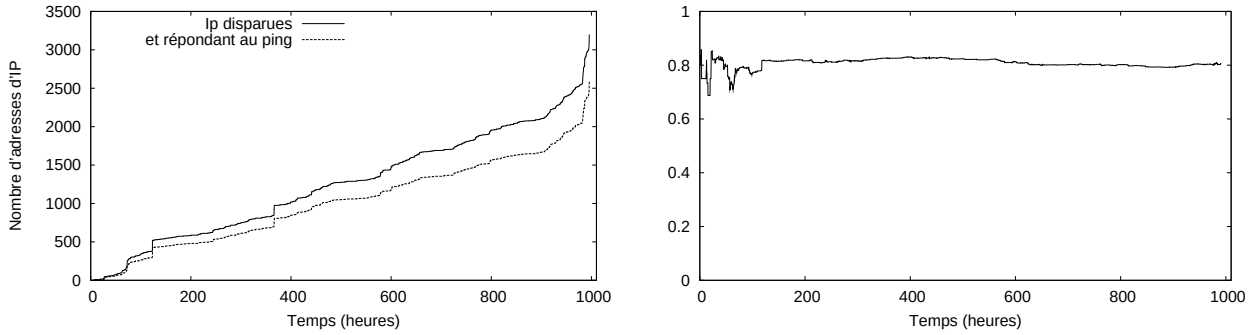


FIG. 3.13 – **Gauche** : nombre d'adresses IP en fonction du temps t, qui sont observées avant le temps t et qui ne sont plus observées après t (courbe pleine) ; nombre de celles parmi ces adresses IP qui ont répondu au dernier `ping` (courbe en pointillés). **Droite** : rapport entre le nombre d'adresses IP qui ne sont plus observées et le nombre de celles qui ne sont plus observées mais qui ont répondu au dernier `ping`.

3.4 Conclusion

Nous avons montré dans ce chapitre une observation inattendue concernant la dynamique de la topologie de l'internet : un moniteur qui effectue périodiquement une mesure du type `traceroute` vers un ensemble fixe de destinations donne une vue de la topologie qui est en constante évolution pendant toute la mesure. En particulier, on observe continuellement et à un rythme surprenant de nouvelles adresses IP. Ce phénomène est observé avec divers moniteurs et destinations et semble universel.

Nous avons décrit ce phénomène en détail et tenté de déterminer ses causes. Nous avons écarté certaines explications possibles : les adresses IP dynamiques parmi les destinations et les routeurs qui répondent avec un grand nombre d'adresses IP, par exemple des adresses aléatoires. Cela a prouvé que le phénomène observé n'est pas un artéfact de mesure mais une propriété de la dynamique de la topologie.

Nous avons isolé deux facteurs induisant la découverte de nouvelles adresses IP : on découvre de nouveaux AS et on découvre de nouvelles adresses IP à l'intérieur des AS déjà découverts.

À partir de l'historique des tables de routage enregistrées par le projet *Route Views*, nous avons pu conclure que la découverte de nouveaux AS est causée par la dynamique de routage BGP. Ces AS existaient avant le début de la mesure et sont devenus visibles suite à des changements de routage.

Suite à ces conclusions sur les AS, nous avons étudié le rôle des changements de routage dans l'apparition de nouvelles adresses et la disparition d'anciennes. Cette étude a montré que les changements de routage jouent un rôle important dans ces phénomènes, et que beaucoup des adresses qui apparaissent existaient avant le début de la mesure mais n'étaient pas visibles.

Nous avons montré qu'une grande partie des disparitions et apparitions d'adresses IP n'en étaient pas en réalité, mais étaient seulement dues à une dynamique de routage qui cache certaines adresses IP de nos mesures et en fait voir d'autres qui existaient. Pendant ce temps, il y a effectivement de nouvelles adresses qui sont allouées et d'autres qui sont mises hors service et, nous ne savons pas quel est l'impact de ces adresses sur le phénomène observé. Il serait intéressant de pouvoir le mettre en évidence. Les adresses IP apparaissent et disparaissent avec des liens mais des liens peuvent aussi apparaître et disparaître entre des adresses IP existantes. L'observation de l'évolution des liens

pourraient apporter des informations intéressantes pour comprendre le phénomène observé sur les adresses IP. De plus, cela apportera une meilleure compréhension de la dynamique de la topologie d'une manière générale, et aidera à la modélisation.

Une autre direction consiste à avoir recours aux simulations pour étudier le phénomène. Pour cela il faut avoir un modèle dynamique de la topologie, ce qui ouvre un autre vaste sujet sur la dynamique de l'internet.

Chapitre 4

Mesure de la topologie de l'internet : évaluation de l'approche distribuée

Ainsi qu'on l'a vu dans le chapitre 1, de nombreux travaux ont entrepris de mesurer la topologie de l'internet de manière distribuée : à partir de chaque point de mesure, un moniteur découvre une partie de la topologie en collectant des routes vers un ensemble de destinations. Notre objectif dans ce chapitre est d'évaluer cette méthode de mesure sur deux points essentiellement :

- l'influence du nombre de moniteurs et de destinations ainsi que de leur choix ;
- la différence entre la mesure obtenue et l'objet réel.

Des travaux ont déjà été effectués sur ce thème [8, 27, 28] et plus récemment [60]. En particulier, les auteurs de [27, 28] ont étudié de manière systématique l'influence du nombre de moniteurs et de destinations sur les propriétés de la vue de la topologie obtenue lorsqu'on simule une mesure sur un graphe connu.

Nous utiliserons ici la même approche mais en nous appuyant sur des données réelles qui proviennent des mesures radar décrites dans le chapitre 2. Il faut noter que ces données ont l'avantage de fournir les routes réelles entre moniteurs et destinations, et donc notre étude ne repose pas sur des simulations.

Pour atteindre nos objectifs, l'idéal serait de comparer la mesure obtenue avec la topologie réelle, ce qui est impossible car nous ne la connaissons pas. Nous utiliserons donc une méthode introduite dans [36] que nous présenterons dans la section 4.2, qui consiste à comparer un graphe avec les différentes vues qu'on peut obtenir en l'explorant.

Dans la section 4.3 nous présenterons une analyse de l'influence du choix des moniteurs et des destinations sur la vue obtenue. Nous étudierons cet impact sur la vue en termes de nombre de nœuds et de liens, puis sur d'autres propriétés.

En prenant en compte l'influence du choix des moniteurs et des destinations, nous comparerons dans la section 4.4 les différentes vues obtenues avec différents nombres de moniteurs et de destinations au moyen de propriétés statistiques de graphe. Nous montrerons que les propriétés statistiques se comportent de manières différentes, c'est-à-dire que certaines semblent pouvoir être estimées correctement à partir d'un échantillon de la topologie, et d'autres non.

4.1 Description des données

Les données utilisées dans ce chapitre proviennent de la mesure *radar* décrite dans le chapitre 2 (voir section 2.3 pour plus de détails). Afin d'étudier l'impact du nombre de moniteurs et de destina-

tions sur la vue obtenue, nous avons besoin de mesures pour lesquelles les moniteurs utilisent le même ensemble de destinations. Nous avons donc utilisé les données d'un ensemble S de 11 moniteurs qui ont effectué la mesure avec un même ensemble D de 3 000 destinations. À chaque passe de mesure i effectuée par le moniteur j, les chemins mesurés entre ce moniteur et les destinations forment un arbre $t_{i,j}$ dans lequel la racine est le moniteur et les feuilles sont les destinations. Les nœuds sont des adresses IP vues sur les chemins du moniteur vers les destinations et les liens représentent les sauts au niveau IP d'un nœud à un autre. La partie de la topologie observée à partir du moniteur j, que nous notons g_j est l'union (non-orientée) de tout ce qui a été vu à toutes les passes de mesure effectuées par ce moniteur :

$$g_j = \bigcup_i t_{i,j}.$$

Il faut noter que si une machine n'a pas répondu à un paquet, alors elle sera représentée dans l'arbre $t_{i,j}$ correspondant par une étoile. Dans un tel arbre, toutes les étoiles sont considérées comme différentes. Cela pose problème quand nous voulons construire l'union g_j des arbres : on ne peut pas savoir si une étoile qui apparait au même endroit dans plusieurs arbres correspond à une même machine ou pas. Pour résoudre ce problème, nous avons choisi de ne pas prendre en compte les étoiles et les liens qui leurs sont associés dans les données. Cette solution n'est pas sans inconvénient car les arbres $t_{i,j}$ ne sont plus connexes dans ce cas. Cependant, cela ne constitue pas un problème pour ce que nous voulons faire ici, car l'union des arbres $t_{i,j}$ est presque connexe : dans chaque graphe g_j de nos données, la plus grande composante connexe rassemble au moins 92% des nœuds.

Remarquons d'ores et déjà que tous les moniteurs ne sont pas équivalents car les tailles des graphes g_j diffèrent beaucoup : le plus petit a 16 469 nœuds alors que le plus grand en a 26 447.

Nous allons maintenant définir le graphe de la topologie observée, noté G, comme étant l'union des g_j pour tous les moniteurs :

$$G = \bigcup_j g_j.$$

C'est un graphe de 42 141 nœuds et 165 438 liens.

Il est important de noter que l'union des graphes ne garde qu'une seule copie de chaque lien. Par conséquent, tous les graphes que nous considérons n'ont pas de liens multiples et ne contiennent aucune boucle.

4.2 Méthodologie

Soit $G = (V, E)$ le graphe défini dans la section 4.1 représentant notre vue de la topologie. On note $n = |V|$ son nombre de nœuds et $m = |E|$ son nombre de liens.

4.2.1 Les explorations de G

Notre approche consiste à étudier les vues de G obtenues en l'explorant avec des sous-ensembles des moniteurs et des destinations. Nous allons comparer les propriétés du graphe original G à celles de ces vues.

Nous appelons $g_{j,k}$ le graphe obtenu en faisant l'union des chemins entre le moniteur j et la destination k : ce sont tous les nœuds et liens apparus dans les chemins entre j et k dans toutes les passes de mesure. Si S' est un sous-ensemble de moniteurs et D' un sous-ensemble de destinations,

$G_{S'D'}$ est la vue de G obtenue avec les moniteurs de S' et les destinations de D', et est définie ainsi :

$$G_{S'D'} = \bigcup_{\substack{j \in S' \\ k \in D'}} g_{j,k}.$$

4.2.2 Courbes en niveaux de gris

Si nous considérons une propriété quelconque p[1], comparer cette propriété entre le graphe original G et chaque vue $G_{S'D'}$ est complexe. Cela conduit d'une manière générale à comparer les valeurs correspondant à un très grand nombre de graphes. Dans le but de rendre l'analyse plus facile et pratique, nous allons faire usage des courbes en niveau de gris de manière similaire à ce qui a été fait dans [27, 28]. Considérons un rectangle de longueur $|D|$ et de hauteur $|S|$. Chaque point (d,s) du rectangle correspond à un graphe $G_{S'D'}$, avec $d = |D'|$ et $s = |S'|$. Le point est d'un gris dont le niveau dépend de la valeur de la propriété p correspondant à ce graphe. Il va du noir pour $p = 0$ au blanc quand p est égale à sa valeur maximale. La variation du niveau de gris est linéaire : si un point est deux fois plus sombre qu'un autre point alors la valeur associée au premier point est aussi deux fois plus petite. Notons que la valeur maximale d'une propriété n'est pas nécessairement celle du graphe original G mais peut-être atteinte avec une de ses vues $G_{S'D'}$. Par contre, on aura toujours ici $G_{SD} = G$. Par conséquent, les points plus sombres que le point du coin supérieur droit correspondent aux cas où la propriété est sous-estimée, tandis que les points plus clairs correspondent à des vues où elle est surestimée. Pour finir, afin d'améliorer la lecture de ces courbes comme dans [27, 28], nous avons ajouté les lignes de niveau 10%, 50% et 90%. La ligne de niveau l% est définie comme étant l'ensemble des points où p vaut l% de la valeur maximale à 0.01 près. La ligne de niveau 10% est représentée avec la couleur blanche, la ligne niveau 50% est représentée avec deux couleurs : noire pour l'horizontale et blanche pour la verticale, et la ligne de niveau 90% est noire [2]. Ces lignes sont d'une aide précieuse pour l'interprétation des courbes en niveau de gris, comme on le constatera par la suite.

Il faut remarquer qu'on a autant de graphes $G_{S'D'}$ que de points (d,s) dans le rectangle ; obtenir de tels graphes pour tous les points représente beaucoup de calculs. Dans le but de réduire les calculs, nous procédons de la manière suivante : étant donné un ordre sur les $|S|$ moniteurs et un ordre sur les $|D|$ destinations, le point (d,s) correspond au graphe $G_{S'D'}$ tel que S' contient les s premiers moniteurs de l'ordre sur S et D' contient les d premières destinations de l'ordre sur D. Ceci entraîne que le graphe correspondant au point $(s-1, d-1)$ est contenu dans le graphe correspondant au point (s, d), ce qui accélère les calculs.

4.3 Impact du choix des moniteurs et des destinations

La vue de la topologie obtenue dépend du nombre de moniteurs et du nombre de destinations, mais aussi de quels moniteurs et destinations sont utilisés car des ensembles différents de moniteurs (ou de destinations) de même taille peuvent fournir des vues très différentes. Par conséquent la courbe en niveau de gris définie dans la section 4.2 dépend du choix de l'ordre utilisé sur les moniteurs et les destinations : deux ordres différents produiront deux courbes différentes.

Dans cette section, nous évaluons l'impact du choix des sources et des destinations sur la vue qu'on obtient. Nous allons d'abord étudier le nombre de nœuds et de liens de la vue obtenue, puis

[1] Toutes les propriétés que nous considérons dans ce chapitre ont des valeurs réelles non-négatives.

[2] Voir un exemple typique où toutes les lignes sont bien visibles dans la figure 4.5 (gauche).

des propriétés plus complexes.

4.3.1 Nombre de nœuds et nombre de liens

Dans cette partie nous allons mettre en évidence l'impact du choix des moniteurs et des destinations sur le nombre de nœuds observés. Nous allons voir que ce choix a une grande influence.

Comme nous l'avons déjà expliqué ci-dessus, étudier toutes les vues possibles de la topologie obtenues avec des ensembles de moniteurs et destinations représente beaucoup de calculs : pour chaque nombre de moniteurs s et chaque nombre de destinations d on a $C^s_{|S|} \times C^d_{|D|}$ possibilités, et donc $\sum_s \sum_d C^s_{|S|} \times C^d_{|D|}$ possibilités au total, ce qui est considérable.

Nous allons donc considérer un ordre sur les moniteurs (resp. sur les destinations) et construire le graphe G obtenu en considérant les moniteurs (resp. destinations) en fonction de cet ordre, et en ajoutant à chaque étape tous les nœuds et liens vus avec le moniteur (resp. la destination)[3] correspondant.

Pour un ordre donné, nous pouvons alors obtenir la courbe montrant l'évolution de la taille du graphe (en termes de nombre de nœuds ou de nombre de liens) en fonction du nombre de moniteurs (resp. de destinations) considérés. Les différences entre les courbes correspondant à différents ordres donneront alors une idée de l'impact du choix des moniteurs (resp. destinations).

Pour faire apparaître les différences, on peut chercher les écarts possibles entre les ordres : pour un nombre k de moniteurs, quels sont le maximum et le minimum de la taille des graphes (en termes de nombre de nœuds) qu'il est possible d'obtenir en fusionnant les nœuds et liens vus par k moniteurs (resp. destinations) ?

Nous désignons par $M_s(k)$ la taille maximale d'un graphe obtenu avec k moniteurs :

$$M_s(k) = \max_{S' \subseteq S, |S'|=k} |G_{S'D}|,$$

et par $m_s(k)$ la taille minimale d'un tel graphe :

$$m_s(k) = \min_{S' \subseteq S, |S'|=k} |G_{S'D}|.$$

De façon similaire, nous désignons par $M_d(k)$ la taille maximale d'un graphe obtenu avec k destinations :

$$M_d(k) = \max_{D' \subseteq D, |D'|=k} |G_{SD'}|,$$

et par $m_d(k)$ la taille minimale d'un tel graphe :

$$m_d(k) = \min_{D' \subseteq D, |D'|=k} |G_{SD'}|.$$

Pour calculer les fonctions maximum et minimum il faut calculer les graphes correspondant à tous les choix possibles afin de trouver le plus grand et le plus petit. Cela représente une quantité de calculs qui ne peut être réalisée. Nous proposons donc une heuristique gloutonne pour approximer le maximum : à chaque étape nous considérons le graphe obtenu dans l'étape précédente puis nous choisissons le moniteur (resp. destination) qui augmente le plus le nombre de nœuds quand on

[3] Les nœuds et liens vus avec un moniteur (resp. une destination) sont ceux appartenant aux chemins de ce moniteur vers les destinations considérées (resp. des moniteurs considérés vers cette destination).

l'ajoute au graphe. Nous désignons par $M'_s(k)$ (resp. $M'_d(k)$) la taille du graphe obtenu à l'étape k. Inversement, nous approximons le minimum en commençant avec le moniteur (resp. destination) qui découvre le plus petit nombre de nœuds, puis nous choisissons à chaque étape le moniteur (resp. destination) qui augmente le moins le nombre de nœuds quand on l'ajoute. Nous désignons par $m'_s(k)$ (resp. $m'_d(k)$) la taille du graphe obtenu à l'étape k. Nous supposons que ces heuristiques sont proches des valeurs réelles ; elles en sont en tous cas une borne inférieure et une borne supérieure, respectivement.

Afin de compléter cette étude, nous calculons aussi le maximum et le minimum obtenus sur 1000 ordres aléatoires : nous avons choisi 1000 ordres aléatoires sur les moniteurs (ou les destinations) et observé la taille des graphes obtenus en ajoutant un à un les moniteurs (ou les destinations) dans cet ordre ; pour chaque k, nous avons sélectionné la plus grande et la plus petite valeur observée sur les 1000 ordres.

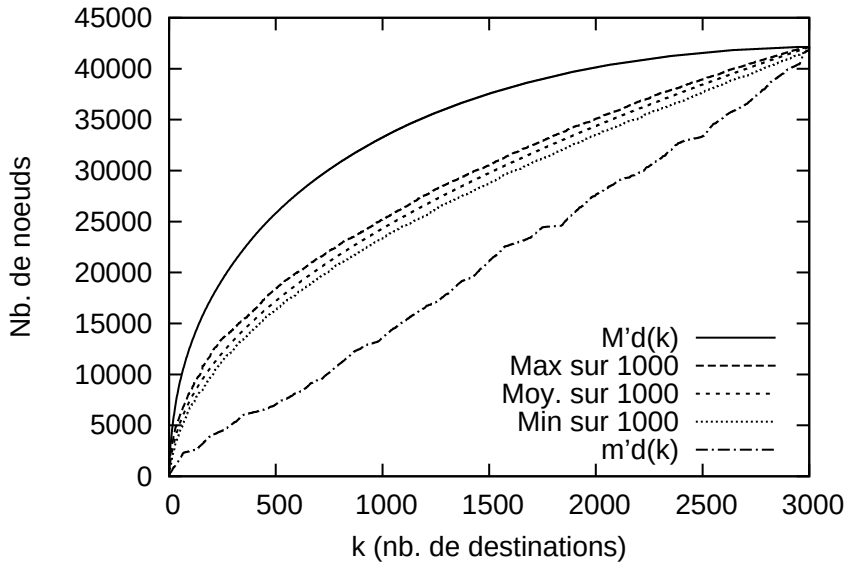


FIG. 4.1 – Impact du choix des destinations sur le nombre de nœuds de la vue obtenue. De haut en bas : $M'_d(k)$, valeur maximale observée sur 1 000 ordres, moyenne sur 1 000 ordres, valeur minimum observée sur 1 000 ordres et $m'_d(k)$.

La figure 4.1 présente ces différentes notions de maximum et minimum, ainsi que la taille moyenne observée sur 1000 ordres, en fonction du nombre de destinations. Il faut noter que 1000 est négligeable devant le nombre total d'ordres possibles qui est 3000!. Nous observons plusieurs choses. Premièrement, il y a une différence importante entre le maximum $M'_d(k)$ et le minimum $m'_d(k)$. Pour 500 destinations par exemple, le nombre de nœuds observés varie entre 7 500 environ à plus de 25 000 ; quand l'ordre maximum découvre 80% des nœuds, l'ordre minimum n'en voit que 34%. Cela montre à quel point l'ordre dans lequel on considère les destinations peut avoir une influence importante sur les propriétés observées.

Cependant, cette différence n'est pas si claire quand nous considérons des ordres aléatoires : les courbes de la moyenne, du minimum et du maximum parmi les 1 000 ordres aléatoires sont proches les unes des autres et loin de $M'_d(k)$ et $m'_d(k)$. Cela semble indiquer que concernant le nombre de nœuds, il y a des ordres atypiques qui donnent des résultats vraiment différents de la moyenne, mais que la plupart des ordres sont proches de la moyenne.

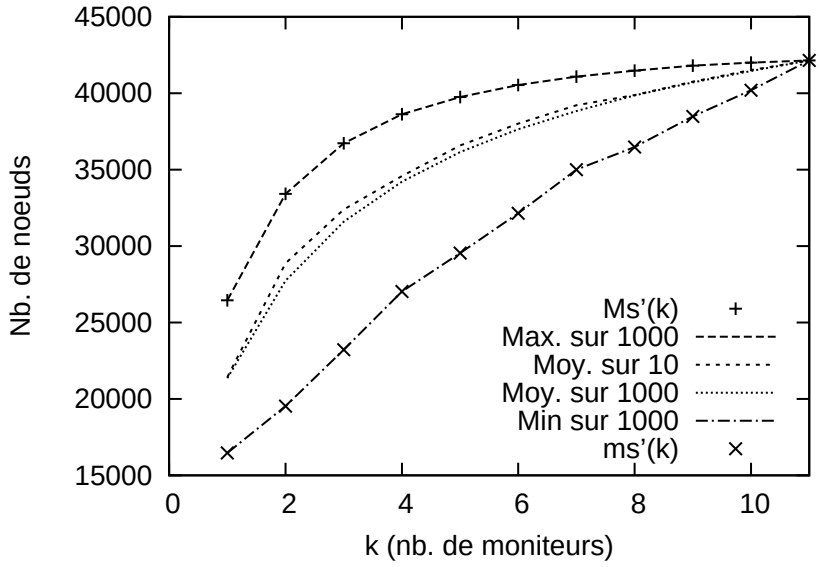


FIG. 4.2 – Impact du choix des moniteurs sur le nombre de nœuds de la vue observée. De haut en bas : $M'_s(k)$, valeur maximale observée sur 1 000 ordres, moyenne sur 1 000 ordres, moyenne sur 10 ordres, valeur minimale observée sur 1 000 ordres et $m'_s(k)$.

La figure 4.2 présente les différentes fonctions de maximum et minimum, ainsi que la taille moyenne observée sur 1000 ordres aléatoires en fonction du nombre de moniteurs (1000 est de loin inférieur à 1% du nombre total d'ordres, qui est 11! $\sim 40.10^6$). On constate que contrairement aux destinations, le maximum et le minimum sur 1000 ordres aléatoires sont égaux à $M'_s(k)$ et $m'_s(k)$ et ne sont pas proches de la moyenne. Cela montre aussi que les ordres sur les moniteurs ne sont pas aussi proches les uns des autres que ceux sur les destinations. Dans le but d'estimer la vitesse de convergence, on a alors calculé la moyenne sur 10 ordres (voir figure 4.2). On constate une convergence relativement rapide de la moyenne car on peut voir que la moyenne sur les 1000 ordres n'est pas très différente de celle sur 10.

Comme pour les destinations, l'écart entre $M'_s(k)$ et $m'_s(k)$ est important, la différence entre ces deux fonctions atteint jusqu'à 33% du nombre total de nœuds avec seulement 2 moniteurs. Cela montre que les ordres sur les destinations et également sur les moniteurs jouent un rôle important.

Nous avons fait les mêmes courbes pour le nombre de liens observés. Nous ne les présentons pas ici car elles sont similaires à celles du nombre de nœuds.

Nous concluons finalement que l'ordre sur les moniteurs et les destinations a une influence sur le nombre de nœuds et le nombre de liens du graphe observé. Nous avons aussi montré que considérer la moyenne sur 10 ordres aléatoires semble donner une bonne approximation de la moyenne et donc des observations attendues. Nous allons voir dans la section suivante que cet ordre influence aussi d'autres propriétés du graphe.

4.3.2 Impact de l'ordre sur d'autres propriétés

Étant donné un ordre sur les moniteurs et les destinations, nous calculons les propriétés statistiques du graphe $G_{S'D'}$ obtenu avec les $|S'|$ premiers moniteurs et les $|D'|$ premières destinations de l'ordre. Nous avons présenté ci-dessus l'impact de l'ordre sur la proportion de nœuds et liens décou-

degrée moyen	nb moniteurs	nb destinations	clust. global	nb moniteurs	nb destinations
7.852343	11	2993	0.160401	1	244
7.856401	11	2976	0.256501	1	5
7.855130	11	2975	0.125881	3	358
7.853955	11	2914	0.496428	1	353
7.852128	11	2997	0.152805	5	28
7.854578	11	2976	0.250000	1	1
7.854771	11	2984	0.154679	2	244
7.891898	11	2934	0.131765	4	396
7.853233	11	2973	0.185765	2	3
7.853207	11	2996	0.136787	3	503
graphe original G					
7.851641	11	3000	0.101155	11	3000

TAB. 4.1 – Valeurs maximales du degré moyen et du coefficient de clustering global atteintes pour 10 ordres différents. Chaque ligne correspond pour un ordre aléatoire, et donne les valeurs maximales observées, ainsi que le nombre de moniteurs $|S'|$ et de destinations $|D'|$ avec lequel elles sont atteintes.

verts et montré qu'il induit des variations importantes. Nous allons maintenant voir son impact sur d'autres propriétés.

La table 4.1 présente les valeurs maximales observées pour le degré moyen et le coefficient de *clustering* global pour 10 ordres aléatoires sur les sources et les destinations. Nous observons que le maximum est différent pour tous les ordres et que l'écart peut être important dans certains cas. Par conséquent la valeur maximale observée dans un ordre donné ne peut être considérée comme représentative. Cela montre les différences importantes qu'il peut y avoir entre différents ordres aléatoires. En d'autres termes les propriétés observées avec un ordre choisi peuvent être spécifiques à celui-ci. Notons que pour des propriétés, comme le degré moyen par exemple, on n'observe pas de différence forte pour le maximum entre différents ordres (ce qui n'implique pas qu'il n'y ait pas de différence entre les ordres).

En conclusion, le choix des moniteurs et des destinations peut avoir un impact fort sur nos observations. Cet impact va aussi jouer sur les différentes courbes en niveau de gris (définies dans la section 4.2) correspondant à différents ordres aléatoires : on pourra percevoir des différences entre des courbes correspondant à des ordres différents. Cependant, calculer toutes les courbes en niveaux de gris pour toutes les propriétés et tous les ordres possibles représente un coût de calcul trop élevé.

Dans le but d'avoir une estimation représentative de chaque propriété pour chaque nombre de destinations et de moniteurs considéré, nous calculons 10 courbes en niveau de gris, chacune correspondant à un ordre aléatoire sur les moniteurs et sur les destinations. Puis nous calculons la courbe en niveau de gris moyenne définie comme suit : pour une propriété p donnée, chaque point (d,s) de la courbe est la moyenne des points correspondants des 10 courbes en niveau de gris. Nous présenterons des exemples de courbes en niveau de gris correspondant à des ordres différents ainsi que leur moyenne. Nous allons voir que pour certaines propriétés une seule courbe en niveau de gris peut être représentative tandis que pour d'autres, ce n'est pas le cas.

4.4 Propriétés statistiques

Dans cette section, nous allons étudier des propriétés statistiques de graphe qui sont parmi les plus importantes. Pour chaque propriété, nous allons montrer l'impact des moniteurs et des destinations (leur nombre et leur choix) à l'aide des courbes en niveau de gris définies dans la section 4.2.

4.4.1 Degré moyen et densité

Les deux premières propriétés que nous allons étudier sont le degré moyen et la densité. Nous présentons dans la figure 4.3 (gauhe) la variation de la densité en fonction du nombre de destinations considérées, pour différents nombres de moniteurs. On observe que l'écart entre les différents nombres de moniteurs est faible dès que l'on considère un certain nombre de destinations. Cet écart tend à devenir très petit quand le nombre de destinations augmente. Nous avons fait ces calculs pour un nombre significatif d'ordres aléatoires différents sur les moniteurs et les destinations et avons constaté que l'ordre n'a pas d'influence significative sur ces observations. Nous ne présentons pas ces courbes ici.

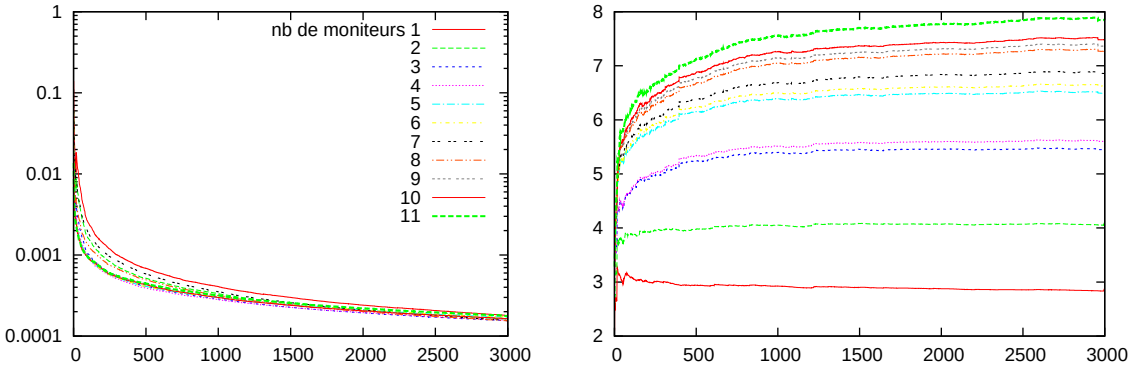


FIG. 4.3 – Variation de la densité (gauche) et du degré moyen (droite) en fonction du nombre de destinations considérées. Chaque courbe correspond à un nombre de moniteurs différent. La légende de la courbe du haut est la même que pour celle du bas.

Nous ne présentons pas de courbe en niveaux de gris pour la densité car l'écart entre le maximum et la plupart des valeurs est très important. Par conséquent la courbe est de couleur quasi-uniforme (on ne perçoit aucune variation) et donc inexploitable.

La figure 4.3 (droite) présente la variation du degré moyen en fonction du nombre de destinations considérées, pour différents nombres de moniteurs. On constate qu'après une forte variation au début, les courbes semblent atteindre un plateau. Il faut remarquer qu'elles ne sont cependant pas constantes : les courbes correspondant à des petits nombres de moniteurs semblent décroître légèrement quand le nombre de destinations augmente, alors que les courbes correspondant à des nombres élevés de moniteurs semblent croître légèrement.

Cela correspond à un changement dans les densités correspondantes : pour un petit nombre de destinations, considérer tous les moniteurs mène à une densité plus faible que considérer un seul moniteur. Cette tendance est inversée quand le nombre de destinations est grand et, quand toutes les destinations sont considérées, augmenter le nombre de moniteurs conduit à un graphe plus dense.

La figure 4.4 présente trois courbes en niveau de gris pour le degré moyen, correspondant à des ordres aléatoires différents sur les moniteurs et les destinations. Comme on peut le voir il n'y a pas

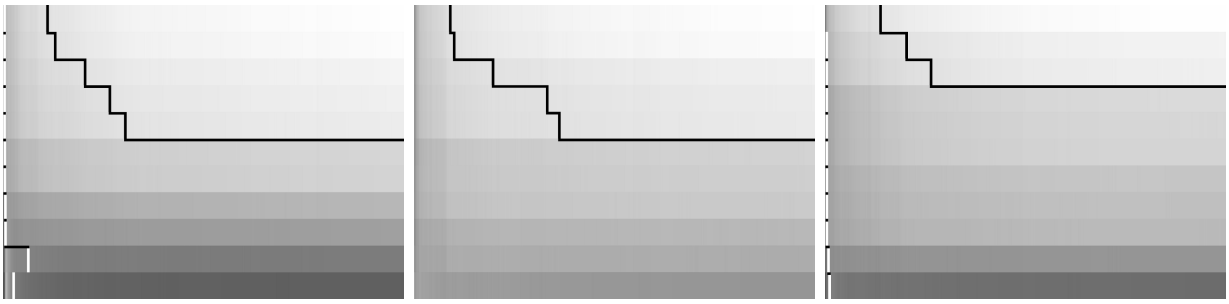


FIG. 4.4 – Degré moyen pour trois ordres aléatoires différents.

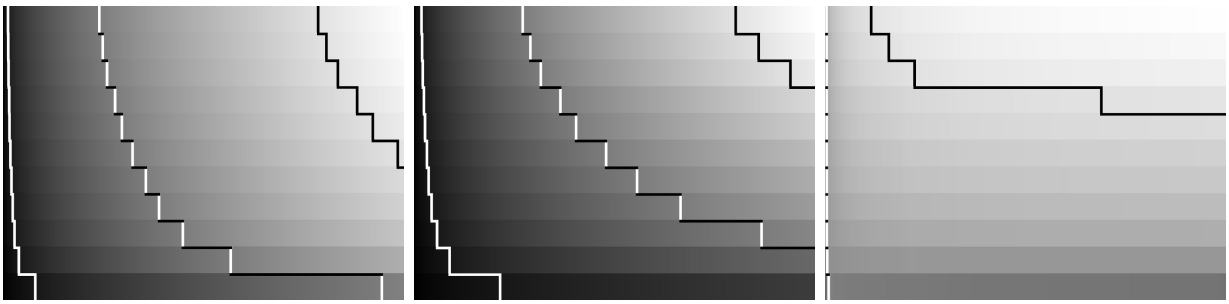


FIG. 4.5 – De gauche à droite : nombre de nœuds, nombre de liens et degré moyen. Moyenne sur dix ordres aléatoires différents.

beaucoup de différences entre ces courbes : celle du milieu a un gris plus uniforme (la ligne de niveau 50% n'apparaît pas) et est donc plus précise puisque la différence entre le maximum et le minimum observés est petite. Le nombre de moniteurs et de destinations nécessaires pour atteindre une certaine précision dépend aussi de leur choix : les courbes à gauche et au milieu de la figure 4.4 atteignent la ligne de niveau 90% avec moins de moniteurs et de destinations que celle de la figure 4.4 à droite.

La figure 4.5 montre la moyenne de 10 courbes en niveau de gris pour le nombre de nœuds (gauche), le nombre de liens (milieu) et le degré moyen (droite). Ces trois propriétés augmentent quand le nombre de moniteurs et de destinations augmentent, du moins dès que celui-ci devient suffisamment grand. Ceci est trivial pour le nombre de nœuds et le nombre de liens car utiliser davantage de moniteurs et de destinations conduit à observer plus de nœuds et de liens. Le cas du degré moyen est différent, car l'augmentation du nombre de moniteurs et de destinations peut en théorie faire augmenter le nombre de liens moins que le nombre de nœuds, et donc entraîner une décroissance du degré moyen. En effet, la figure 4.3 (bas) montre que pour un petit nombre de moniteurs, le degré moyen décroît quand le nombre de destinations augmente. Cependant une fois que le nombre de moniteurs considérés atteint 3, le degré moyen augmente avec le nombre de moniteurs et de destinations.

Le degré moyen est obtenu en divisant deux autres propriétés (le nombre de liens et le nombre de nœuds). Cela a des conséquences importantes. En effet si les deux propriétés sont évaluées avec le même biais, le quotient peut être exempt de ce biais : l'estimation du degré moyen est bonne quand le quotient entre le nombre de nœuds et le nombre de liens est précis, même si ces nombres eux-mêmes sont faux. C'est bien ce qu'on observe en pratique : la courbe du degré moyen présente un gris beaucoup plus uniforme que celles du nombre de nœuds et du nombre de liens, ce qui montre qu'il est mieux estimé que ces deux quantités.

En conclusion, nos analyses ont révélé une relation non triviale entre le degré moyen et la densité.

Nous avons aussi montré que le degré moyen est beaucoup mieux estimé que le nombre de nœuds et de liens. Ceci confirme les résultats obtenus par simulations dans [27, 28].

4.4.2 Coefficient de clustering

Le calcul du coefficient de *clustering* global dépend du nombre de triangles et de triplets dans le graphe, exactement comme le degré moyen dépend du nombre de liens et du nombre de nœuds. Par conséquent, nous allons étudier ces trois propriétés ensemble.

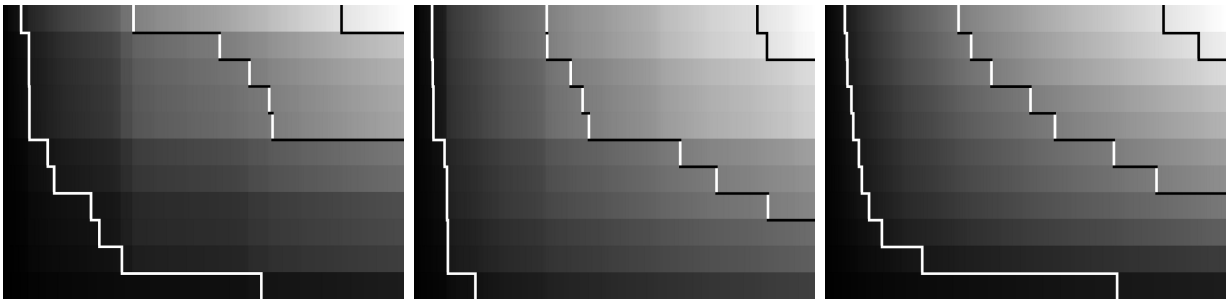


FIG. 4.6 – Nombre de triangles. Gauche et milieu : deux ordres aléatoires sur les moniteurs et les destinations. Droite : moyenne sur 10 ordres.

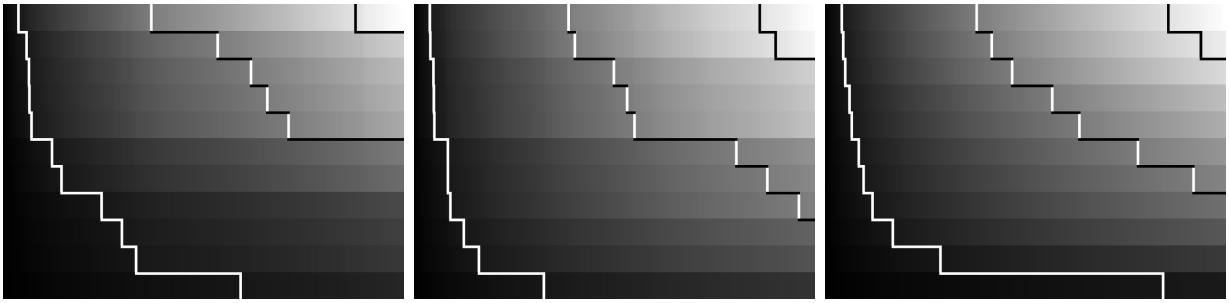


FIG. 4.7 – Nombre de triplets. Gauche et milieu : deux ordres aléatoires sur les moniteurs et les destinations. Droite : moyenne sur 10 ordres. On utilise les même ordres que ceux de la figure 4.6.

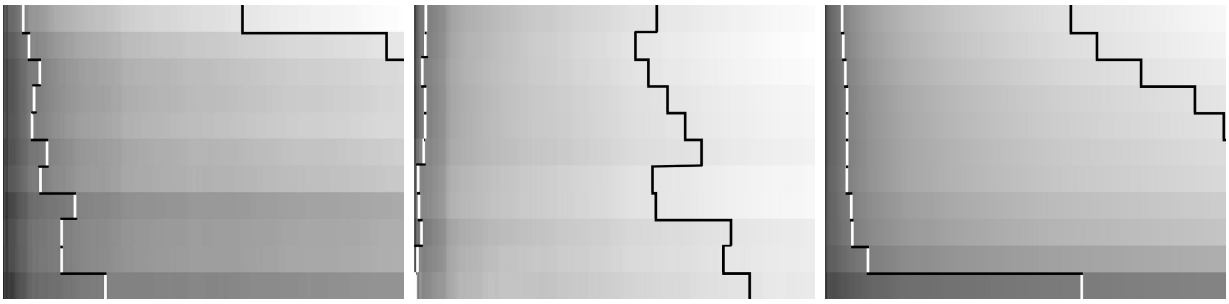


FIG. 4.8 – Coefficient de *clustering* local. Gauche et milieu : deux ordres aléatoires sur les moniteurs et les destinations. Droite : moyenne sur 10 ordres. On utilise les même ordres que ceux de la figure 4.6.

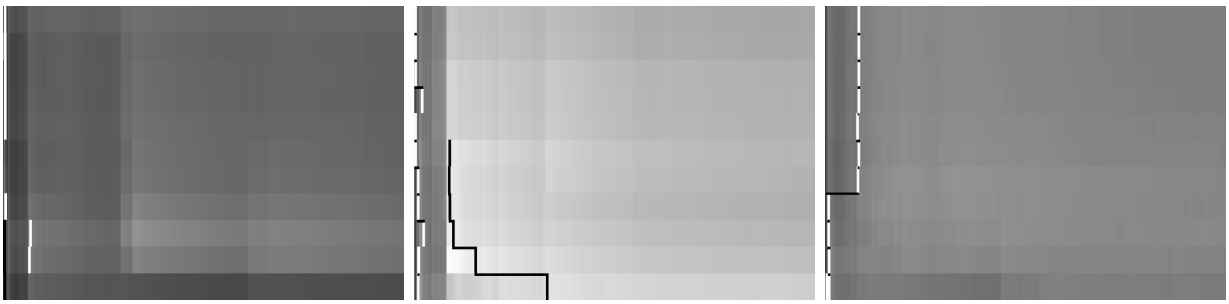


FIG. 4.9 – Coefficient de *clustering* global. Gauche et milieu : deux ordres aléatoires sur les moniteurs et les destinations. Droite : moyenne sur 10 ordres. On utilise les même ordres que ceux de la figure 4.6.

La précision de l'estimation des coefficients de *clustering* globaux et locaux dépend de la rapidité de découverte des triangles par rapport aux triplets. Nous pouvons obtenir une bonne estimation si nous découvrons à la même vitesse les triangles et les triplets, même si leur nombre est mal estimé. L'ordre dans lequel on considère les moniteurs et les destinations a une influence sur la vitesse de découverte des triplets et triangles. Cependant, cet impact est encore plus fort sur les coefficients de *clustering* ainsi qu'on peut le voir sur les figures 4.8 et 4.9 : pour deux ordres différents, les courbes des coefficients de clustering peuvent être très différentes.

Nous allons comparer le coefficient de *clustering* global et le coefficient de *clustering* local. La variation du *clustering* global est très différente de celle du *clustering* local, comme nous pouvons le voir dans les figures 4.8 et 4.9. Le gris du *clustering* local tend à devenir plus clair quand le nombre de moniteurs et de destinations augmente, et cela quel que soit l'ordre considéré. Mais il faut remarquer que l'ordre joue beaucoup sur sa vitesse d'évolution. Si nous considérons les deux ordres aléatoire de la figure 4.8, on observe un comportement différent : pour l'un de ces ordres (figure 4.8, milieu) on atteint la ligne de niveau 90% avec seulement un moniteur, et la variation du gris n'est pas importante quand le nombre de moniteurs augmente. Ceci veut dire que l'ordre considéré sur les destinations dans ce cas-ci donne une bonne estimation du *clustering* local. Par contre on voit dans la figure 4.8 (gauche) qu'un autre ordre sur les destinations montre un comportement différent : dans ce cas l'augmentation du nombre de moniteurs influence beaucoup la variation du gris : il faut au moins 10 moniteurs (et un grand nombre de destinations) pour atteindre la ligne de niveau 90%.

On observe pour le *clustering* global (figure 4.9, gauche et milieu) que le gris ne devient pas plus clair quand le nombre de moniteurs et de destinations augmente. Au contraire les points les plus clairs sont éloignés du coin supérieur droit, et ils semblent ne pas dépendre du nombre de moniteurs et de destinations. Cela signifie que le *clustering* global est toujours surestimé dans une vue $G_{S'D'}$, quel que soit l'ordre [4] et même pour des petits nombres de moniteurs et de destinations. L'écart entre les valeurs maximales atteintes par le *clustering* global pour des ordres différents, qui représente le point auquel il est surestimé, est important, voir la table 4.1.

Nous allons maintenant considérer les courbes en niveau de gris moyennes pour le nombre de triangles, le nombre de triplets, le *clustering* global et le *clustering* local, voir les figures 4.6, 4.7, 4.8 et 4.9 (droite). La courbe du *clustering* global présente un niveau de gris uniforme, ce qui montre que la plupart des valeurs sont proches les unes des autres. L'augmentation du nombre de moniteurs et de destinations n'a presque aucune influence. Nous observons un tout autre comportement avec le *clustering* local : augmenter le nombre de moniteurs et de destinations améliore l'estimation du *clustering* local.

En conclusion, les moniteurs et les destinations (par leur nombre ou leur choix) influencent différemment les *clustering* globaux et locaux. Le *clustering* local dépend plus du nombre de moniteurs et de destinations que le *clustering* global. On peut donc améliorer l'estimation du *clustering* local en augmentant le nombre de moniteurs et de destinations, même si l'influence de l'ordre est importante. Le *clustering* global est plus influencé par le choix des moniteurs et des destinations que par leur nombre.

Les auteurs de [27, 28] ont seulement considéré le *clustering* global dans leur étude sur les modèles de graphe. La courbe moyenne des coefficients de *clustering* (figure 4.9, droite) présente une uniformité du gris similaire à celle des modèles qu'ils ont utilisés. Nos observations sur des données réelles confirment donc les résultats de leurs simulations.

[4] C'est ce que nous avons observé pour tous les ordres aléatoires que nous avons générés.

4.4.3 Distance moyenne

La distance moyenne fait partie des propriétés les plus utilisées pour décrire la topologie de l'internet et plus généralement les graphes de terrain. Cependant, le calcul de la distance moyenne a un coût très élevé. Nous utilisons ici une heuristique proposée dans [36] pour approximer la distance moyenne. À l'étape i du processus, nous choisissons de manière aléatoire un nœud v_i et nous calculons $d(v_i)$, sa distance moyenne à tous les autres nœuds, avec une complexité linéaire en temps $O(m)$ et en espace $O(n)$. Alors la i-ème approximation de la distance moyenne est

$$D_i = \frac{1}{i} \sum_{j=1}^{i} d(v_j).$$

L'itération s'arrête dès que les variations dans l'estimation sont plus petites qu'un nombre ϵ pendant au moins i_{min} étapes, *i.e.* $|D_{k+1} - D_k| < \epsilon$, pour tout k, $i - i_{min} \leqslant k < i$. Les variables i_{min} et ϵ sont des paramètres de contrôle qui assurent qu'au moins i_{min} itérations seront exécutées et que les variations dans les i_{min} dernières itérations ne sont pas supérieures à ϵ. Nous utilisons ici $i_{min} = 1$ et $\epsilon = 0.1$

La figure 4.10 présente les courbes en niveau de gris pour deux ordres différents sur les moniteurs et les destinations et la moyenne sur dix ordres.

Quand on utilise peu de moniteurs et de destinations, on obtient une surestimation de la distance moyenne du graphe complet. L'estimation devient rapidement plus précise quand le nombre de moniteurs et de destinations augmente. Cette observation peut-être comprise de la façon suivante : avec peu de moniteurs le graphe est presque un arbre, et la distance moyenne est donc surestimée. Ceci change relativement vite quand on augmente le nombre de moniteurs et de destinations.

FIG. 4.10 – Distance moyenne. Gauche et milieu : deux ordres aléatoires sur les moniteurs et les destinations. Droite : moyenne de 10 ordres aléatoires.

Contrairement à d'autres propriétés, l'impact de l'ordre des moniteurs et des destinations sur la distance moyenne est plutôt faible. Il y a beaucoup de similarités entre les résultats pour différents ordres. On n'observe pas de différence significative entre les différents ordres aléatoires, et la moyenne de 10 ordres. Cela montre que le choix des moniteurs et des destinations a un faible impact sur la distance moyenne observée.

Nous observons sur les différentes courbes des fluctuations du niveau de gris quand le nombre de moniteurs et de destinations est petit. Le gris devient très vite uniforme lorsqu'on atteint un certain nombre de moniteurs et de destinations. À partir de ce nombre, la distance moyenne ne varie plus beaucoup quand on augmente le nombre de moniteurs et de destinations et est donc rapidement bien estimée.

En conclusion, nous avons en général une bonne estimation de la distance moyenne. Elle devient plus précise dès qu'on considère un petit nombre de moniteurs et de destinations, et cela quel que soit l'ordre considéré. Le choix des moniteurs et des destinations n'a pas d'influence nette sur l'estimation de la distance moyenne.

C'est le même résultat qui est observé avec presque tous les modèles utilisés dans [27, 28].

4.4.4 Distribution des degrés

Nous allons maintenant étudier la distribution des degrés des nœuds. La figure 4.11 présente la distribution des degrés du graphe complet. Elle montre une distribution des degrés hétérogène. Dans cette section, nous allons étudier l'influence des moniteurs et des destinations sur son estimation.

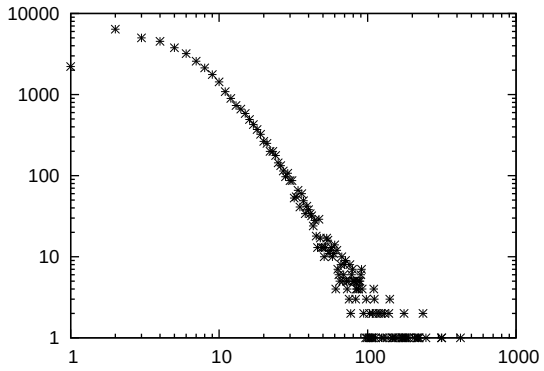


FIG. 4.11 – Distribution des degrés du graphe complet.

Nous allons tout d'abord montrer l'impact du choix des moniteurs et des destinations sur la distribution des degrés qu'on observe. La figure 4.12 montre la distribution des degrés de différentes vues $G_{S'D'}$ pour des nombres différents $s = |S'|$ de moniteurs et $d = |D'|$ de destinations. Pour chaque taille, on présente trois distributions. Les trois distributions correspondent à trois choix aléatoires des moniteurs et des destinations.

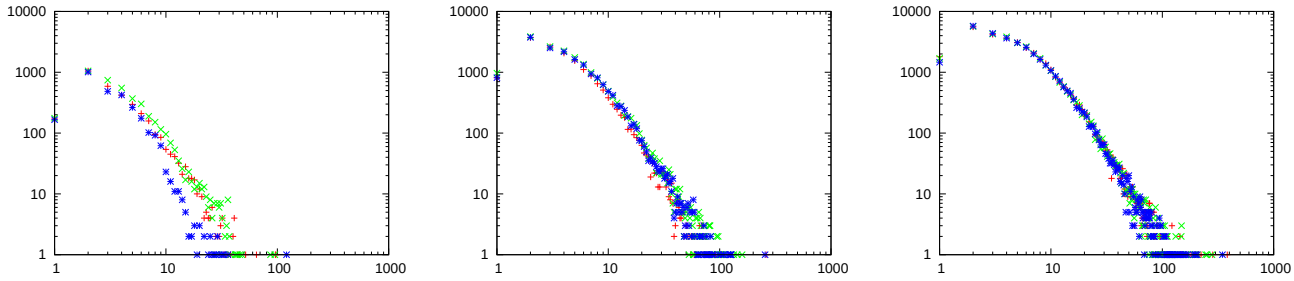


FIG. 4.12 – Impact du choix des moniteurs et des destinations sur la distribution des degrés. Chaque figure présente 3 distributions des degrés correspondant à différents choix de moniteurs et de destinations. Gauche : S' contient 18% des moniteurs et D' contient 6% des destinations. Milieu : 45% pour S' et 26% pour D'. Droite : 81% pour S' et 67% pour D'.

On peut voir que pour des ensembles S' et D' petits (figure 4.12 gauche), la différence entre les distributions est perceptible pour les degrés supérieurs à 10 (on observe une courbe qui s'écarte un peu des deux autres), tandis qu'elles coïncident pratiquement pour les degrés plus petits (par exemple

les nœuds de degrés 1 et 2). Mais on observe un comportement différent quand les ensembles S' et D' sont assez grands. Les trois distributions tendent à se superposer quand le nombre de moniteurs et de destinations augmente, voir figure 4.12 (milieu et droite).

Ces courbes montrent que le choix des moniteurs et des destinations n'a qu'une influence mineure sur la distribution des degrés observée.

Dans le but d'étudier l'impact du nombre de moniteurs et de destinations sur la distribution des degrés, nous allons faire varier séparément le nombre de moniteurs et de destinations. Étant donné qu'on a vu ci-dessus que le choix des moniteurs et des destinations n'a pas une influence significative sur la distribution des degrés, nous allons donc considérer différents nombres de moniteurs et de destinations sans nous préoccuper de quels moniteurs et destinations sont considérés.

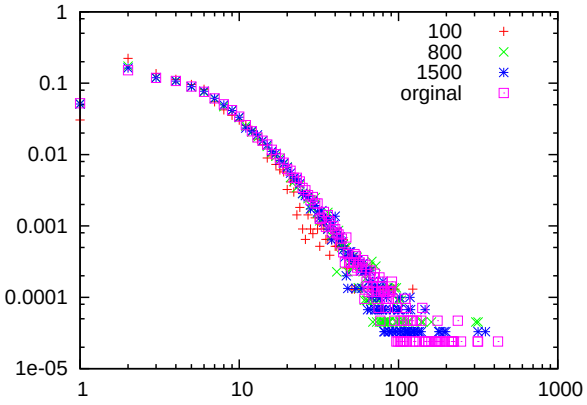


FIG. 4.13 – Impact du nombre de destinations sur la distribution des degrés. Le nombre de destinations $d = |D'|$ varie de 100 à 3 000 ; le nombre de sources $s = |S|$ ne change pas.

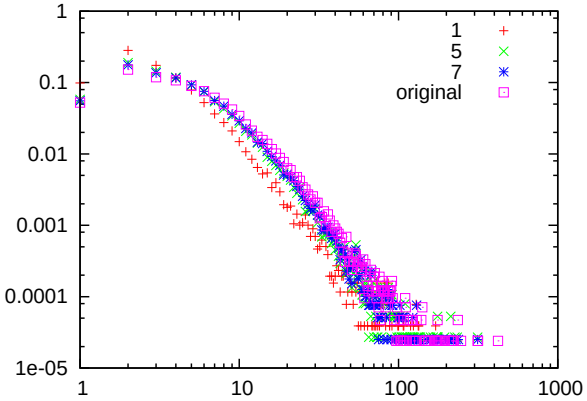


FIG. 4.14 – Impact du nombre de moniteurs sur la distribution des degrés. Le nombre de moniteurs $s = |S'|$ varie de 1 à 11 ; le nombre de destinations $d = |D|$ ne change pas.

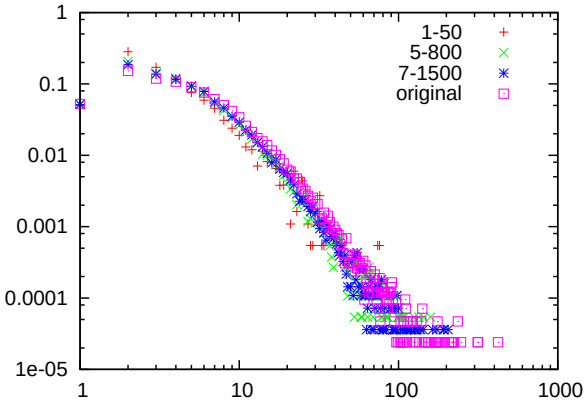


FIG. 4.15 – Impact du nombre de moniteurs et de destinations sur la distribution des degrés. Le nombre de moniteurs s et de destinations d varient en même temps.

La remarque globale que l'on peut faire en observant les trois courbes de la figure 4.13 est qu'il n'y pas une grande différence entre les différentes distributions malgré les écarts importants entre les nombres de moniteurs et de destinations considérés. Ces observations confirment celles qui sont faites dans [27, 28] par simulations.

Les distributions de la figure 4.13 se superposent plus parfaitement que celles des figures 4.14 et 4.15. Elle montre que la distribution des degrés est plus précise quand le nombre de moniteurs est élevé. On peut voir que la variation du nombre de destinations ne change pas significativement la distribution : même avec seulement 100 destinations (soit $3,4\%$ environ) la différence entre l'estimation et l'original n'est pas significative.

Quand on observe les courbes des figures 4.14 et 4.15 on peut discerner une convergence vers la distribution du graphe complet. Les observations faites ci-dessus montrent que le nombre de moniteurs joue un grand rôle dans cette convergence. Le nombre de destinations n'a pratiquement pas d'influence sur la distribution.

En résumé, on peut dire qu'on a une bonne estimation de la distribution des degrés même avec un petit nombre de moniteurs et de destinations, et elle est encore plus précise quand le nombre de moniteurs est plus grand.

Il est important de noter que dans tous les cas, on observe une distribution hétérogène. C'est aussi ce qu'observent les auteurs de [27, 28] avec des modèles de graphes ayant une distribution des degrés en loi de puissance [3, 41, 42].

4.5 Conclusion

Nous avons présenté ici plusieurs expériences dans le but d'évaluer la mesure de la topologie par exploration distribuée. Pour cela, nous avons utilisé les données du chapitre 2 que nous avons décrites en détail en montrant les différentes structures utilisées. Contrairement aux travaux précédents effectués en ce sens [27, 28], nous nous sommes appuyés sur des données réelles et non sur des simulations.

Ensuite nous avons présenté une méthodologie d'évaluation de l'exploration distribuée déjà pratiquée sur des modèles de graphe [27, 28].

Dans le but d'étudier l'impact du choix des destinations et des moniteurs, nous avons considéré l'ordre dans les ensembles de moniteurs et de destinations. Nous avons montré en étudiant plusieurs échantillons que l'influence de l'ordre est significative et devait être prise en compte dans la méthodologie et pour cela nous avons entrepris de considérer la moyenne. Nous avons montré que les propriétés statistiques ne sont pas toutes influencées de la même manière : certaines propriétés comme la distribution des degrés sont robustes et semblent ne pas dépendre fortement du choix des moniteurs et destinations ou de leur nombre. Par contre, d'autres propriétés comme le coefficient de *clustering* en dépendent significativement.

On retrouve pour certaines propriétés comme le degré moyen les mêmes observations que celles faites sur des modèles [27, 28]. Par contre pour des propriétés comme le coefficient de *clustering* global, il est très difficile de conclure sur une comparaison avec les différents modèles. Néanmoins, la moyenne présente quelques similitudes avec certains modèles de graphes aléatoires avec un coefficient de clustering élevé [25, 27], et une différence nette avec d'autres comme le graphe aléatoire [21].

Nous envisageons de continuer l'étude par d'autres propriétés de la topologie de l'internet comme la centralité, le diamètre, etc. Par ailleurs, nous avons utilisé peu de moniteurs par rapport au nombre de destinations. Ceci est justifié par le fait qu'en général l'exploration distribuée est faite avec beaucoup plus de destinations que de moniteurs, mais il serait intéressant d'avoir un plus grand nombre de moniteurs dans l'étude de l'exploration distribuée.

Conclusion et perspectives

Nous avons abordé dans cette thèse la problématique de l'étude de la dynamique de l'internet, plus particulièrement au niveau IP de la topologie. Pour cela, nous avons introduit une nouvelle approche : en nous focalisant sur la portion de la topologie qu'une machine observe autour d'elle (qui est plus simple et rapide à mesurer ce qui rend facile de répéter la mesure) que nous appelons *vue ego-centréee*, nous sommes arrivés à capturer une dynamique significative.

Pour cela, nous avons conçu un outil appelé `tracetree`, qui permet de mesurer avec efficacité cette portion de la topologie. Nous avons mené une campagne de mesure de type *radar* qui consiste en une répétition périodique de mesures `tracetree`, à partir d'une centaine de machines pendant plusieurs semaines et avec une fréquence élevée. Ces données, ainsi que l'outil, sont fournies à la communauté, ce qui est une contribution importante en soi. Nous avons ensuite effectué des analyses préliminaires des données qui ont montré qu'elles permettent de détecter des propriétés non triviales de la dynamique.

Nous nous sommes ensuite intéressés à la dynamique des vues ego-centrées. Nous avons mis en évidence un fait remarquable : les moniteurs découvrent continuellement des adresses IP à un rythme nettement plus élevé que ce à quoi l'on s'attendait. Nous avons étudié ce phénomène en détail et envisagé plusieurs causes possibles. Nous avons montré que la dynamique de routage joue un grand rôle dans ces observations : la plupart des adresses IP qui sont découvertes au fil de la mesure existaient depuis le début de la mesure et sont devenues visibles suite à des changements de routage. Nous n'avons cependant pas obtenu d'explication formelle pour ce phénomène.

Enfin, nous avons étudié l'avantage d'une méthode de mesure distribuée de la topologie de l'internet. Nous avons pour cela utilisé les données radar que nous avons récoltées. L'approche utilisée consiste à étudier comment la topologie mesurée évolue quand on fait varier le nombre de moniteurs et de destinations. Nous avons montré que certaines propriétés ne sont pas automatiquement mieux estimées quand on augmente le nombre de moniteurs et de destinations. Nous avons aussi vu que le choix des ensembles de moniteurs et de destinations peut influencer l'estimation de certaines propriétés. Nous avons montré que toutes les propriétés ne sont pas influencées au même degré par le nombre et le choix des moniteurs et des destinations. Par exemple, certaines sont quasiment indépendantes du nombre de moniteurs et de destinations et d'autres le deviennent uniquement à partir d'une certaine taille de l'ensemble des moniteurs et des destinations. Nos travaux ont permis de confirmer sur des données réelles certains résultats qui ont été établis à partir de modèles.

Au final, l'approche ego-centrée que nous avons introduite a permis de collecter des données sur la dynamique de la topologie de l'internet.

Nous avons recueilli d'importantes informations sur la dynamique à travers l'analyse de ce qu'observent les moniteurs, ce qui a permis de tirer une certaine caractérisation de la dynamique.

L'analyse des données a montré que la topologie de l'internet évolue beaucoup plus vite que ce à quoi on pouvait s'attendre. Ceci est en soi un résultat important. De plus, cela a d'importantes

conséquences sur la mesure de la topologie : l'objet mesuré n'est pas le même au début et la fin de la mesure. Ceci joue un rôle dans la métrologie. Nos travaux indiquent que pour étudier le biais introduit par la mesure, il faut non seulement prendre en compte le nombre de moniteurs et de destinations mais également leur choix, ainsi que le fait que la topologie évolue pendant la mesure.

Notre travail ouvre de nombreuses perspectives. Nous proposons une synthèse des différents aspects qui sont à nos yeux les plus pertinents et prometteurs, sans prétendre en faire une liste exhaustive.

Naturellement l'essentiel des perspectives concerne l'analyse des données récoltées qui constitue une suite logique de notre travail. Les données que nous avons obtenues contiennent d'importantes informations sur la dynamique de l'internet et plus généralement de sa topologie. Bien que nous ayons entamé au chapitre 3 une caractérisation de la dynamique, il reste beaucoup de travail à faire dans ce sens.

Nous avons vu au chapitre 2 que de façon générale les moniteurs que nous avons étudiés se comportaient de manière globalement similaire. Mais on a aussi observé quelques différences en fonction des moniteurs. On se pose alors des questions sur l'uniformité de la dynamique : l'internet semble ne pas avoir la même dynamique partout. Il est donc important d'approfondir de telles questions par des analyses poussées pour mieux comprendre la dynamique de l'internet.

Les méthodes de détection d'événements présentées au chapitre 2 ont permis de montrer qu'il est possible de détecter des événements d'importance majeure sur internet. Cependant ces méthodes restent préliminaires. Il faudrait les améliorer ou trouver d'autres approches plus formelles. De plus, un événement important peut être vu différemment par les moniteurs. Par exemple, certains moniteurs peuvent l'observer comme un simple événement de moindre importance. Il serait intéressant d'étudier de quelle manière un même événement est observé par différents moniteurs. Cela permettra de caractériser certains événements et permettre de développer des méthodes automatiques de détection.

Dans l'optique de trouver une explication à la découverte continuelle d'adresses IP étudiée dans le chapitre 3, nous envisageons de reproduire le phénomène par simulation. Dans ce cas, où nous avons un contrôle sur tous les paramètres, nous pouvons déterminer exactement les causes de ce phénomène. Pour cela, nous avons besoin d'utiliser des graphes dynamiques issus de modèles, sur lesquels nous allons simuler une mesure radar. Il faut noter cependant qu'il n'existe pas actuellement de modèle consensuel de la dynamique de l'internet au niveau IP. Nous comptons reproduire le phénomène par des modèles simples pour essayer d'en comprendre les causes.

Une perspective complémentaire à la précédente est la modélisation de la dynamique de l'internet. La caractérisation de la dynamique que nous avons entamée pourra à long terme permettre de proposer un modèle de graphe dynamique de la topologie de l'internet.

Nous avons montré qu'il n'existe pas de solution parfaite pour obtenir une carte de la topologie de l'internet : en une seule passe de mesure un moniteur ne peut pas découvrir tout ce qu'il peut voir autour de lui, et fusionner les données de plusieurs passes conduit à obtenir une carte de la topologie avec des données obsolètes. Trouver un meilleur compromis permettant de construire une carte plus précise serait d'un grand apport dans le domaine. Par exemple, en caractérisant les nœuds et liens selon leur durée d'apparition, on pourrait par le nombre et la fréquence des passes de mesure définir des intervalles de confiance pour la mesure. Ceci pourra permettre d'augmenter la précision de la carte.

D'une manière générale, les graphes de terrain sont de nature dynamique et des efforts ont été faits pour étudier cela. Il y a donc des travaux sur l'analyse de leur dynamique, mais la plupart se sont portés sur des cas particuliers. Il y a très peu de choses générales concernant l'étude de

la dynamique. Les observations et questions que nous avons soulévées dans cette étude sont tout aussi pertinentes pour d'autres graphes de terrain dynamiques. Transposer les analyses que nous avons faites sur d'autres graphes dynamiques permettrait donc d'entamer facilement leur analyse. Cela permettrait également de mettre en perspective ce que nous avons fait et pourrait apporter un éclairage nouveau sur le sujet.

Bibliographie

[1] D. Achlioptas, A. Clauset, D. Kempe, and C. Moore. On the bias of traceroute sampling : or, power-law degree distributions in regular graphs. In *ACM Symposion on Theory Of Computing (STOC)*, 2005.

[2] D. Achlioptas, A. Clauset, D. Kempe, and C. Moore. On the bias of traceroute sampling : or, power-law degree distributions in regular graphs. *To appear in journal of ACM*, 2009.

[3] R. Albert and A.-L. Barabási. Emergence of scaling in random networks. *Science*, 286 :509–512, 1999.

[4] R. Albert and A.-L. Barabási. Topology of evolving networks : local events and universality. *Physical Review Letters*, 85(24) :5234–5237, 2000.

[5] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with paris traceroute. In *Internet Measurement Conference*, pages 153–158, 2006.

[6] B. Augustin, T. Friedman, and R. Teixeira. Measuring load-balanced paths in the internet. In *SIGCOMM*, October 2007.

[7] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286 :509–512, 1999.

[8] P. Barford, A. Bestavros, J. Byers, and M. Crovella. On the marginal utility of network topology measurements. In *ACM/SIGCOMM IMC*, 2001.

[9] CAIDA – Skitter project. `http://www.caida.org/tools/measurement/skitter/`.

[10] R. Callon. Use of OSI IS-IS for Routing in TCP/IP and Dual Environments, IETF RFC 1195. 1990 (December).

[11] A. Carmignani, G. Di Basttista, W. Didimo, F. Matera, and M. Pizzonia. Visualization of the autonomous systems interconnections with hermes. *Proc. Graph Drawing*, pages 150–163, 2000.

[12] H. Chang, S. Jamin, and W. Willinger. Internet connectivity at the AS-level : an optimization-driven modeling approach. In *Proc. ACM SIGCOMM MoMeTools workshop*, 2003.

[13] H. Chang, S. Jamin, and W. Willinger. To peer or not to peer : modeling the evolution of the Internet's AS-level topology. In *Proc. of IEEE INFOCOM*, 2006.

[14] 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.

[15] A. Clauset and C. Moore. Accuracy and scaling phenomena in internet mapping. *Phys. Rev. Lett*, 2005.

[16] L. Dall'Asta, I. Alvarez-Hamelin, A. Barrat, A. Vázquez, and A. Vespignani. Exploring networks with traceroute-like probes : Theory and simulations. *Theoretical Computer Science (TCS)*, 355(1) :6–24, 2006.

[17] L. Dall'Asta, J.I. Alvarez-Hamelin, A. Barrat, A. Vzquez, and A. Vespignan. A statistical approach to the traceroute-like exploration of networks : theory and simulations. In *Workshop on combinatorial and Algorithmic Aspects of Networking*, 2004.

[18] Dimes. `http://www.netdimes.org/new/`.

[19] B. Donnet and T. Friedman. Internet topology discovery : a survey. *IEEE Communications Surveys and Tutorials*, 2007. to appear.

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

[21] P. Erdös and A. rényi. On random graphs. *Publ. Math. Debrecen*, 6 :290–297, 1959.

[22] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *ACM SIGCOMM*, 1999.

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

[24] R. Govindan and H. Tangmunarunkit. Heuristics for internet map discovery. In *IEEE INFOCOM*, volume 85, pages 1371–1380, March 2000.

[25] J.-L. Guillaume and M. Latapy. Bipartite graphs as models of complex networks. In *Workshop on Combinatorial and algorithmic Aspects of Networking (CAAN)*, 2004.

[26] J.-L. Guillaume and M. Latapy. Complex networks metrology. In *Complex systems*, 2005.

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

[28] J.-L. Guillaume, M. Latapy, and D. Magoni. Relevance of massively distributed explorations of the internet topology : qualitative results. *Computer Networks*, 50(16) :3197–3224, 2006.

[29] B. Huffaker, M. Fomenkov, D. Moore, D. Plummer, and K. Clally. Distance metrics in the internet. In *IEEE International Telecommunication Symposium (ITS) Brasil*, September 2002.

[30] Y. Hyun, A. Broido, and K. Claffy. Traceroute and BGP AS path incongruities. `http://www.caida.org/outreach/papers/2003/ASP/`.

[31] V. Jacobson. "taceroute", 1989. `ftp://ftp.ee.lbl.gov/traceroute.tar.gz`.

[32] C. Labovitz, G. Robert Malan, and F. Jahanian. Origins of internet routing instability. In *Proc. IEEE INFOCOM*, pages 218–226, 1999.

[33] M. Lad, D. Massey, and L. Zhang. Visualizing internet routing changes. *IEEE Transactions on Visualization and Computer Graphics, special issue on Visual Analytics*, 2006.

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

[35] M. Latapy. Grands graphes de terrain : mesure et métrologie, analyse, modélisation, algorithmique. In *Mémoire d'HDR Université Pierre et Marie Curie (UPMC-Paris 6)*, 2007.

[36] M. Latapy and C. Magnien. Measuring fundamental properties of real-world complex networks. In *Proc. IEEE INFOCOM*, 2008.

[37] M. Latapy, C. Magnien, and F. Ouédraogo. A radar for the internet. In *Proc. first International Workshop on Analysis of Dynamic Networks (ADN), in conjunction with IEEE ICDM 2008*, 2008.

[38] L. Li, D. Alderson, W. Willinger, and J. Doyle. A first-principles approach to understanding the internet's router-level topology. In *ACM SIGCOMM*, 2004.

[39] C. Magnien, F. Ouédraogo, G. Valadon, and M. Latapy. Fast dynamics in internet topology : preliminary observations and explanations. In *Proc. Fourth International Conference on Internet Monitoring and protection (ICIMP)*, 2009.

[40] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat. Systematic topology analysis and generation using degree correlations. In *Proc. of ACM SIGCOMM*, 2006.

[41] M. Molloy and B. Reed. A critical point for random graphs with a given degree sequence. *Random Structures and Algorithms*, pages 161–179, 1995.

[42] M. Molloy and B. Reed. The size of the giant component of a random graph with a given degree sequence. *Combin. Probab. Comput.*, pages 295–305, 1998.

[43] T. Moors. Streamlining traceroute by estimating path lengths. In *Proc. IEEE Workshop on IP Operations and Management*, October 2004.

[44] J. Moy. OSPF Version 2, IETF RFC 2328. 1998 (April).

[45] NLANR. National laboratory for applied network research. `http://www.nlanr.net/`.

[46] R. Oliveira, B. Zhang, and L. Zhang. Observing the evolution of internet AS topology. In *ACM SIGCOMM*, 2007.

[47] V. N. Padmanabhan and L. Subramanian. An investigation of geographic mapping techniques for internet hosts. In *SIGCOMM*, 2001.

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

[49] Paris Traceroute. `http://www.paris-traceroute.net/`.

[50] 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 *Proc. IEEE Infocom*, 2003.

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

[52] R. Pastor-Satorras and A. Vespignani. Evolution and structure of the internet. In *Cambridge University Press*, 2004.

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

[54] T. Petermann and P. De Los Rios. Exploration of scale-free networks. In *Eur. Phys. J.B*, 2004.

[55] PlanetLab Consortium. Projet PlanetLab, 2002. Voir `http://www.planet-lab.org/`.

[56] J. Postel. Internet protocol. RFC 791, September 1981.

[57] Radar. Programmes et donnée. `http://www-rp.lip6.fr/~latapy/Radar/`.

[58] P. De Los Rios. Exploration bias of complex networks. In *Proceedings of the 7th conference on Statistical and Computational Physics Granada*, 2002.

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

[60] Y. Shavitt and U. Weinsberg. Quantifying the importance of vantage points distribution in internet topology measurements. In *Proc. IEEE INFOCOM*, Avril. 2009.

[61] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with rocketfuel. In *Proc. of ACM/SIGCOMM*, August 2002.

[62] Tcptraceroute project. `http://michael.toren.net/code/tcptraceroute/`.

[63] Team Cymru. `http://www.team-cymru.org/`.

[64] University of Oregon. Route Views, University of Oregon Route Views project. `http://www.antc.uoregon.edu/route-views/`.

[65] D. Veitch, B. Augustin, R. Teixeira, and T. Friedman. Failure control in multipath route tracing. In *Proc. IEEE INFOCOM*, Avril. 2009.

[66] F. Viger, B. Augustin, X. Cuvellier, B. Orgogozo, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Traceroute anomalies : detection and prevention in internet graphs. *Computer Networks*, 52, 2008.

[67] F. Wang, N. Feamster, and L. 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.

[68] X. Wang and D. Loguinov. Wealth-based evolution model for the internet's AS-level topology. In *Proc. of IEEE INFOCOM*, 2006.

[69] D. J. Watts and S. H. Strogatz. Collective dynamics of smallworld networks. *Nature*, 393 :440–442, 1998.

[70] J. Xia, L. Gao, and T. Fei. Flooding attacks by exploiting persistent forwarding loops. In *Proc. ACM SIGCOMM Internet Measurement Conference*, 2005.