UNIVERSITÉ PIERRE ET MARIE CURIE - PARIS 6
UFR D'INFORMATIQUE

THÈSE DE DOCTORAT

spécialité : INFORMATIQUE

présentée par
Fabien VIGER

Contributions à la mesure et à la modélisation de la topologie de l'Internet

Thèse encadrée par Matthieu LATAPY

soutenue le 13 septembre 2007 devant le jury composé de :

Serge FDIDA	co-directeur
Philippe JACQUET	examinateur
Jean-Claude KÖNIG	rapporteur
Matthieu LATAPY	co-directeur
Jean-Jacques PANSIOT	rapporteur
Christophe PRIEUR	examinateur
Michèle SORIA	examinateur

Remerciements

Ces trois années de thèse, en continuité avec le stage de DEA, furent fort diversifiées. J'ai pu toucher à tout, participer à des projets de différentes envergures, découvrir le milieu académique et l'enseignement, apprendre à travailler seul et en équipe, à échanger et collaborer avec des scientifiques d'autres disciplines, et beaucoup d'autres choses. Au-delà de la modeste contribution scientifique, ma thèse fut une formidable expérience formatrice.

Je tiens à remercier Matthieu Latapy pour m'avoir encadré d'excellente façon tout au long de cette expérience. Ses conseils et les discussions que nous avons pu avoir furent aussi agréables qu'instructifs. Il a bien souvent dépassé le cadre de simple encadrant scientifique et a toujours été présent quand j'avais besoin de lui. C'est également lui qui m'a encouragé à diversifier mon activité et je lui en suis reconnaissant. Je souhaite à tout thésard de trouver un directeur comme lui.

Je remercie également Serge Fdida pour m'avoir accueilli dans son équipe et qui a pris en charge dans un premier temps la responsabilité officielle de mon encadrement.

Le manuscrit de cette thèse n'a été finalisé qu'en juillet, et je remercie particulièrement mes deux rapporteurs, Jean-Claude König et Jean-Jacques Pansiot, pour avoir consacré leur temps et leur attention à sa lecture en plein mois d'août, et pour m'avoir fourni de précieux commentaires. Il en va de même pour Philippe Jacquet, Christophe Prieur et Michèle Soria pour avoir accepté de faire partie de mon jury.

Mes collaborateurs directs ont tenu une place importante. Il est difficile de tout résumer en quelques lignes, mais je tiens à les remercier pour tous ce qu'ils m'ont apporté, et notamment Clémence Magnien pour sa bonne humeur, sa compagnie en période de deadline, et ses nombreuses relectures de ma prose, Benjamin Orgogozo pour son caractère, ses présentations et sa compagnie, Alain Barrat pour son réalisme scientifique et ses questions qui font mouche, Eric Kolaczyk pour son ouverture et son esprit de synthèse, Brice Augustin pour les échanges d'opinions et d'expériences, Renata Teixera pour son expertise et sa disponibilité, Timur Friedman pour ses qualités de coordinateur, et aussi Xavier Cuvellier, Luca Dall'Asta, Cun-Hui Zhang, et Damien Bobillot. J'exprime également une gratitude toute spéciale à Guillaume Goron pour m'avoir beaucoup appris en termes d'encadrement.

Mes collègues du bureau 6A51 et leur bonne humeur ont contribué grandement à faire des journées au bureau un plaisir, et leur seule présence représentait souvent une excellente motivation à rester travailler plus souvent ou plus longtemps. Dans le désordre, je remercie Pascal et son lutin facétieux, Clémence et ses cris, Vincent et ses blagues, Benjamin et sa béarnaise, Mahendra qui fait tout pour être geek, Jean-Loup et son sourire

inimitable, Mohssen et sa gentillesse, Nicolas et son bronzage, les deux Philippe et leur sympathie, Jean-Philippe et ses citations, Anne-Ruxandra et sa fraîcheur, Michel et ses horaires incroyables et Mohamed et ses petits bruits.

Hors du cadre du travail, je tiens à remercier mes parents, qui m'ont toujours soutenu, et mes soeurs Cécile, Agnès et Marion ainsi que mes amis, qui ont contribué à rendre ces trois années aussi agréables. Dans le désordre : Vanessa, Fred, Jip, Pascal, Tristan, Charles, Titi, Jul, Rémi, Oliv, Gog, Adrian, Elisa, Jen, Clarisse, Anne-Rux, Virginie, Karine, Johanna, Felix, Joël, Audrey, mais aussi tous les volleyeurs, le car 4, les joueurs de poker, et tous les autres.

Table des matières

Introduction

Le rôle majeur joué par l'Internet dans le monde d'aujourd'hui le place au centre de nombreuses recherches. Son gigantisme, sa complexité et sa dynamique nous amènent à le considérer non plus comme un simple *outil* à améliorer, mais comme un *objet d'étude* à part entière.

Dans cette thèse, nous nous intéressons à la *topologie* de l'Internet, c'est-à-dire la vision très réduite et abstraite du réseau comme un ensemble de *nœuds* connectés entre eux par des *liens*. Les premiers correspondent en première approximation aux routeurs qui font transiter l'information, les seconds aux câbles ou autres liens physiques ou logiques transmettant l'information d'un routeur à l'autre. Bien que cette topologie ne soit qu'une image très épurée du réseau, elle capture certaines de ses propriétés fondamentales. Sous forme de *graphe*, elle devient plus facile à décrire et à manipuler dans un contexte théorique.

La mesure et la modélisation de la topologie de l'Internet soulèvent de nombreux problèmes difficiles, notamment liés au difficile mariage de la théorie et de la pratique : l'objet étudié est si complexe que toute manipulation, mesure, modélisation passe forcément par une simplification. Cette démarche simplificatrice est difficile à bien mener : garder trop de complexité peut rendre la situation inextricable, alors qu'à l'inverse une vision trop simplifiée peut donner des résultats inadéquats.

Nous abordons dans cette thèse plusieurs questions : comment modéliser la topologie de l'Internet ? plus précisément, comment générer des graphes, sortes d'images virtuelles de l'Internet, sans s'écarter trop de la réalité ? Comment savoir si la mesure partielle et biaisée de l'Internet dont on dispose représente bien la réalité, et comment peut-on la corriger ? Nous tentons d'apporter des éléments de réponse, apportant à chaque fois notre pierre à un vaste édifice dépassant largement le cadre de cette thèse. Le fil directeur et l'originalité de notre contribution reste le rapprochement entre les visions abstraites et concrètes de l'Internet : nous cherchons à rendre les premières plus réalistes, et à donner un cadre plus rigoureux aux dernières.

Le premier chapitre de cette thèse permettra au lecteur de se familiariser avec les objets étudiés, le vocabulaire utilisé et l'état de l'art. Les chapitres suivants rentrent dans le vif du sujet et abordent le détail de nos contributions scientifiques. Le Chapitre 2 traite de la modélisation et introduit un modèle simple et rigoureux que nous rendons applicable à des graphes dont la taille atteint le gigantisme de l'Internet. Ensuite, le Chapitre 3 s'intéresse à la métrologie de la topologie de l'Internet, et fournit des solutions pour évaluer certaines propriétés de l'Internet à partir de mesures incomplètes. Enfin, le Chapitre 4 présente nos

travaux sur la correction de la mesure, et plus précisément de l'outil `traceroute`, dont nous étudions les défauts et pour lequel nous proposons une version améliorée.

Chapitre 1

Contexte et positionnement

Cette thèse aborde plusieurs sujets venus de contextes scientifiques différents, mais s'articulant autour de la mesure et de la modélisation de l'Internet. Avant de rentrer dans le vif du sujet avec les chapitres suivants, nous introduisons ici les notions que le lecteur doit connaître préalablement, et mettons en évidence les motivations sous-jacentes à l'étude du réseau Internet en général et à sa mesure et modélisation en particulier.

1.1 Complexité de l'Internet

1.1.1 Quelques bases techniques

L'Internet comme infrastructure physique est perçu de manière assez floue par le grand public dont la connaissance se limite généralement aux noms des principaux fournisseurs d'accès, sans forcément savoir que l'Internet permet – en théorie – à tout couple d'ordinateurs qui y sont reliés, quelle que soit leur position dans le monde, de communiquer.

Chaque ordinateur connecté sur l'Internet est censé posséder une adresse `IP` de 32 bits, souvent représentée comme 4 octets qui sont des nombres entre 0 et 255 (voir Figure 1.1). Cette adresse lui est propre et permet donc de l'identifier. Le sigle IP tient pour *Internet Protocol*, et spécifie – entre autres – la manière d'encapsuler des données avant de les envoyer sur l'Internet. Lorsque des données sont transmises sur l'Internet, elles sont subdivisées en *paquets*, de tailles limitées (le plus souvent inférieures à 1500 octets). Chaque paquet, avant d'être envoyé sur le réseau, doit se voir accoler un en-tête `IP` qui consiste en quelques octets d'informations sur le paquet, dont notamment l'adresse `IP` du destinataire, qu'on désignera dans la suite par *adresse destination*.

L'utilisation de l'Internet en tant que moyen de communication ne nécessite donc pas plus d'informations que l'adresse destination. De ce point de vue, l'Internet n'est qu'une boîte noire reliant entre eux des ordinateurs et permettant à chacun d'envoyer des données à n'importe quel autre. Cette vision de l'Internet, schématisée dans la Figure 1.1, comme une entité unique au comportement simple ne reflète que sa *fonction*, et pas sa *constitution*.

Dans cette partie, et plus globalement dans cette thèse, nous nous intéressons à l'In-

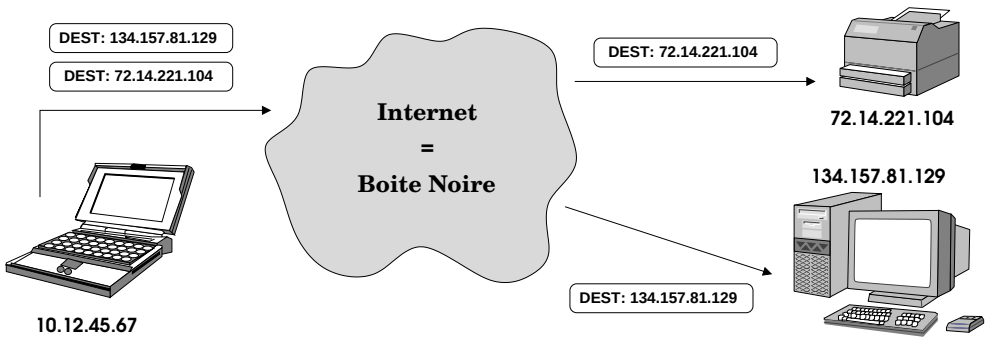

FIG. 1.1 – L'Internet, vu comme une boîte noire

ternet en lui-même, autrement dit aux *infrastructures* permettant une telle transmission des données : comment fonctionnent-t-elles ? de quoi sont-elles constituées ? et en décrivant le réseau de manière plus approfondie, nous verrons apparaître plusieurs niveaux de complexité.

1.1.2 Une structure décentralisée : les AS, ou systèmes autonomes

L'Internet n'est pas la propriété d'une organisation unique. Il n'est en fait que l'interconnexion de parties autonomes du réseau, appelées AS (pour le sigle anglophone *Autonomous System*), gérées par des entreprises ou des consortiums. Un AS est connecté à d'autres AS par l'intermédiaire de routeurs de bordure, appelés *passerelles*, parfois co-gérées par les AS qu'elles connectent, comme esquissé sur la Figure 1.2. L'AS lui-même est constitué d'un ensemble de *routeurs* reliés en réseau. Lorsqu'un paquet traverse un AS, il est relayé de routeur à routeur. Les fournisseurs d'accès à l'Internet sont des AS particuliers, en ce qu'ils sont directement reliés à un certain nombre d'*utilisateurs finaux* (particuliers, entreprises ou autres entités n'étant pas eux-mêmes des AS).

Chaque AS établit un contrat pour la transmission des données avec ses voisins, c'est-à-dire les AS ou utilisateurs finaux auxquels il est directement relié. Considérons deux AS A et B reliés entre eux. On trouvera 2 principaux types de contrats :

- A est *client* de B (et B est alors *fournisseur* de A) : A paye B pour le trafic transitant de A à B et de B à A.
- A et B on un contrat de *peering* : le trafic peut circuler librement entre A et B. Ce type de libre échange peut être soumis à condition et par exemple être restreint aux transits dont la *destination* est gérée par A ou B.

D'autres cas de figure peuvent se présenter. Par exemple on peut imaginer que A paye B pour le trafic sortant (de A vers B) et fasse en même temps payer B pour le trafic entrant (de B vers A). Mais les 3 cas cités ci-dessus constituent la majorité des accords actuels [19]. Ajoutons que l'ensemble des contrats passés par un AS n'est pas nécessairement public : la plupart des AS ne divulguent pas cette information.

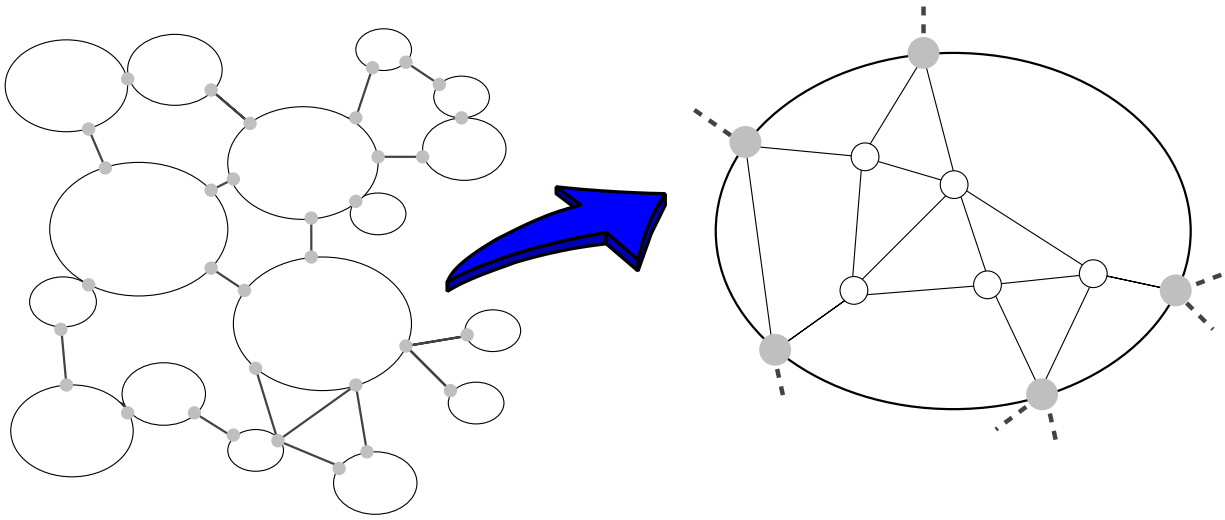

FIG. 1.2 – Schéma représentant une interconnexion d'AS (grands ovales). Les routeurs de bordure, ou passerelles, sont représentés avec des ronds grisés. À droite du schéma, un AS est agrandi et sa structure interne, composée de routeurs (les ronds blancs), est révélée.

On peut donc déjà affiner la vision de l'utilisateur envoyant des données vers une adresse quelconque : ces données vont transiter d'AS en AS jusqu'à atteindre la destination. Chaque AS, en recevant les paquets, se chargera de choisir les AS à qui il pourra les transmettre en fonction de l'adresse destination indiquée dans les paquets. À l'intérieur de l'AS, les paquets transiteront par un certain nombre de routeurs, eux-mêmes choisissant le prochain routeur à qui transmettre les paquets, toujours en fonction de l'adresse destination.

À chacune de ces étapes, l'AS se chargeant du transit du paquet n'a pas besoin de connaître la route complète suivie par ce dernier : il ne s'occupe que du transfert vers le prochain AS. La transmission se fait donc à un niveau *local* tout le long du chemin.

Le processus permettant aux AS de s'organiser entre eux de manière globale pour que tous les paquets `IP` sur l'Internet soient correctement acheminés vers leur destination est le protocole BGP (*Border Gateway Protocol*) [93]. Il permet à chaque AS de déterminer à quel AS (parmi ses voisins directs) il faut transmettre un paquet en fonction de son adresse `IP` destination. Nous ne détaillerons pas son fonctionnement, qui est complexe [94], mais l'idée générale est que les AS partagent leurs informations pour obtenir une vision globale – mais très partielle – du réseau, qui leur permet de s'organiser.

L'Internet est donc constitué de parties autonomes, liées entre elles localement, mais rassemblées autour du respect de certaines règles ou protocoles. Ces derniers sont matérialisés dans certaines RFC (*Request for Comments* [91]), consensus entre parties indépendantes qui sont devenues les standards de l'Internet. Ces conventions permettent à l'ensemble de se comporter de manière apparemment unifiée, mais la nature profonde de l'Internet reste décentralisée.

1.1.3 Un réseau hétéroclite

La gestion décentralisée de l'Internet ainsi que son évolution historique ont provoqué une grande disparité, qui se ressent à tous les niveaux.

- La densité du réseau a tendance à suivre la répartition géographique de la population.
- De nombreux facteurs nationaux influencent la structure de l'Internet : la richesse, la politique en matière de réseaux (selon qu'il finance, surveille, administre, ou n'intervient pas), le terrain (plaine, montagne), ou encore la présence de pays voisins et leurs besoins en termes de connectivité.
- Les AS ont différentes fonctions : fournisseurs d'accès au grand public, AS 'centraux' constituant le cœur de l'Internet, AS réduits physiquement à une salle de routeurs dans laquelle d'autres AS s'interconnectent... Les tailles varient énormément, de même que la structure : certains seront *par se* quasi arborescents alors que d'autres seront très denses.
- Les installations sont hétérogènes : des câbles transocéaniques aux fils téléphoniques branchés sur les modems ADSL, les paramètres comme le débit ou la longueur maximale changent. La *nature* des liaisons n'est elle-même pas limitée aux câbles reliant *deux* points : citons les câbles coaxiaux, les boucles de fibres optiques, les liaisons WiFi ou WiMax, par satellite...
- Les constructeurs de matériel réseau ont leurs différences, et l'évolution technologique est assez rapide pour créer des disparités entre un AS qui vient de se rééquiper et un AS qui le fera l'année suivante.

L'Internet est donc profondément hétérogène, et cet aspect ne peut être ignoré tant la diversité de l'Internet apparaît à toutes les échelles. Dès lors, il semble vain d'imaginer pouvoir décrire le réseau sans perdre une partie de l'information. Nous montrons pour exemple dans la Figure 1.3 un aperçu d'une très petite partie de la topologie `IP`, coloriée par AS (des sommets d'une même couleur représentent des routeurs appartenant au même AS), qui, malgré le fait qu'elle soit obtenue par sondage *uniforme*, fait apparaître le caractère hétéroclite de l'Internet. Nous discuterons plus en détail ce type de mesures dans la suite.

1.1.4 Un réseau mal connu

Si l'hétérogénéité du réseau rend sa description ardue, sa structure décentralisée ne rend pas les choses plus simples. La concurrence et les risques d'attaques incitent en effet de nombreux AS à ne pas communiquer leur topologie de réseau. L'Internet est donc mal connu, car on ne peut se contenter des informations partielles données par les AS de bonne volonté (voir par exemple [129, 131, 128, 130]) pour se faire une image précise du réseau dans sa *globalité*.

Des bases de données globales existent cependant : citons notamment les bases WHOIS des Registres Internet Régionaux [22, 23, 24, 26, 27], les données de géolocalisation possédées par des sites commerciaux [117, 118, 116], les bases de données d'entrées BGP [124] ou encore les systèmes DNS[1]. Ces informations sont cependant souvent peu fiables (selon leur

[1]DNS signifie *Domain Name System* et permet de convertir un nom de domaine de type upmc.fr en

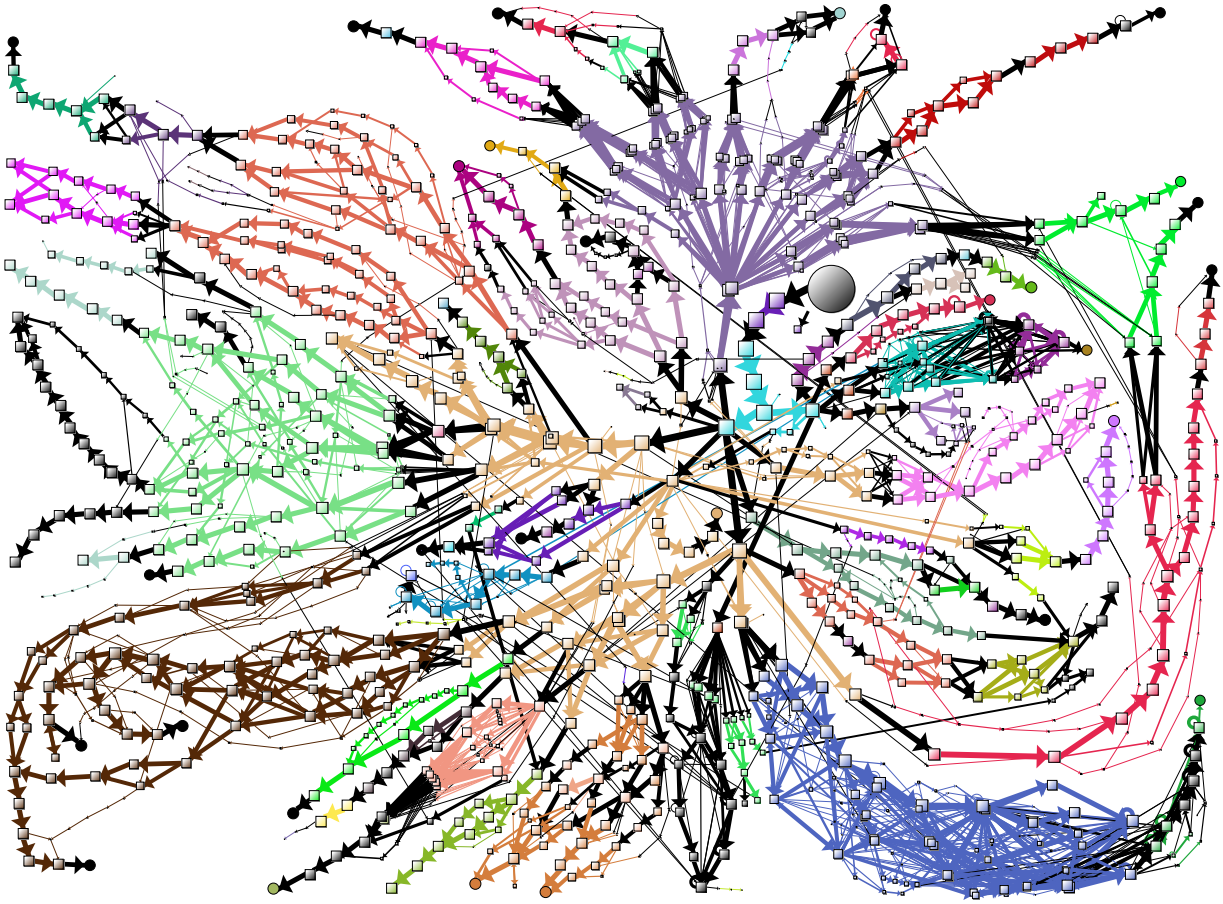

FIG. 1.3 – Aperçu d'une partie de la topologie IP de l'Internet, obtenue en sondant périodiquement les chemins parcourus depuis une source (le gros sommet rond) vers 50 destinations (les autres sommets ronds) prises au hasard dans l'espace d'adressage IP. Les nœuds représentent des routeurs, coloriés selon l'AS auquel ils appartiennent. L'épaisseur des liens illustre la fréquence avec laquelle les sondes y sont passées.

fournisseur) car incomplètes et mal mises à jour. Quant aux bases de données recueillant la topologie IP de l'Internet, elles sont rares et sont soit d'accès restreint [126, 122], soit approximatives et/ou trop partielles [115, 84]. Quant aux outils publics [46, 100] conçus pour l'interrogation en direct du réseau pour obtenir ce type d'information de manière locale, ils ne permettent pas d'effectuer une mesure conséquente de l'Internet en des délais raisonnables.

Citons également des bases de données publiquement disponibles permettant de déterminer (là encore, avec des erreurs possibles) l'AS auquel appartient une adresse IP [63, 70, 69, 73]. Pour résumer les informations disponibles :

adresse IP, et réciproquement. Ce système est en réalité maintenu et géré de manière distribuée, mais fonctionne en première approche comme une seule base de donnée cohérente et globale.

- La topologie IP complète est loin d'être connue [114], et on ne sait pas encore bien évaluer le degré de partialité des données existantes.
- Environ 83% des adresses IP attribuées ont une entrée DNS permettant de les traduire en nom de domaine[2].
- La géolocalisation des adresses IP, en plein développement, reste imparfaite et incomplète, et surtout très inégale selon les régions [45].
- L'association des adresses IP aux AS est, elle, relativement bien connue puisque les bases de données disponibles publiquement citées plus haut permettent d'associer au moins 3/4 des adresses IP attribuées, avec un taux d'erreur *a priori* bas (voir la Figure 1.3 : les nœuds dont l'AS n'a pas pu être obtenu sont en noir, et remarquer que les colorations paraissent cohérentes avec la connectivité).

1.1.5 L'Internet comme un système complexe

Aspects multi échelles

Malgré la connaissance imparfaite de l'Internet et les difficultés citées ci-dessus, les AS parviennent à administrer leur réseau convenablement. La clé est dans une approche *multi échelles*. Pour prendre un exemple, considérons le réseau de *free* montré en Figure 1.4. Ce réseau est dimensionné et géré au moins sur trois échelles : l'échelle nationale, qui prend en compte les liens entre grandes villes françaises et étrangères, le niveau francilien qui se charge du réseau en Ile-de-France, et enfin le réseau final de distribution aux particuliers, autour des NRA (nœuds de raccordement des abonnés) qui constituent le lien entre le réseau de free et les passerelles résidentielles, elles-mêmes reliées directement aux abonnés. Cette gestion peut se faire de manière presque totalement indépendante : une équipe pourrait se charger exclusivement du réseau national pendant qu'une autre s'occuperait du réseau francilien.

Cet exemple est pris à l'échelle d'un seul AS, mais l'Internet s'organise de manière comparable, chaque AS étant géré indépendamment. Que ce soit de la surveillance, une mise en place, ou une intervention, l'administration de l'Internet se fait fondamentalement à plusieurs échelles, traitées de manières quasi indépendantes. Si une telle vision donne des résultats satisfaisants, elle n'en reste pas moins imprécise : le fait de découper le réseau en couches successives et de les analyser séparément ne reflète pas toutes les interactions qui existent entre elles.

Un objet "vivant"

La vision de l'Internet comme un système multi échelles, dont les sous-unités fonctionnent indépendamment mais dont le tout forme un objet cohérent assimilable à une seule entité, fait penser à un organisme vivant composé d'organes, eux-mêmes composés

[2]Source : expérience faite en résolvant 7828 adresses IP aléatoires répondant au 'ping', en février 2007, à l'aide de la commande `host` de BIND [25], disponible sur la majorité des systèmes Unix/Linux.

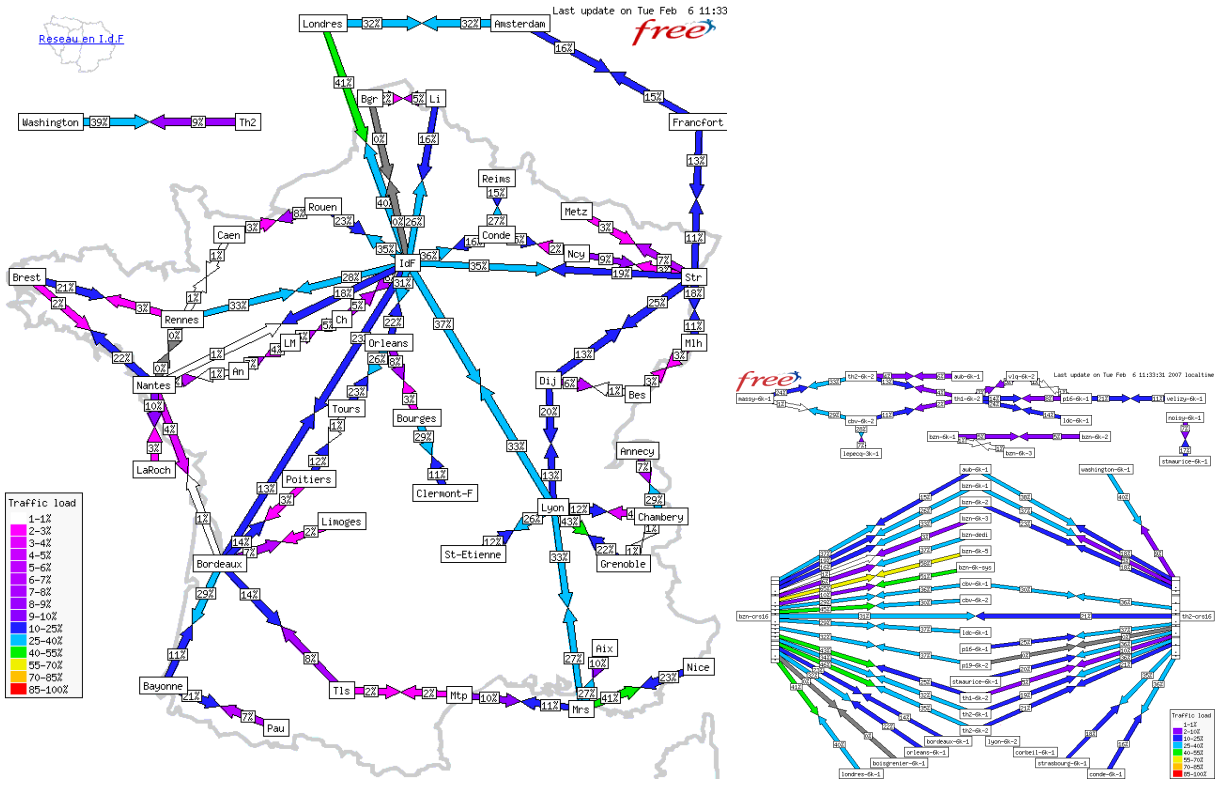

FIG. 1.4 – Cartes de surveillance du réseau de *free*. À gauche, le réseau national, à droite, un zoom sur le réseau de l'Ile-de-France (représenté par le rectangle *IdF* sur la carte nationale).

de cellules, elles-mêmes composées de protéines et autres composants élémentaires et ainsi de suite.

La métaphore s'applique tout particulièrement au comportement de l'Internet, qui se développe selon les volontés et usages combinés des individus, et reste par là même imprévisible. Voir l'Internet comme un objet vivant, c'est reconnaître tout sa complexité, au delà de sa fonction de boîte de connexion universelle.

Réalisant cela, les physiciens se sont penchés sur de nombreuses problématiques liées à l'Internet en considérant ce dernier comme un objet *inconnu* qu'il faut *sonder*. Contrairement à l'approche classique des informaticiens qui étudient en détail les protocoles, les spécifications et les matériels, les physiciens tentent de déterminer les lois régissant le comportement de l'Internet par l'*observation*. Cette approche est relativement nouvelle et s'est répandue hors du domaine de la physique. Voir [85] pour une synthèse. Dans cette thèse, nous abordons l'Internet de cette manière.

1.2 Enjeux : pourquoi connaître l'Internet ?

Nous avons insisté, dans la section précédente, sur la complexité de l'Internet et les difficultés posées par son étude. Cependant, étant donnée la nature auto-organisée de ce dernier et son bon fonctionnement, on peut se demander si une meilleure connaissance de l'Internet apporterait réellement quelque chose. Nous allons, dans cette section, répondre à cette question en montrant un aperçu des bénéfices que l'on pourrait en tirer, et des possibilités qui s'ouvriraient à nous.

1.2.1 Développement logiciel

`IP` s'occupe de l'acheminement d'un paquet de données d'un point du réseau à l'autre et offre un certificat permettant de savoir si les données de contrôle `IP` (contenue dans l'en-tête `IP`) ont été corrompues grâce au *Checksum* ou Somme de Contrôle [88]. Cette fonctionnalité est limitée car en pratique les transferts ne se limitent que très rarement à un seul paquet envoyé unidirectionnellement. Pour organiser des *flux* de données, on fait donc appel à des protocoles de niveau supérieur. Le plus connu, et le plus utilisé aujourd'hui, est le protocole TCP (pour *Transfer Control Protocol*) [87], qui offre :

- des accusés de réception permettant de s'assurer du bon transfert des données,
- un contrôle du débit en fonction de la congestion de la liaison,
- une procédure de mise en ordre des paquets pour la reconstitution de la donnée complète,
- un système de *ports* permettant à un même utilisateur de faire tourner en parallèle plusieurs applications utilisant le réseau, sans que celles-ci ne rentrent en conflit et ne mélangent leurs données.

La mise au point et l'optimisation de tels protocoles nécessite une bonne connaissance du fonctionnement *interne* de l'Internet, au delà du simple paradigme de boîte noire acheminant les paquets vers leur destination. Pour exemple, le mécanisme de contrôle de débit utilisé par le protocole TCP dépend directement de la congestion du réseau traversé, et son utilisation massive sur le réseau pose alors une problématique non triviale de cohésion du système. En effet, des dynamiques pathologiques peuvent se mettre en place [107, 57, 72], les effets pervers d'un algorithme étant difficiles à prévoir. On peut par exemple imaginer une situation où N flux se partageant un même lien surchargé vont, par un effet de synchronisation malheureuse du protocole, diminuer leur débit en même temps et ainsi sous-utiliser la bande passante disponible. La connaissance des situations à problèmes et de leur fréquence est donc un facteur important dans la conception d'un protocole de transmission, quel que soit son niveau. Ne pouvant être sûr d'avoir énuméré toutes les situations à problèmes de manière exhaustive à cause de la nature complexe de l'Internet, tout nouveau protocole doit faire ses preuves à l'essai : sinon en pratique, du moins sur une simulation réaliste du réseau.

C'est d'autant plus vrai pour les protocoles de plus haut niveau tel HTTP [92], qui utilisent un protocole sous-jacent comme TCP, mais en ajoutant des degrés de complexité, concrétisé par l'ajout d'informations de contrôle dans les données. La simulation est un

atout précieux, et de nombreux travaux scientifiques y ont recours. Elle reste l'intermédiaire logique entre la conception et la mise en pratique. Elle permet notamment, lors de la conception d'une nouvelle technologie :

- de vérifier le bien-fondé des prévisions théoriques concernant son comportement
- de détecter les situations spéciales provoquant son échec et dévoilant ses faiblesses, et dans le cas contraire de se convaincre qu'elle n'en a pas,
- d'observer son comportement sur des réseaux de différentes échelles, et de déterminer ainsi si la mise en pratique à l'échelle de l'Internet est viable,
- de simuler des conditions extrêmes, rares voire quasi impossibles sur le réseau réel, et de s'assurer ainsi une marge de sécurité pour la mise en pratique.

Une bonne connaissance du réseau, importante lors de l'ébauche de nouveaux protocoles réseaux, est bien évidemment cruciale au moment de la simulation, qui se doit d'être réaliste.

1.2.2 Développement matériel

Après avoir vu les enjeux liés à l'innovation des techniques *logicielles* liées à l'Internet, et concernant l'ensemble des utilisateurs, nous nous intéressons aux enjeux concrets liés au réseau *physique*.

Dimensionnement

La croissance de l'Internet, qui vient de dépasser en 2007 le milliard d'utilisateurs dans le monde [120], est telle que de nombreux opérateurs se lancent encore aujourd'hui dans la construction de nouveaux AS, pendant que les opérateurs existants étendent les leurs. La problématique du dimensionnement de tels réseaux est alors cruciale au vu des lourds investissements nécessaires. Avant de créer un réseau, ou de moderniser un réseau existant, il est donc important de bien connaître la demande et son évolution afin de construire un réseau assurant une bonne qualité de service. Il faut donc anticiper et localiser les principaux flux de données qui auront lieu entre les futurs équipements et l'extérieur.

Il est important également de posséder une bonne expertise en termes de topologie. Quelle topologie reviendra la moins chère tout en assurant une qualité de service et une couverture suffisante ? Si des modèles de conception simples existent et sont utilisés, notamment par l'ARCEP [3], le problème reste difficile. Même avec une vision très simpliste, la solution optimale ne peut se calculer sur ordinateur au delà de quelques milliers de nœuds [66]. L'expertise humaine (et ses erreurs) résout alors le problème tant bien que mal, et la connaissance globale de l'Internet apporte une plus-value importante.

[3] l'Autorité de Régulation des Communication Électroniques et des Postes s'occupe (entre autres) de réguler le marché français de l'accès à Internet.

Robustesse

Une importante caractéristique de l'Internet est sa *robustesse*. En pratique, par exemple, la plupart des AS conçoivent la topologie de leur backbone de manière à obtenir au moins une 2-connexité sur leurs principaux centres, ce qui équivaut à dire que toute coupure accidentelle d'une unique liaison n'affectera pas la connexité des centres critiques. Les protocoles de routage permettent en effet un réacheminement relativement rapide, souvent de l'ordre de la minute [52]. Néanmoins, la charge additionnelle supportée par les équipements faisant office de pont provisoire peut s'avérer trop forte et dégrader l'ensemble du réseau.

L'Internet est robuste, mais est-il fiable pour autant ? En pratique, les pannes isolées et incidents divers (dont maintenance des équipements) occasionnant des coupures locales de réseau ne sont pas rares [129]. Un AS doit donc évaluer la probabilité de panne ou d'incident sur son réseau et être en mesure de répondre à n'importe quelle de ces pannes, et devra donc concevoir dès le début un réseau capable de résister à ces incidents, évaluer le manque à gagner provoqué par les pannes potentielles et se préparer en fonction.

D'autres scénarios sont envisageables, notamment l'attaque. Si les attaques directes sur les équipements physiques ne font pas légion, les attaques de *déni de service* (DoS), qui visent à paralyser un ou plusieurs routeurs ou serveurs en l'inondant sous un trafic illégitime, peuvent réellement nuire à la connectivité du réseau [30, 21]. Plus naturels, de nombreux phénomènes légitimes peuvent également s'apparenter à une attaque : par exemple, la diffusion soudaine d'un contenu de type 'scoop' amenant des millions d'internautes à échanger un fichier dans les minutes qui suivent son apparition s'apparentera à une attaque sur la source du fichier.

Dans tous les cas, un AS doit être conscient de ce type de risques pour bien gérer son réseau.

Diagnostic

La surveillance du réseau permet d'anticiper un certain nombre de phénomènes pathologiques, voire de réagir à temps. Elle permet également à un AS de mieux comprendre l'utilisation de son réseau par ses clients, et ainsi favorise – comme discuté plus haut – le dimensionnement futur du réseau et l'anticipation des problèmes potentiels. Nous montrons en Figure 1.5 un exemple de telle surveillance, disponible en ligne [128], pour un réseau très petit par son nombre de nœuds (12). Sur des réseaux plus grands, la surveillance, même si elle permet de détecter voire de diagnostiquer les gros incidents, reste difficile à gérer de par la masse d'information décrivant le réseau. Il est facile de se retrouver perdu devant l'immense liste des statistiques possibles que l'on peut effectuer, sans savoir laquelle aura une réelle signification.

1.3 Modélisation

Nous avons montré dans les sections précédentes le besoin d'outils et de concepts permettant de mieux comprendre l'Internet, et rappelé les difficultés posées par le traitement

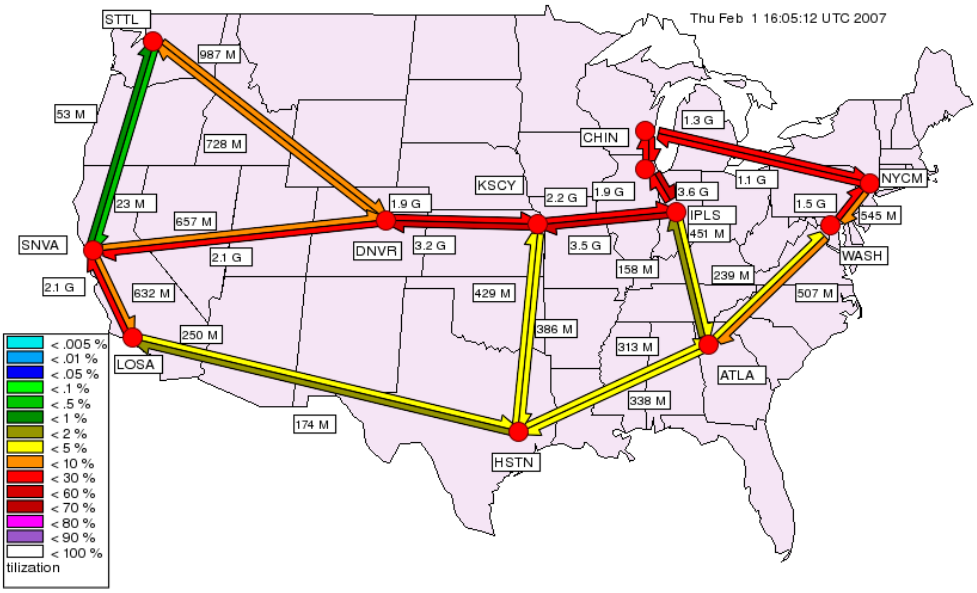

FIG. 1.5 – Page de diagnostic (en direct) du réseau académique américain “Internet 2”

d'un objet aussi complexe. La modélisation, comme nous allons le voir dans cette section, permet de sélectionner dans la masse d'informations qui décrit l'Internet les données les plus pertinentes pour l'approche de telle ou telle problématique. Elle permet notamment à un point de vue humain de cerner des aspects de l'Internet en s'affranchissant de la complexité liée aux aspects ignorés. L'abstraction de l'Internet se fait sur trois niveaux : topologie, routage et trafic.

1.3.1 Topologie

L'Internet physique peut se résumer à une collection de routeurs et de câbles, ce qui correspond à une *interconnexion* entre machines. Ce point de vue introduit la vision de l'Internet comme un **graphe** plutôt qu'une simple boîte noire connectant les utilisateurs. Un graphe se définit mathématiquement par un couple d'ensemble (V, E), où V est l'ensemble des *sommets* du graphe (aussi appelés nœuds), et E est l'ensemble des *arêtes* (aussi appelées liens). Les arêtes sont des couples (ou paires) de sommets : $E \subseteq V \times V$. Une telle vision de l'Internet est dite *topologique* car elle se focalise sur le graphe lui-même, qu'on désigne aussi par la **topologie** du réseau. Nous nous focalisons dans cette thèse sur la topologie `IP` qui correspond – idéalement – à la modélisation des routeurs par des sommets, et des liens physiques par des arêtes. En réalité, le terme “topologie `IP`” est plus précis et ne correspond pas exactement à cette vision simplifiée, notamment en raison de la diversité des routeurs et liens existants qui empêche une classification aussi catégorique. En effet, la vision de topologie `IP` considère comme un routeur unique certains regroupements de routeurs, et ne considère que des liens uniques entre routeurs alors que ceux-ci peuvent être

multiples. Pour le moment, cette vision intuitive mais inexacte convient à notre propos, et nous reviendrons sur ce point en temps utile. Notons que d'autres abstractions topologiques sont envisageables : par exemple, la vision correspondant à la réalité physique, où chaque point de transit (en incluant les switchs, hubs, points d'amplification optique, etc.) correspond à un sommet et chaque lien physique correspond à une arête. Une autre vision beaucoup utilisée se situe au niveau des AS : les AS sont les sommets, et sont reliés selon les liaisons inter-AS existantes comme décrit en Section 1.1.2. Voir par exemple [70].

Il existe de nombreuses variantes de la notion de graphe, permettant entre autres les arêtes multiples, introduisant des arêtes *dirigées*, ou donnant des conditions plus restrictives. Dans le cas d'une vision de l'Internet en tant que graphe, il suffit de voir que les nœuds du graphe en question représentent les routeurs, et les liens représentent les câbles ou éventuellement les autres supports de communication.

D'autres variantes proposent un enrichissement intéressant de la notion de graphe dans notre contexte : on peut par exemple ajouter des caractéristiques aux nœuds et/ou aux liens, comme une couleur, un poids, ou toute autre grandeur. Ce type de caractéristique supplémentaire permet d'utiliser des graphes plus riches, et donc plus proches de la réalité. En pratique, pour modéliser l'Internet, on pourra utiliser un graphe dont les liens seront munis d'un *poids* correspondant au débit et où les nœuds auront une *taille* correspondant au nombre maximum de paquets pouvant être traités par seconde et une couleur correspondant à l'AS dont ils font partie. De nombreuses autres caractéristiques du réseau physique peuvent être prises en compte dans ce type de modélisation, qui a l'avantage de pouvoir bénéficier du corpus de théorèmes et de techniques obtenus par la théorie et l'algorithmique des graphes.

Pour finir, précisons que la topologie a prouvé son importance dans le fonctionnement de l'Internet. Par exemple, des travaux ont montré que l'efficacité de la diffusion multicast s'avère dépendante de la géométrie de l'Internet [2, 3], ce qui justifie déjà l'intérêt qu'on peut porter à la connaissance et à la modélisation de la topologie.

1.3.2 Routage

À la vision de l'Internet en tant qu'interconnexion de machines, on peut substituer la vision fonctionnelle d'une entité d'*acheminement* de données. Le but de l'Internet est de convoyer des paquets de données, et le **routage** en est le mode de fonctionnement : quand un routeur reçoit un paquet destiné à une certaine adresse X, il saura – à l'aide d'une table de routage – par lequel de ses liens sortants il faudra le renvoyer. De proche en proche, le paquet arrivera à destination : la *route* qu'il suit est alors matérialisée par la suite de routeurs qu'il traverse. On parlera donc d'une route *depuis* une certaine adresse IP *vers* une autre adresse IP. Typiquement, les routes sont relativement courtes : les paquets traversent très rarement plus d'une quarantaine de routeurs. Notons qu'entre une source et une destination donnée, la route suivie n'est pas forcément unique, et peut varier au fil du temps.

La seule vision de graphe ne permet pas de prendre ce fonctionnement en compte, aussi faut-il ajouter au modèle topologique un modèle de routage lorsqu'on veut simuler

l'Internet. Le rôle d'un modèle de routage est de définir la manière dont les paquets sont transmis à travers le réseau. En pratique, le routage est effectué par les routeurs, dont le fonctionnement global doit obéir à un certain nombre de règles fixées (le plus souvent par consensus, comme les RFC). Mais les routeurs présentent de nombreuses spécificités selon leur marque et selon l'AS qui les utilise. Ce dernier définit la politique de routage comme il l'entend, et le grand nombre d'AS, ainsi que leur indépendance, rend la modélisation du routage sur l'Internet une tâche difficile.

Même en imaginant un Internet composé de routeurs homogènes se comportant de la même façon, le routage peut rester réfractaire à toute analyse globale car il est effectué par des *millions* de machines fonctionnant en parallèle, chacune dotée d'une puissance de calcul non négligeable. Autant en termes d'analyse théorique que de simulation informatique, une politique de routage, simple quand elle est considérée au niveau d'un unique routeur, peut, utilisée à grande échelle, se révéler inextricable.

Pour simuler le routage, deux approches principales existent :

- L'approche **distribuée**, que nous n'étudierons pas ici, se focalise sur le comportement d'un routeur, et simule ce comportement – de manière simplifiée – pour chaque nœud de son réseau virtuel.
- L'approche **globale** fait le choix délibéré du non réalisme (puisque l'Internet n'est pas géré de manière centralisée) pour obtenir une définition plus simple mathématiquement parlant, et donc plus apte à l'analyse. Elle consiste essentiellement à décrire la politique de routage en se basant sur la connaissance globale du réseau. On pourra par exemple faire emprunter aux paquets le chemin ayant la plus grande bande passante parmi l'ensemble des chemins envisageables.

Ces deux visions sont assez différentes dans leurs usages, même si elles peuvent donner les mêmes résultats. En termes de simulation, l'approche distribuée souffre souvent d'une grande difficulté à être mise en pratique, car simuler le comportement de millions de routeurs en parallèle peut demander d'énormes ressources de calcul. C'est en partie ce qui nous a poussé à nous tourner plus particulièrement vers l'approche globale. Un des modèles les plus naturels, que beaucoup de travaux privilégient dans le cadre d'études qualitatives de la dynamique de l'Internet, en fait partie : il s'agit du modèle de routage par **plus courts chemins**[4]. Ce modèle suppose que tout paquet envoyé par A vers B empruntera un plus court chemin dans le graphe représentant la topologie du réseau, ce qui revient à dire que le paquet traversera le moins de routeurs possible (voir les Sections 1.4.4 et 3.1.3 pour plus d'information sur la modélisation du routage par les plus courts chemins).

1.3.3 Trafic

Une fois que la topologie et le routage de l'Internet ont été modélisés, il manque encore un élément pour pouvoir mener des simulations : le trafic. L'Internet ne fait que relayer les flux de trafic existants entre l'ensemble des hôtes. La topologie et le routage décrivent la

[4] On considère ici des graphes non valués : les plus courts chemins sont ceux qui traversent le moins de noeuds.

manière dont ces flux sont transportés dans le réseau, mais ne dit rien sur leur composition qui doit donc elle aussi être modélisée.

L'élément central, dans la modélisation des flux de données, est la **matrice de trafic**. Si on numérote les N hôtes connectés à l'Internet par $1, 2, \cdots, N$ la matrice de trafic T sera un tableau $N \times N$ ou l'élément $T_{i,j}$ de la i-ième ligne et de la j-ième colonne représente la quantité (ou le débit) de données que l'hôte i transmet à l'hôte j. Cette matrice peut être considérée à un instant t, ou sur une plus longue période de temps. Les modèles de trafic existants sont nombreux et de natures variées [7, 16, 34, 97].

L'étude de la mesure et de la modélisation du trafic sur l'Internet constituent un domaine à part entière. Cependant, nos travaux se focalisant essentiellement sur la topologie et la mesure de celle-ci (qui implique aussi l'étude du routage), nous ne développerons pas davantage ce sujet.

1.3.4 Approches mixtes

Dans l'Internet, les trois niveaux d'abstraction que sont la topologie, le routage et le trafic sont intrinsèquement liés : les politiques de routage sont définies en prenant en compte la topologie, le trafic à son tour dépend de la qualité des liaisons hôte à hôte, qui dépendent elles-mêmes de la topologie et du routage, mais la topologie et le routage évoluent également, au fur et à mesure que les AS développent et étendent leur réseau, en fonction des besoins, c'est-à-dire du trafic déjà existant ...

Il peut paraître dès lors artificiel de séparer ces trois entités. Cette séparation est pourtant logique dans le contexte de la recherche, puisqu'elle permet de traiter des objets bien cernés, qui sont de natures profondément différentes. De plus, elle est compatible avec les outils de mesure de l'Internet existants : certains mesurent la topologie, d'autres observent le routage, et d'autres le trafic. De plus, même séparée des autres, chacune de ces visions de l'Internet est extrêmement complexe, et reste mal connue.

Il existe cependant des approches mixtes, des modèles liant topologie et routage [37], ou routage et trafic, ou d'autres combinaisons. Par exemple, les modèles s'intéressant à l'évolution de la bande passante utilisée dans un lien donné [133] sont impliqués dans les trois niveaux d'abstraction et ne peuvent se réduire à l'une des trois catégories.

Nous nous focalisons dans cette thèse sur la topologie. De plus, nous ne considérons que la topologie *primitive* : nœuds et liens simples, sans poids, couleur, etc. Nous utilisons plusieurs modèles topologiques au long de cette thèse, mais n'utilisons qu'à un seul endroit un modèle de routage, dont les caractéristiques ne sont pas déterminantes dans nos résultats, comme nous le verrons.

1.4 L'Internet comme un graphe

Les modèles de topologies sont souvent des versions simplifiées du réseau observé. Il est dès lors difficile d'estimer la qualité d'un graphe en tant que modèle d'un réseau, car on

ne sait pas évaluer la *ressemblance* de deux graphes de manière simple. En pratique, on compare donc certaines propriétés du graphe modèle avec celles du graphe modélisé, et la ressemblance – c'est-à-dire la qualité du modèle – est évaluée en fonction. De nombreux modèles ont même pris cette approche à la lettre et ont été directement conçus dans le but d'imiter un certain ensemble de propriétés du graphe originel.

Dans cette section, nous allons passer en revue les principales propriétés de graphes, introduire les notations qui y sont liées, et exposer les principales méthodes basées sur ces propriétés permettant de comparer deux graphes. Nous donnerons, pour chaque propriété, la manière dont elle se comporte généralement pour les grands réseaux rencontrés en pratique. Ceci constitue un préliminaire à la description des modèles eux-mêmes, qui aura lieu dans la section suivante. Pour toute la suite de cette thèse, il est également important d'être familier avec le vocabulaire et les concepts introduits ici.

1.4.1 Définitions et notations

Rappelons qu'un graphe G est un couple (V, E), où V est un ensemble fini de sommets et E un ensemble d'arêtes : $E \subseteq V \times V$. On dit que le graphe est *non orienté* quand $(a, b) \in E \;\Rightarrow\; (b, a) \in E$. De plus, il est dit *simple* si $\forall (a, b) \in E,\ a \neq b$ [5]. Quand un graphe est non orienté, on désigne indifféremment les arêtes (a, b) et (b, a) par $(a - b)$. S'il est orienté, on désignera (a, b) par l'*arc* $a \to b$. Dans cette thèse, les graphes seront non orientés, sauf indication contraire ou utilisation explicite du terme *arc* ou de liens *fléchés*. Si l'arête $(a - b)$ existe, on dit également que a et b sont *voisins*. Le *voisinage* $\mathrm{Vois}(v)$ d'un sommet v est l'ensemble de ses voisins.

Le nombre de sommets du graphe est noté $N = |V|$, son nombre d'arêtes est noté M. Comme E est l'ensemble des *couples* de sommets liés, on obtient la relation $M = |E|/2$ pour tout graphe non orienté. De plus, comme le graphe est simple, le nombre d'arêtes M ne peut excéder le nombre de paires d'éléments distincts de V, et donc $M \leq N \cdot (N-1)/2$. Nous utiliserons également le terme de *taille* d'un graphe en parlant du nombre de sommets N.

On peut réaliser l'opération intuitive de prendre un "morceau" d'un graphe à l'aide du concept de *sous-graphe*. Un sous-graphe de (V, E) est un graphe (V', E') tel que $V' \subset V$ et $E' \subset (E \cap (V' \times V'))$.

1.4.2 Degrés et densité

Le *degré* $deg(v)$ d'un sommet v est le nombre de ses voisins, c'est-à-dire le nombre de sommets partageant une arête avec lui : $deg(v) = |\mathrm{Vois}(v)|$. Le degré *moyen* z d'un graphe est la moyenne de cette quantité sur l'ensemble de ses sommets. Cette propriété est l'une des plus fondamentales car elle agit sur de nombreuses autres caractéristiques du réseau, comme nous verrons par la suite. Le degré moyen s'obtient en fonction du nombre de

[5] Cette contrainte interdit les boucles $(a - a)$. Notons que les arêtes multiples (ou multi-arêtes) ont déjà été interdites implicitement quand E a été défini en tant qu'*ensemble*.

sommets N et d'arêtes M par $z = \frac{2M}{N}$. Une autre quantité liée aux degrés est la densité θ du graphe, qui chiffre la proportion $\frac{|E|}{|V \times V|}$ d'arêtes existantes parmi les arêtes possibles. Elle est liée de manière immédiate au degré moyen par $\theta = \frac{z}{N-1} = \frac{2M}{N \cdot (N-1)}$, et vérifie $\theta \in [0, 1]$.

Le degré moyen donne une information importante sur le graphe, mais reste moins riche que la donnée complète des degrés de tous les sommets $deg(v_1), \cdots, deg(v_N)$, que l'on désignera par *séquence* de degrés. Cette dernière est cependant souvent difficile à obtenir sur des mesures incomplètes de réseaux, et on pourra lui substituer la loi de *distribution* des degrés qui donne la répartition statistique (éventuellement approchée) des degrés parmi les sommets du graphe.

Pour résumer, on utilise principalement ces trois propriétés globales liées au degré :

- Le degré moyen donne une information concise, qui est du coup extrêmement significative (relativement à la quantité d'information fournie). On peut lui substituer la densité dans certains cas.
- La *séquence* des degrés d'un graphe est la donnée complète des degrés de chaque sommet.
- La loi de distribution des degrés (ou distribution de degrés) d'un graphe donne un *aperçu statistique* de la séquence de degrés attendue.

Le degré moyen et la distribution de degrés ont l'avantage d'être mesurables *localement* : on pourra les évaluer de manière approchée sur un **sous-graphe** (à condition que celui-ci constitue un échantillon statistiquement représentatif). Ce sont donc des propriétés *intensives*, qui changent peu si on considère une vision partielle, non biaisée, du graphe[6].

Sur les réseaux rencontrés en pratique, le degré moyen est souvent petit par rapport au nombre de sommets. Ces réseaux sont donc *peu denses*. Une autre propriété intéressante de beaucoup de réseaux réels est le caractère fortement **hétérogène** de leur distribution de degrés. Alors que les sommets de très faible degré (typiquement, entre 0 et le degré moyen z) constituent l'immense majorité, un nombre significatif de sommets atteignent des degrés très élevés. En pratique, on observe pour de nombreux réseaux une forte ressemblance entre la distribution de leurs degrés et une loi de puissance : la probabilité d'avoir un degré k est inversement proportionnelle à une puissance réelle de k, ce qui peut s'écrire $P_k \sim k^{-\alpha}$. Le réel α est appelé l'exposant de la loi de puissance, et il détermine la forme de la distribution. Des exemples de telles distributions mesurées sur des réseaux réels sont montrés en Figure 1.6 et leur forme de loi de puissance est mise en évidence par l'adoption de l'échelle logarithmique.

Pour comparer deux graphes en se basant sur leurs degrés, on pourra donc, en plus du degré moyen, comparer les exposants des lois de puissance associées à leurs distributions de degrés (si c'est pertinent). Notons qu'en pratique, les exposants mesurés se situent le plus souvent entre 2 et 3 [80].

Il est également important de préciser que, contrairement aux lois de puissances "propres" pour lesquelles la moyenne est fixée automatiquement par l'exposant de la distribu-

[6] Il est généralement admis [80], comme c'est le cas pour de nombreux modèles, que les degrés d'un sous-graphe de taille N/k sont diminués en moyenne d'un facteur d'échelle logarithmique en $O(\log k)$, ce qui reste faible par rapport à la division par k des quantités *extensives* telles que N ou M.

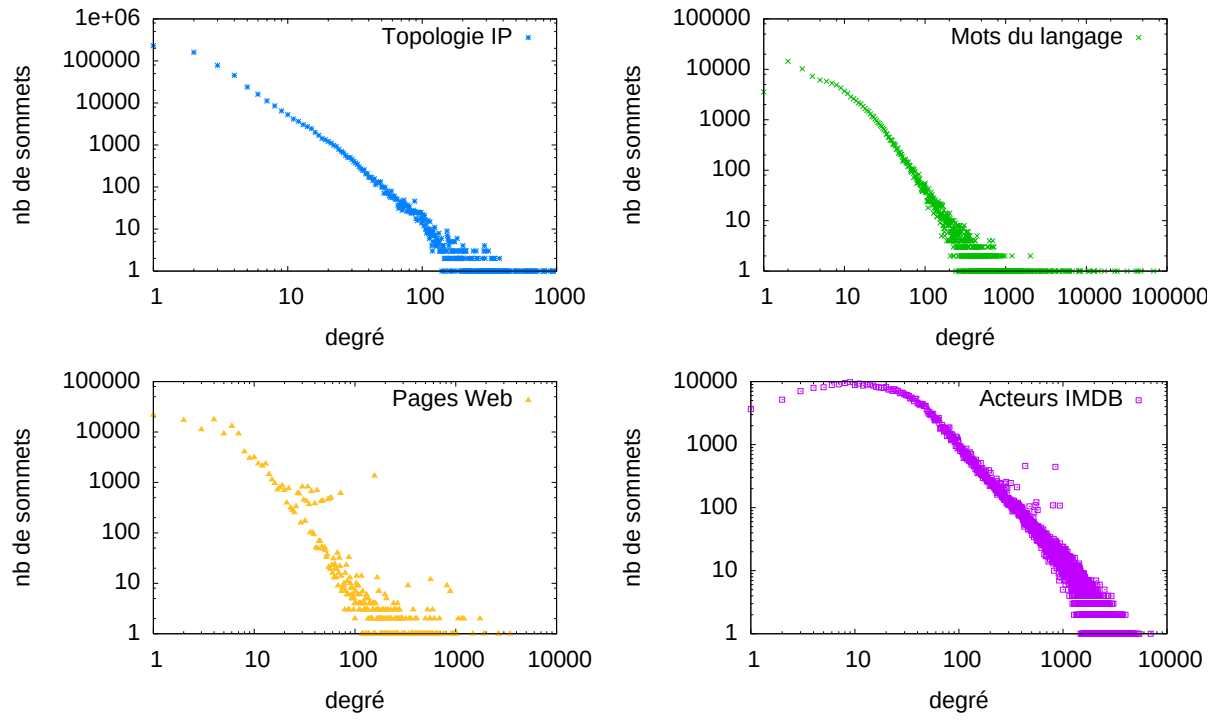

FIG. 1.6 – Distribution des degrés dans quelques grands réseaux. De haut en bas et de gauche à droite : une mesure de la topologie IP de l'Internet (projet Skitter), le réseau des co-occurrences de mots dans les définitions du dictionnaire (deux mots sont voisins quand ils apparaissent dans une même définition), une partie du graphe du Web (deux pages sont voisines quand l'une pointe vers l'autre), et le réseau des acteurs de cinéma donné par IMDB [119] (deux acteurs sont voisins s'ils ont joué ensemble dans un film).

tion[7], les lois rencontrées en pratique s'assimilent plus à des lois de puissance déviées, ayant un comportement asymptotique (vers les hauts degrés) respectant bien une loi de puissance mais présentant des déviations pour les petites valeurs. Dans le cadre de notre travail, nous avons systématiquement utilisé le même type de loi pour modéliser ce phénomène :

$$P(X = k) = \beta(k + \mu)^{\alpha}$$

où β est la constante de normalisation, μ est un réel arbitraire qui permet d'obtenir le degré moyen voulu, et α est l'exposant de la loi de puissance. En pratique, nous déterminons les valeurs adéquates de β et de μ pour un exposant α et une moyenne z voulus par calcul numérique.

[7] En effet, l'espérance d'une loi de probabilité discrète sur k proportionnelle à $k^{-\alpha}$ est donnée par $z = \left(\sum_{k=1}^{\infty} k \cdot k^{-\alpha}\right) / \sum_{k=1}^{\infty} k^{-\alpha} = \zeta_{\alpha-1}/\zeta_{\alpha}$.

1.4.3 Connexité

On dit que deux sommets a et b d'un graphe $G = (V, E)$ sont connectés quand il existe un *chemin* de a à b dans le graphe, c'est-à-dire une suite de sommets commençant par a et finissant par b telle que deux sommets consécutifs soient reliés par une arête. Plus formellement, cette condition se note :

$$\exists\, v_0, \cdots, v_l \in V \mid v_0 = a, v_l = b \text{ et } \forall i \in \{1, \cdots, l\},\ (v_{i-1}, v_i) \in E$$

L'entier l est la *longueur* du chemin.

Dans les graphes non orientés, il est clair que si a est connecté à b, alors b est connecté à a. De plus, tout sommet est connecté à lui-même (par un chemin de longueur nulle), et si a est connecté à b et b est connecté à c, alors a est connecté à c.

Pour chaque sommet v du graphe, l'ensemble des sommets qui lui sont connectés, noté C_v, forme une *composante connexe* : à l'intérieur de cette composante connexe, tous les sommets sont connectés entre eux. Tout graphe peut se décomposer en composantes connexes, qui forment une partition de l'ensemble des sommets. On dit qu'un graphe est *connexe* quand il ne possède qu'une seule composante connexe, qui contient donc tous les sommets.

On peut, en comparant deux graphes, comparer leur connexité (sont-ils tous les deux connexes ?). On peut aller plus loin (dans le cas de graphes non connexes) et comparer la répartition des tailles des composantes connexes, par exemple. Dans beaucoup de grands graphes, qu'ils soient synthétiques ou tirés d'exemples concrets, la plus grande composante connexe joue un rôle particulier car elle rassemble beaucoup plus de sommets que les autres composantes : on parle alors de la *composante connexe géante*. Si la répartition des tailles des composantes connexes est trop dure à comparer entre deux graphes, on pourra se limiter à comparer la taille de leurs composantes connexes géantes respectives. Cette taille peut être exprimée en termes extensifs, c'est-à-dire en nombre de sommets, ou en termes intensifs : on parlera de la proportion de sommets appartenant à la composante connexe géante. Ajoutons qu'on affine parfois la donnée la taille de la composante connexe géante par celle de la deuxième composante la plus grande.

Introduisons également le concept d'*arbre*, ou graphe arborescent. Un arbre est un graphe connexe particulier, car il possède le nombre minimal d'arêtes nécessaire pour connecter le graphe, à savoir $N - 1$ arêtes si N est le nombre de sommets. Un arbre ne possède aucun cycle, c'est-à-dire aucun chemin partant et arrivant au même sommet. Un résultat classique de la théorie des graphes montre que ces deux dernières propriétés sont équivalentes, et sont nécessaires et suffisantes pour montrer qu'un graphe est arborescent.

1.4.4 Distance

Nous venons d'introduire la notion de chemin dans un graphe. On peut définir la *distance* $d(a, b)$ entre deux sommets a et b comme la longueur du ou des plus courts chemins entre a et b. S'il n'existe pas de chemin entre a et b (si a et b ne sont pas connectés), on pose $d(a, b) = \infty$.

L'ensemble des distances entre chaque paire de sommets est représenté par la matrice des distances D, qui est une matrice carrée indexée par les éléments de V et dont les éléments $D_{i,j}$ correspondent aux $d(i,j)$. Pour comparer qualitativement deux grand graphes, plutôt que de comparer leur matrice d'adjacence, on pourra se restreindre à plusieurs agrégats statistiques, définis à partir de l'ensemble des couples de sommets connectés $E^* = \{(i,j) \in V \times V \,|0 < d(i,j) < \infty\}$:

- Le *diamètre* d_{max} d'un graphe est la distance maximale entre deux sommets connectés :
$$d_{max} = \mathrm{Max}_{(a,b)\in E^*} d(a,b)$$
- La distance moyenne $\overline{d}$ d'un graphe est la moyenne de la distance entre deux sommets connectés :
$$\overline{d} = \frac{1}{|E^*|} \sum_{(a,b)\in E^*} d(a,b)$$

Citons également l'*excentricité* $exc(v) = \mathrm{Max}_{w\in C_v}(d(v,w))$ ou l'*éloignement moyen* $d_{moy}(v) = (1/|C_v|)\sum_{w\in C_v} d(v,w)$ d'un sommet, que nous n'utiliserons pas dans cette thèse. Ces définitions se simplifient si on se restreint aux graphes connexes : la distance est alors finie entre chaque couple de sommets et $E^* = V \times V \setminus \{(v,v), v \in V\}$.

Sur de nombreux réseaux rencontrés en pratique, on parle de l'*effet petit monde*, qui exprime le fait que les sommets sont étonnamment proches par rapport à la grande taille du réseau et sa faible densité. La distance moyenne et le diamètre sont effectivement très petits par rapport à N sur la très large majorité des réseaux rencontrés en pratique, ils sont en fait plutôt proches de $\log N$.

1.4.5 Coefficient de Clustering

Le clustering[8] mesure la propension qu'ont certains sommets à former des groupes bien interconnectés, par opposition à un branchement aléatoire avec les autres sommets du graphe [113, 11]. Inspirée du principe "les amis de mes amis sont mes amis", une première définition du coefficient de clustering $c(v)$ d'un sommet v est la probabilité que deux de ses voisins soient liés entre eux :

$$c(v) = \frac{|E \cap \mathrm{Vois}(v)^2|}{deg(v)(deg(v)-1)}$$

Le coefficient de clustering global du graphe est alors la moyenne des coefficients de clustering de ses sommets. Il existe cependant une variante qui compte le nombre total de triangles (triplets de sommets liés deux a deux) et le compare au nombre de triplets connectés :

$$c' = \frac{|\{(a,b,c) \in V^3 | (a,b),(b,c),(c,a) \in E\}|}{|\{(a,b,c) \in V^3 | (a,b),(b,c) \in E\}|}$$

[8] Souvent traduit par "densité locale" ou "cliquicité", l'usage est toutefois d'utiliser le terme anglo-saxon.

Ces deux définitions souffrent toutefois de leur limitation aux seules considérations des triangles, alors que certains graphes particuliers peuvent présenter des zones denses où les triangles restent relativement rares. De nombreuses variantes moins locales existent, jouant le même rôle mais de natures différentes [80, 5, 29]. Citons pour exemple la probabilité de bouclage autour d'un sommet v, qui mesure la probabilité qu'une marche aléatoire partant du sommet v et de longueur fixée repasse par v. Plus v se trouvera dans une zone dense, plus la marche aléatoire aura de chance de rester confinée dans cette zone et plus cette probabilité sera grande.

Sur les réseaux rencontrés en pratique, le clustering est souvent présent : les voisins d'un sommet ont effectivement tendance à être voisins, ou plus généralement des groupes de sommets se trouvent davantage interconnectés entre eux qu'avec les autres : on dit qu'ils forment des communautés (ou *clusters* en anglais, d'où le terme *clustering*). La notion de clustering reste mal comprise, et il est difficile de comparer finement deux graphes à partir de cette propriété. Le coefficient de clustering est souvent employé à cet effet, mais le résultat se résume essentiellement à compter les triangles, et ne doit pas être pris comme un réel résultat sur le "taux de regroupement" des sommets en communautés.

1.4.6 Autres propriétés

Les propriétés citées ci-dessus constituent le panel de base [80, 5, 29] pour l'analyse d'un graphe. Il existe de nombreuses autres mesures plus complexes, et nous ne chercherons pas à toutes les introduire ici. Présentons tout de même brièvement un certain nombre d'entre elles :

- La *centralité* [112, 39] d'un sommet mesure la probabilité qu'aura un plus court chemin pris au hasard dans le graphe de passer par ce sommet.
- L'*assortativité* [71] des degrés d'un graphe peut se voir comme la propension des sommets de fort degrés à se lier plutôt aux autres sommets de fort degré plutôt qu'aux sommets de faible degré.
- En allant plus loin, l'analyse d'un graphe par *dk-series* [68] consiste à mesurer sa distribution de degrés, la distribution des couples de degrés de sommets liés par une arête, puis la distribution des triplets de degrés de sommets connectés, et ainsi de suite.
- La *robustesse* [49] d'un graphe mesure sa capacité à rester connecté malgré le retrait de sommets ou d'arêtes. Ce retrait peut être dû à des évènements aléatoires (comme des pannes) ou ciblés (comme des attaques).

1.4.7 Récapitulatif

Il est important de retenir que les réseaux rencontrés en pratique ont des propriétés non triviales en commun, comme nous l'avons détaillé dans cette section. Nous rassemblons ici les principaux résultats qui concernent la large majorité de ces réseaux :

- Le degré moyen est faible par rapport à leur taille.

- La distribution de degrés est très hétérogène : la plupart des sommets ont très peu de voisins et néanmoins quelques sommets en ont un nombre considérable. Le comportement asymptotique de cette loi ressemble à une loi de puissance dont l'exposant est le plus souvent compris entre 2 et 3.
- Les distances entre sommets sont faibles, de l'ordre du logarithme de la taille du graphe.
- Le coefficient de clustering est relativement élevé, en tout cas par rapport à ce que donnerait un branchement aléatoire des sommets entre eux.

1.5 Modèles de topologies

Nous allons présenter dans cette section des modèles de topologie *généralistes* qui décrivent la forme qu'aura un graphe, comment il est construit, et permettent de *générer* de tels graphes.

La généralité des modèles abordés ici leur permet de s'appliquer à de nombreux types de réseaux, sans se restreindre au cas particulier de l'Internet. Ce choix est délibéré, notre travail de thèse s'étant porté à l'origine vers les grands réseaux d'interactions en général avant de se focaliser sur l'Internet, mais aussi à cause des propriétés intéressantes des modèles généralistes, moins réalistes mais plus "propres" et moins réfractaires à l'analyse que les modèles spécifiques à l'Internet.

1.5.1 Tirage aléatoire

Un graphe purement aléatoire se construit à partir d'un ensemble de N sommets initialement non connectés et d'un réel p appelé *probabilité de connexion*. Pour chacune des $N \cdot (N-1)/2$ paires de sommets $\{i, j\}$, on crée l'arête $(i - j)$ avec probabilité p, et on laisse i et j non connectés avec probabilité $1-p$. Dans le cas de grands graphes, c'est-à-dire pour N grand, le degré moyen du graphe obtenu se situe très près de sa valeur attendue $\hat{z} = p \cdot (N-1)$. Cela permet donc de générer un graphe de taille fixée et de degré moyen voulu. Le graphe obtenu est aléatoire puisqu'il correspond à un graphe tiré au hasard uniformément dans sa classe de graphes, mais possède en fait une propriété aléatoire encore plus forte : toute arête de $V \times V$ peut exister avec la même probabilité. Cette propriété permet d'appliquer de nombreux outils analytiques, ce qui rend les graphes aléatoires de ce type bien connus. On peut par exemple prédire la taille des composantes connexes, la distance moyenne, le coefficient de clustering et la distribution de degrés attendus [33, 80]. On constate que la distance moyenne des grands graphes aléatoires est petite, qu'il existe une composante connexe géante, que le clustering est faible, et que la distribution des degrés est homogène puisqu'elle suit approximativement une loi de Poisson.

Notons qu'une variante propose de fixer le nombre *exact* d'arêtes du graphe puis de les assigner aléatoirement. Cette variante ne change que très peu les diverses propriétés du graphe par rapport au modèle décrit ci-dessus.

Un des principaux défauts du modèle aléatoire simple est l'homogénéité des degrés obtenus, souvent décalée par rapport aux réseaux rencontrés en pratique. Un autre modèle bien étudié, le *configuration model*, propose de générer un graphe aléatoire ayant une séquence de degrés fixée. En première approximation, le branchement peut être considéré comme aléatoire, et la probabilité que deux sommets soient reliés est alors directement proportionnelle au produit de leurs degrés. Les possibilités analytiques restent larges avec ce modèle et dans le cadre de cette approximation : la taille de la composante connexe géante, le coefficient de clustering et la distance moyenne attendus peuvent être déterminés à partir de la séquence de degrés [76, 77, 80].

D'autres extensions ont été créées, par exemple pour générer des graphes aléatoires avec une séquence de degrés et des coefficients de clustering donnés [47, 29]. Nous ne les détaillons pas car ils ne sont pas utilisés dans cette thèse.

Le point important ici est que le caractère aléatoire de ces modèles, en plus de rendre l'analyse plus facile, permet d'obtenir un graphe **non biaisé**. Cette garantie est précieuse, comme nous le verrons par la suite. La contrepartie est que ces modèles restent synthétiques et souvent très loin de la réalité.

1.5.2 Modèles de croissance

Une autre famille de modèles s'appuie sur la dynamique des réseaux réels. Ces modèles tentent de capturer la dynamique d'évolution du réseau, puis l'appliquent de manière incrémentale pour générer un graphe représentatif du réseau observé.

Le modèle d'attachement préférentiel [9] en fait partie. Il modélise l'arrivée de nouveaux sommets les uns après les autres, chacun créant à son arrivée un nombre prédéfini k d'arêtes en choisissant ses voisins aléatoirement parmi les sommets déjà présents avec une probabilité proportionnelle au degré de ces derniers. À la fin, le graphe a un degré moyen $z = \frac{2M}{N} = 2k$. Une des propriétés remarquables de ce modèle est qu'il génère des graphes ayant une distribution de degrés proche asymptotiquement d'une loi de puissance d'exposant 3, ce qui est souvent beaucoup plus proche des réseaux réels qu'une distribution homogène. De nombreuses variantes se sont inspirées de ce modèle pour, entre autres, obtenir des exposants de loi de puissance variables, ou encore des degrés mieux répartis pour les petites valeurs (notamment en deçà de k).

Une variante intéressante, celle de Dogorovtsev et Mendes [29], produit des graphes ayant un fort clustering, ce qui n'est pas le cas du modèle d'attachement préférentiel de base. Ce modèle consiste, dans sa version la plus simple, à relier tout nouveau sommet aux deux extrémités d'une arête prise au hasard. Cela revient qualitativement à un attachement préférentiel, puisque la probabilité qu'un sommet de degré k soit à l'extrémité d'une arête prise au hasard est essentiellement proportionnelle à k.

Ces types de modèles sont souvent faciles à mettre en œuvre, mais restent difficile à analyser. Par opposition aux graphes aléatoires, ils sont également biaisés par leur méthode de construction qui leur donne des propriétés très particulières. Ces propriétés peuvent être voulues, comme la connexité, mais sont souvent non prévues et parfois indésirables : par

exemple, un graphe de Dogorotsev-Mendes est forcément planaire et un graphe obtenu par attachement préférentiel de k arêtes à chaque étape ne compte aucun sommet de degré inférieur à k. Cela induit des comportements imprévus lors des simulations mettant en jeu ces graphes, et il n'est jamais évident d'estimer la fiabilité des résultats obtenus quand on ne connaît pas toutes les particularités inhérentes au type de graphe utilisé. Pour exemple, les simulations faites au Chapitre 3 mettent en évidence un comportement très particulier du modèle d'attachement préférentiel.

1.5.3 Autres

Il existe une multitude de familles de modèles de topologie. Passons en revue les plus notoires :

- Les modèles hiérarchiques (voir par exemple [62] pour un modèle de la topologie de l'Internet) construisent un graphe incrémentalement, en s'occupant de groupes de sommets dans un ordre correspondant à un certain type de hiérarchie. Par exemple, un modèle de réseau des liaisons aériennes entre aéroports peut commencer par tracer les lignes transcontinentales, puis les moyen courriers, et enfin les vols nationaux ou régionaux.
- Les modèles positionnels (voir par exemple [95] et sa bibliographie) construisent un réseau dans un espace métrique à N dimensions en se basant sur les positions des sommets dans cet espace. Typiquement, deux sommets auront d'autant plus de chance d'être reliés qu'ils sont proches dans la métrique donnée.
- Les modèles réguliers construisent des graphes à partir de schémas prédéfinis, tels qu'une grille ou un anneau. Leurs variantes, tels les *graphes petit-monde* [61], utilisent de telles bases régulières et les modifient, par exemple en les augmentant d'arêtes aléatoires.

Beaucoup de modèles sont le fruit de la combinaison de deux de ces concepts, ou plus. Au final, plus un modèle est complexe, mieux il pourra imiter la réalité, mais en étant plus difficile à analyser formellement et potentiellement biaisé.

1.5.4 Bilan

La diversité des modèles s'explique. Tout d'abord, il existe 2 approches diamétralement opposées : tenter de modéliser un réseau en reproduisant toutes ses particularités ou modéliser un réseau le plus simplement possible en s'occupant uniquement d'un ensemble 'important' de propriétés qu'on essayera de reproduire au mieux. Ensuite, les réseaux réels ont de multiples natures. Une des conséquences est l'apparition de nombreuses familles de modèles, notamment celles s'inspirant de la construction du réseau *au cours du temps*. Notons qu'un modèle créé en s'inspirant d'un certain type de réseaux observés dans la réalité peut parfois être adapté à la modélisation de réseaux de natures profondément différentes.

Parmi la multitude de modèles envisageable, comment choisir le bon ? Cela dépend des besoins et des intérêts de chacun. Nous avons retenu la facilité d'implantation, l'universalité (la capacité à modéliser des types de réseaux totalement différents), et enfin la rigueur. Le

type de modèle idéal dans ces conditions est le tirage d'un graphe aléatoire parmi l'ensemble des graphes vérifiant un certain nombre de propriétés "clés". Parmi les propriétés à imiter, la distribution de degrés semble un bon candidat. Le *configuration model* ou ses variantes constitue donc notre candidat favori. Nous le développerons dans le Chapitre 2.

1.6 Mesure de la topologie IP

1.6.1 Qu'attend-on d'une mesure de l'Internet ?

Comme déjà discuté en Section 1.2, la mesure de l'Internet peut servir à différentes fins, notamment dans le cadre de simulations et donc de modélisations du réseau. Nous avons vu en Section 1.5 que certains modèles se focalisent sur un certain nombre de propriétés du réseau qui doivent donc être déterminées de manière précise pour obtenir une image "fidèle" dans le cadre du modèle. La mesure peut également servir à des fins de diagnostic, dimensionnement, ou autres problématiques ne passant pas forcement par une modélisation du réseau.

On peut aborder la mesure de l'Internet de deux manières différentes :

1. Soit on cherche à rassembler le maximum de données, à obtenir une mesure la plus *grande* possible : la représentativité de notre mesure peut alors être mesurée par son *taux de couverture*, c'est-à-dire la proportion d'éléments du réseau (sommets, arêtes) découverts par la mesure.
2. Soit on cherche à mesurer une ou plusieurs quantités liées au réseau le plus précisément possible. Bien que des mesures plus grandes donnent en général une meilleure information, la *qualité* des mesures prend le pas sur la quantité. La représentativité d'une mesure sera alors quantifiée par la faiblesse du biais qu'elle introduit dans la mesure des quantités concernées, ou par notre capacité à corriger ce biais.

1.6.2 Les outils de base

L'Internet n'a pas été créé avec dans l'esprit la contrainte claire d'être mesurable dans sa globalité, du moins pas sans avoir un contrôle direct sur les machines qui le composent. Les outils permettant de mesurer la topologie IP s'en trouvent réduits en nombre.

L'outil `traceroute`

L'outil `traceroute` [102] s'appuie sur un mécanisme assez largement respecté par la plupart des routeurs : l'expiration du temps de vie d'un paquet transitant dans l'Internet. Chaque paquet IP contient un champ appelé TTL (*Time To Live*), dont la valeur maximale est 255 et qui se voit décrémenté de 1 à chaque fois qu'il transite par un routeur. Lorsque ce champ arrive à 0, le routeur qui l'a décrémenté à 0 doit envoyer à l'émetteur du paquet un message d'erreur lui indiquant que son paquet a été supprimé.

En pratique, selon nos expériences[9], la quasi-totalité des routeurs `IP` agissent de la sorte, car omettre cette précaution serait risquer de laisser des paquets errer indéfiniment dans le réseau, provoquant à terme l'encombrement de ce dernier.

Certains n'envoient par contre pas toujours, voire jamais, le message d'erreur à l'émetteur du paquet, mais nos expériences ont montré que la quasi-totalité d'entre eux l'envoient comme prévu – au moins de manière sporadique.

L'outil `traceroute` exploite ce mécanisme pour déterminer la route suivie par un paquet depuis un hôte source *src* vers une destination arbitraire *dst*, à savoir la séquence des routeurs traversés par le paquet entre *src* et *dst*, ceux-ci étant identifiés par leur adresse `IP`. Le fonctionnement de `traceroute` est schématisé en Figure 1.7. Il envoie d'abord un paquet pourvu d'un TTL égal à 1 à destination de *dst*. Ce paquet est supprimé par le premier routeur présent sur la route de *src* à *dst*, qui envoie à *src* un message d'erreur. L'adresse du routeur étant contenue dans le message d'erreur, `traceroute` peut déterminer l'adresse du premier routeur sur la route vers *dst*. Il envoie ensuite un paquet avec $TTL = 2$ pour récupérer l'adresse `IP` du deuxième routeur, et ainsi de suite jusqu'à ce que le paquet envoyé atteigne la destination. Nous reviendrons en détail sur les subtilités du fonctionnement de `traceroute` quand cela sera nécessaire, dans le Chapitre 4.

Au final, l'outil `traceroute` permet donc de déterminer la route – c'est-à-dire la séquence de routeurs – traversée par un paquet allant vers n'importe quelle destination. Cela procure une information sur la topologie `IP`, puisque toutes les paires de routeurs consécutifs sur la route obtenue correspondent nécessairement à un lien `IP`. On obtient en fait un *chemin* de *src* vers *dst* dans le graphe de la topologie `IP`. L'outil `traceroute` n'est toutefois pas infaillible : comme nous l'avons déjà dit, certains routeurs ne répondent pas et le chemin obtenu est alors incomplet. Nous verrons même dans le chapitre 4 que `traceroute` peut donner des informations trompeuses. Retenons pour l'instant que la mesure topologique donnée par `traceroute` est partielle, et peut même être incorrecte.

Autres outils

D'autres outils permettent de sonder directement ou indirectement, la topologie `IP`. N'étant pas liés directement à l'objet de cette thèse, nous nous contentons de citer les principaux :

- L'exploitation des tables BGP [6, 124], complexe à mettre en œuvre.
- L'utilisation de l'option *source route* des paquets `IP`, qui n'est malheureusement pas supportée par la majorité des routeurs [44].
- Le sondage par envoi récursif de messages IGMP aux routeurs multicasts pour leur demander de communiquer la topologie de leur voisinage [83].
- Le sondage passif du trafic transitant par un ensemble de routeurs, qui doivent donc être contrôlés par l'auteur de la mesure.
- A l'échelle d'un AS, de nombreuses techniques spécifiques au matériel et à l'architecture de l'AS peuvent être envisagées.

[9] voir le Chapitre 4

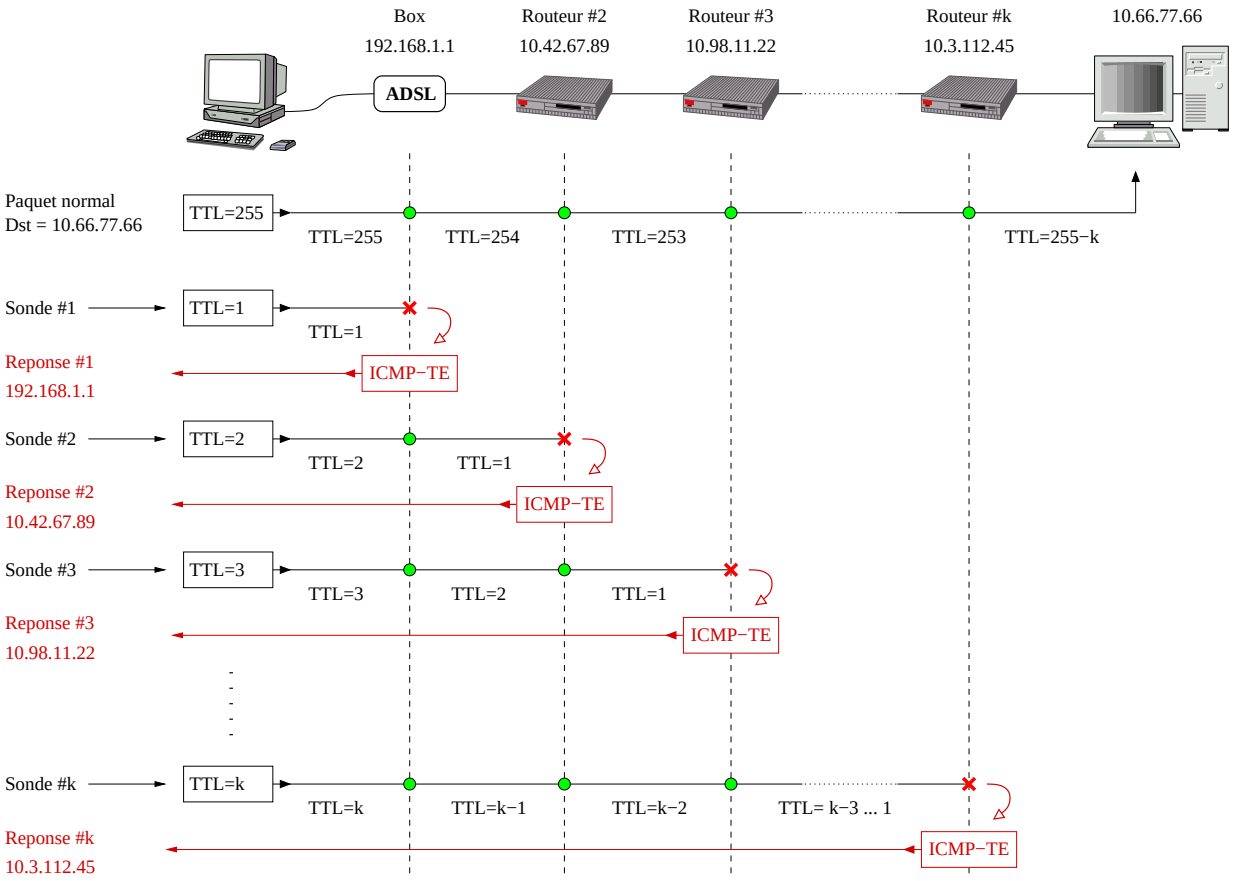

FIG. 1.7 – Fonctionnement schématique de l'outil `traceroute`. Les petits disques représentent une transmission normale et les croix signalent la suppression du paquet et l'émission d'une réponse ICMP *Time Exceeded*.

1.6.3 Mesurer à grande échelle avec `traceroute`

Nous allons montrer ici comment utiliser l'outil `traceroute`, dont le comportement a été décrit dans la section précédente, pour recueillir la topologie de manière organisée.

Méthodes

Pour collecter une mesure significative de la topologie `IP`, lancer un seul `traceroute` ne suffit pas : il faut regrouper l'information collectée par de multiples `traceroute` vers de multiples destinations. La méthodologie est alors un point important, car nombreuses sont les possibilités d'organiser et de regrouper des mesures `traceroute` (voir Figure 1.8).

Supposons d'abord que nous ne disposons que d'une seule source potentielle. Une première option est la simple agrégation des données obtenues par des `traceroute` vers chaque élément d'un ensemble D de destinations : si D est grand, la fusion de tous les chemins obtenus représentera une quantité importante de sommets et d'arêtes du graphe de la to-

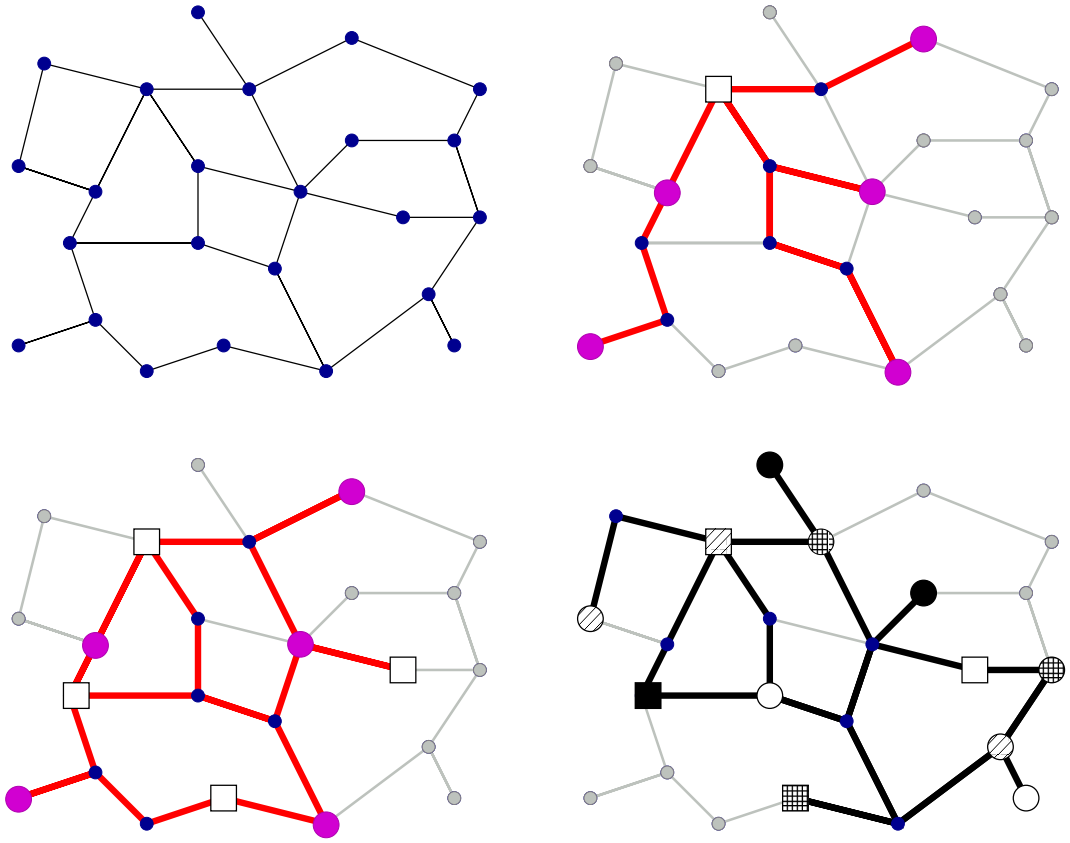

FIG. 1.8 – Différents schémas de sondage. Les sources sont représentées par des carrés, et les destinations par des nœuds de plus grande taille. Les parties du réseau non découvertes sont laissées en gris clair. De haut en bas et de gauche à droite : le réseau dans sa globalité, un sondage avec une source, avec plusieurs sources partageant les mêmes destinations, et enfin avec plusieurs sources ayant chacune leur propre liste de destinations (ici, deux destinations chacune).

pologie IP. Cependant, l'idée de ne lancer les `traceroute` qu'une seule fois est contestable : en effet, le routage évolue à court et long terme, et deux `traceroute` depuis la même source vers la même destination, même consécutifs, peuvent différer de manière significative. On peut donc également choisir de lancer des `traceroute` régulièrement vers les destinations de D en recommençant une fois que toutes les destinations ont été sondées. La mesure s'inscrit alors dans le temps.

Quand on dispose de plusieurs sources, les mêmes schémas peuvent s'appliquer. Reste alors le choix d'utiliser le même ensemble D de destinations pour chacune des sources, pour tenter d'acquérir le maximum d'information sur la topologie autour de ces destinations, ou au contraire d'utiliser des destinations différentes pour chaque source pour couvrir une zone plus grande de l'Internet. La redondance d'information n'est pas forcément un mal

puisqu'elle permet d'augmenter la confiance vis-à-vis de la topologie mesurée plusieurs fois à l'identique, ou au contraire de détecter la présence d'erreurs en cas de conflit.

Limitations

La première limitation de ce type d'étude vient de l'Internet lui-même : la création des paquets ICMP *Time Exceeded* qui signalent à la source que ses paquets ont expiré pendant le transit demande des ressources non négligeables aux routeurs, qui sont plutôt optimisés pour le transit des paquets classiques et traitent souvent les évènements spéciaux – comme l'expiration d'un paquet – par un circuit spécifique plus susceptible de saturation. Il est donc délicat de sonder répétitivement le même routeur, car la mesure se doit de ne pas être invasive pour le réseau. En pratique, deux politiques de protection existent : soit les administrateurs reçoivent une notification automatique lorsque le taux de paquets expirés en transit dépasse un certain seuil et semble donc anormal, soit les routeurs ont une limite interne et ne peuvent alors créer, par exemple, plus d'un message ICMP-TE par seconde. Le sondage répétitif de ce type de routeur ne récoltera de réponses que de manière sporadique.

Une autre limite vient du sondage par *la même source* d'une multitude de destinations. Cela peut aisément saturer la bande passante du lien – souvent unique – reliant la source à l'extérieur, puisque celui-ci est doublement surchargé : d'une part, chaque `traceroute` commence par sonder la même portion de lien reliant la source au reste de l'Internet, d'autre part, ce lien voit passer tous les paquets envoyés par tous les `traceroute`, et chaque réponse ! En sondant de la sorte, la quantité de paquets transitant par les liens proches de la source est proportionnelle au produit du nombre de destinations par la distance moyenne de ces destinations, ce qui dévient vite trop gros. Nous montrons en Figure 1.9 une représentation figurée de la bande passante utilisée dans le réseau par un sondage comme nous l'avons décrit. De même, si plusieurs sources sondent les mêmes destinations, une saturation peut apparaître du côté des destinations.

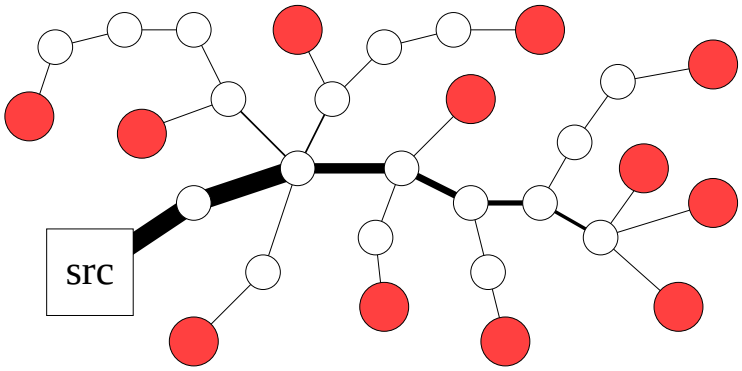

FIG. 1.9 – Bande passante requise par des utilisations de `traceroute` depuis une même source (le nœud *src*) vers plusieurs destinations (en rouge foncé). L'épaisseur des liens est directement proportionnelle au nombre de sondes transitant par eux.

La limitation venant des routeurs peut être traitée en ralentissant le rythme de sondage, ou en évitant d'envoyer trop de sondes vers les mêmes routeurs au cours d'une même passe. Des techniques sophistiquées existent [28, 111] pour éviter la redondance, et ces techniques réduisent également la demande de bande passante aux liens les plus sollicités d'un facteur non négligeable correspondant à la distance moyenne parcourue par les paquets

Pour ces raisons, les mesures `traceroute` sont limitées sur deux points : leur cadence, et le nombre de destinations par source. Adopter des valeurs raisonnables permet de rester non invasif pour le réseau. Les grands projets de mesures de la topologie `IP` prennent en compte ces limitations, qui empêchent notamment de sonder la totalité du réseau en un temps raisonnable, c'est-à-dire moins de quelques années. Étant donné la dynamique de l'Internet, l'obtention d'une carte complète et à jour semble utopique dans l'état actuel des choses.

Projets existants

Différents projets ont déjà suivi ces schémas. Après les premiers du genre dans les années 90, qui sondaient des nombres de destinations relativement restreint [84, 35]. Le projet MERCATOR [44], l'un des pionnier du sondage de la topologie `IP` à grande échelle, utilisait une méthode de découverte de la topologie à partir d'une seule source, en fusionnant les résultats obtenus au cours de la mesure. Le projet Skitter [126] dispose de plus d'une vingtaine de sources éparpillées dans le monde, et scanne régulièrement (une passe dure environ 2 jours) les routes vers 10^6 destinations, depuis l'ensemble de ses sources. Le projet DIMES [96, 122] est l'exemple même du projet multi source à grande échelle : il consiste à distribuer aux particuliers un programme qui tournera en tâche de fond, lancera des `traceroute` et enverra les résultats à un serveur qui s'occupera d'agréger les données de tous les utilisateurs. Certains projets comme Traceroute@Home [127] s'intéressent au bagage théorique nécessaire à ce type de mesure et à leur optimisation, sans forcément se focaliser sur le déploiement de leur propre infrastructure de mesure. D'autres projets de mesure existent [67, 100, 125, 123, 121], mais nous ne les détaillerons pas.

La mesure étant loin d'être parfaite, opter pour la masse de donnée la plus grande n'est pas forcément la meilleure solution, surtout si l'on considère les difficultés pratiques : il est infiniment plus long de mettre en place un projet tel que DIMES que de lancer une exploration à partir d'une seule ou de quelques sources. Nous verrons également que sur certains points, une carte "trop" complète peut s'avérer plus dure à analyser qu'une autre.

1.6.4 Résultats des mesures `traceroute`

Aliasing

Nous décrivons ici plus en détail le comportement des routeurs lorsque ceux-ci reçoivent des paquets dont le TTL a expiré, et discutons des conséquences sur les cartes de topologie `IP` collectées grâce à `traceroute`. Les routeurs appartenant à des entités indépendantes, nous nous référons au consensus donné par la RFC 1812, *Requirements for IP Version*

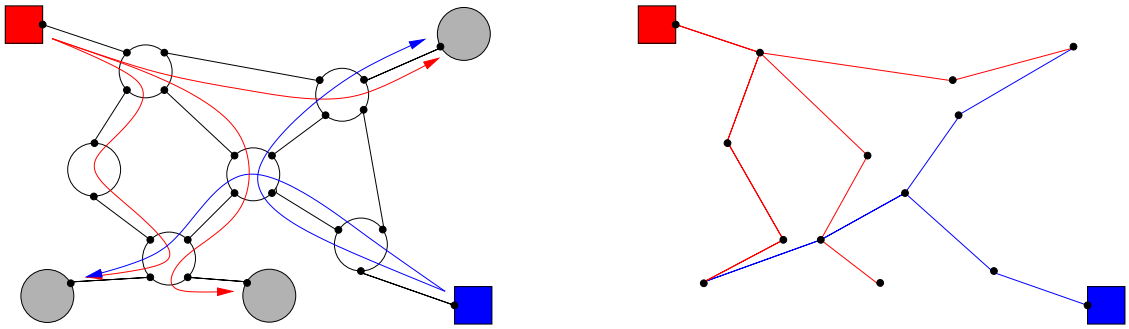

FIG. 1.10 – Illustration de la différence entre réseau physique et graphe de la topologie IP telle qu'elle est vue par `traceroute`, sur un exemple de sondage avec 2 sources (les carrés) et 3 destinations (en grisé). À gauche, le chemin pris par les sondes est représenté par des flèches. À droite, la carte IP telle qu'elle est collectée. Chaque routeur dispose d'un certain nombre d'interfaces (les petits ronds noirs), qui correspondent chacune à une adresse IP distincte.

4 Routers [8] quant à leur fonctionnement, et supposons qu'idéalement tous les routeurs suivent les règles données dans ce document.

Un routeur est un carrefour de câbles, et possède donc une certaine collection de liens entrants, sortants ou les deux, chacun reliés à une interface (carte réseau). Ces interfaces possèdent *a priori* chacune une adresse IP distincte. Lorsqu'un paquet dont le TTL est à 1 pénètre dans le routeur par une de ses interfaces, le routeur supprime le paquet puisque son TTL a expiré et envoie un message d'erreur qui sera émis en principe par l'interface ayant reçu le paquet. L'adresse de réponse des routeurs que l'on voit dans la sortie de `traceroute` correspond donc à l'adresse IP de l'interface par laquelle le paquet sonde est *entré dans le routeur*.

Une adresse IP est bien unique dans le sens que deux interfaces visibles sur l'Internet ne peuvent posséder la même adresse IP, mais nous venons de voir qu'un même routeur possède plusieurs adresses IP. Cela pose un problème évident lors de la conduite de mesures `traceroute` avec plusieurs sources, car lorsqu'un routeur est sondé par plusieurs sources dont les `traceroute` arrivent par des chemins différents, les sources voient chacune une adresse IP différente, et le routeur apparaît comme un ensemble de sommets sur le graphe de la topologie IP. De nombreuses techniques existent [44, 98] pour résoudre ce type de problème et reconnaître les adresses IP appartenant à un même routeur, mais ce problème reste un secteur de recherche à part entière et aucune solution totalement fiable et exhaustive n'est connue à ce jour. Dès lors, l'obtention de cartes rigoureuses de la topologie IP s'avère considérablement difficile dans le cadre de mesures multi sources. La conduite généralement tenue face à ce problème est de regrouper puis fusionner au maximum les adresses IP dont on sait qu'elles correspondent à un même routeur, puis de laisser le graphe tel quel, tout en connaissant son imperfection.

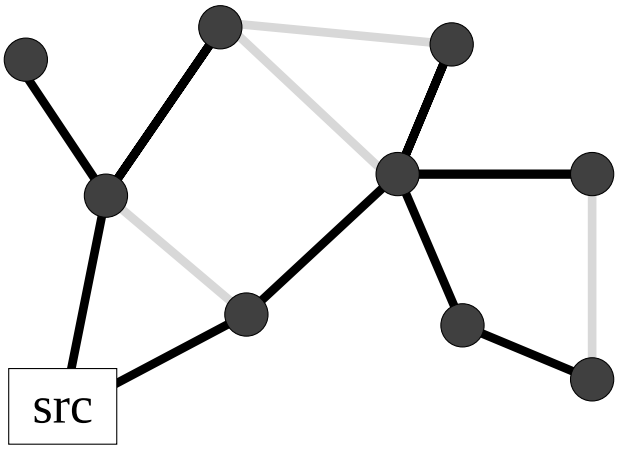

FIG. 1.11 – Partialité forcée des mesures de la topologie `IP` depuis une seule source (le rectangle) : les liens en gris ne sont jamais empruntés, même si on collecte les chemins vers tous les nœuds du graphe.

Des mesures nécessairement partielles

En plus de leur étendue forcément limitée par les contraintes vue en Section 1.6.3, les mesures de la topologie `IP` se heurtent à une barrière théorique les empêchant de recouvrir la totalité du réseau, du moins à partir d'une seule source. Considérons le réseau esquissé en Figure 1.11 : même si la source lance des `traceroute` vers l'ensemble des destinations, aucune route n'empruntera *a priori* d'arêtes transverses (les arêtes en gris clair sur la figure). Tous les sommets seront bien découverts, mais des liens seront manquants. L'utilisation d'autres sources permet de réduire ce problème, mais finalement le seul moyen d'être certain de ne manquer aucun lien de la topologie `IP` serait de lancer des `traceroute` depuis tous les nœuds du réseau vers tous les nœuds du réseau, ce qui est bien sûr infaisable.

De plus, la taille globale du réseau n'étant pas connue, il est difficile de se faire une idée de la représentativité des cartes dont on dispose. Une première technique pour contourner cette difficulté est l'observation chronologique des résultats obtenus par des mesures successives, de plus en plus complètes, et l'extrapolation d'un *seuil* représentant la connaissance totale du réseau. Il est clair qu'en menant continuellement de nouvelles mesures de la topologie `IP`, on découvre une partie du réseau de plus en plus grande. Mais quand doit-on s'arrêter ? Le problème est que plus l'image du réseau que l'on a est complète, moins les mesures apporteront d'informations supplémentaires. Au début de la mesure, un `traceroute` permettra de découvrir plus d'une dizaine de nouvelles adresses `IP` et un nombre équivalent de nouveaux liens, mais au fur et à mesure que l'image du réseau se complète, ce nombre diminue considérablement [10]. Nous montrons en Figure 1.12 le taux de découverte du réseau par le projet Skitter au cours du temps. Comme le suggèrent des travaux récents [65], il est difficile, au vu de ces courbes, d'anticiper un point où la quasi-totalité du réseau serait connue, en continuant à sonder de la sorte : des nouveaux nœuds et de nouvelles

arêtes apparaissent continuellement. La dynamique de l'Internet, qui croît à une vitesse non négligeable en regard du temps demandé par des mesures aussi imposantes que celles de Skitter, peut être mise en cause et compromettre là encore les chances d'évaluer le taux de couverture des cartes `IP` disponibles.

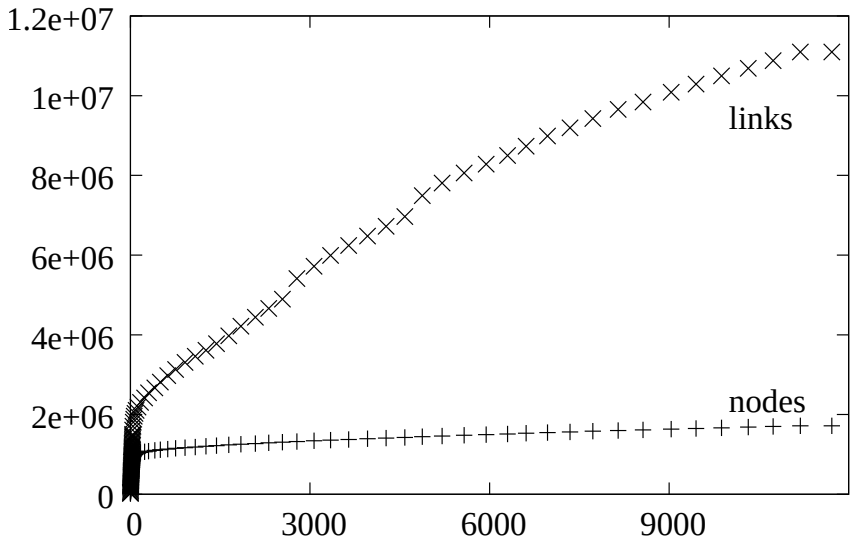

FIG. 1.12 – Evolution de la taille de la carte `IP` réalisée par Skitter, en nombre de sommets (+) et nombre d'arêtes (×), au cours des quelques centaines de passes réalisées entre janvier 2005 et mars 2006, en gardant à chaque passe les mêmes sources et mêmes destinations. L'axe des abscisses correspond au temps écoulé depuis le 1er janvier 2005 en heures.

De nombreux travaux traitent du degré de couverture des mesures de la topologie `IP`, et sont basés soit sur des considérations théoriques [20], soit sur des simulations [48]. Ces travaux défrichent le terrain en enquêtant sur la représentativité d'une mesure de la topologie en fonction de la *densité* du sondage, c'est-à-dire le rapport entre le nombre de sources, de destinations, et le nombre total de sommets du réseau. Dans notre cas, comme nous voulons avoir une idée de la représentativité de nos mesures mais que nous ne disposons précisément que de ces mesures partielles, sans informations sur le réseau global, la densité de sondage est inconnue, et les résultats cités ci-dessus ne sont pas applicables. Il paraît donc difficile d'estimer le degré de partialité des cartes de la topologie `IP` obtenues par `traceroute`. On sait qu'elles sont incomplètes, mais on ne sait pas *à quel point* elles le sont.

En plus de ces problèmes de fond, ajoutons qu'une grande partie de l'Internet reste invisible à tout sondage en raison de protection diverses : sous-réseaux NAT, pare-feu... De plus en plus de réseaux locaux deviennent imperméables au sondage par `traceroute`, qui doit alors se contenter de sonder le *cœur* du réseau.

Qu'en fait-on ?

De nombreuses mesures de la topologie de l'Internet à l'aide des outils décrits dans cette section ont été menées dans des travaux antérieurs et actuels. De ces travaux, un certain nombre d'observations *qualitatives* sont communément admises à propos du graphe de la topologie `IP` [80] :

- Le graphe est peu dense, avec un degré moyen faible (très grossièrement situé entre 2 et 10).
- Le diamètre et la distance moyenne sont faibles également : la topologie `IP` vérifie l'effet petit monde.
- La distribution des degrés des nœuds est hétérogène.
- Le clustering est fort.

En résumé, le graphe de l'Internet vérifie l'ensemble des propriétés usuelles des réseaux rencontrés en pratique (voir Section 1.4).

D'autres propriétés qualitatives observées sur les cartes `IP` ne s'appliquent pas forcément au graphe réel de l'Internet. Pour exemple, un fait auparavant communément admis était que la loi de distribution des degrés suit une loi de puissance, et ce même pour les faibles degrés (voir la Figure 1.6 en haut à gauche). Si cette assertion semble – approximativement – vraie pour les forts degrés, des travaux récents [86, 1] ont introduit le doute à propos des faibles degrés en montrant que des cartes collectées par des méthodes similaires à `traceroute` sur des graphes homogènes pouvaient exhiber une distribution de degrés en loi de puissance pour les degrés inférieurs au degré moyen. La *forme observée* de la distribution ne reflète donc pas nécessairement la forme réelle de la distribution. De fait, le caractère biaisé des mesures `IP` rend toute certitude difficile, même sur des conclusions qualitatives de ce type.

Les mesures quantitatives sont, quant à elles, sujettes à de nombreux problèmes. Le fait qu'on ne connaisse pas le degré moyen de l'Internet [109] est révélateur. Un problème plus grave est l'ignorance de la marge d'erreur ou la taille trop grande de celle-ci : en fournissant l'estimation d'une quantité, on ne pourra souvent pas garantir à quel point on peut potentiellement s'écarter de la réalité.

Le décalage entre ce qu'offre la mesure de l'Internet et les besoins de la modélisation est donc conséquent : si certaines observations qualitatives permettent de se faire une idée du *type* de modèle qui peut convenir pour l'Internet, le manque de déterminisme dans l'évaluation de paramètres clés (comme la distribution de degrés) pose un problème majeur à la modélisation réaliste de la topologie `IP`. Nous nous pencherons sur ce problème dans le Chapitre 3.

1.7 Positionnement

Nous avons décrit les grandes lignes de la composition et du fonctionnement de l'Internet, qui apparaît comme un réseau très complexe. Nous avons montré l'intérêt général que peut avoir une étude de l'Internet et une meilleure connaissance de celui-ci. Nous avons en-

suite focalisé notre attention sur la topologie de l'Internet – que nous désignons également par les termes *topologie* `IP` – et donc considéré ce dernier comme un graphe représentant les interconnexions entre routeurs.

Notre but global est de faire avancer la connaissance et les outils liés à l'étude de cette topologie. Nos travaux, présentés séparément par chapitres, répondent tous, partiellement et à leur manière, à la question : "Comment peut-on mieux connaître et modéliser l'Internet ?". Si la thématique est commune, les apports varient considérablement d'un sujet à l'autre, de l'algorithmique à l'expertise technique en passant par la statistique. Les problématiques abordées autour de la topologie `IP` seront souvent généralisables à des contextes plus larges, comme les grands réseaux d'interactions ou même l'algorithmique des graphes, et nous suivrons cette voie par la suite.

La variété et la richesse du champ d'investigations possibles dans le domaine des grands réseaux et particulièrement de la topologie de l'Internet expliquent en grande partie notre motivation dans cette thèse à aborder des sujets sensiblement différents en termes de contexte scientifique. Se restreindre à une seule approche nous aurait semblé trop réducteur et mal adapté à ce type d'étude.

Chapitre 2

Un modèle théorique simple

Nous nous intéressons dans ce chapitre à la mise en place *en pratique* de la génération aléatoire de graphes suivant le *configuration model* que nous avons rapidement introduit en Section 1.5.1. Il s'agit ici de générer un graphe aléatoire, simple et connexe, ayant une séquence de degrés prédéfinie. Le mot "aléatoire" signifie que le graphe doit être tiré uniformément parmi les candidats.

Nous décrivons d'abord le *configuration model* tel qu'il est le plus couramment employé [80, 81], et montrons les problèmes d'imprécision qu'il engendre. Nous nous intéressons dans le reste du chapitre à une variante plus rigoureuse, mais considérablement plus difficile à mettre en place en pratique. Nous introduisons ensuite le formalisme et les conventions utilisés dans ce chapitre, puis décrivons en détail l'algorithme le plus performant [40] implanté antérieurement à notre travail de thèse. Nous apportons un certain nombre de clarifications et de preuves de points théoriques délicats non évoqués précédemment, et améliorons significativement cet algorithme.

2.1 Contexte

2.1.1 Le *configuration model*

Le *configuration model* a été introduit dans les années 1970 et a été l'objet de nombreuses études depuis (voir [76, 77, 81, 4] et aussi [80] et sa bibliographie). L'idée est de modéliser un réseau en construisant un graphe *aléatoire* ayant les mêmes degrés que le réseau d'origine. Si les degrés exacts dans le réseau original ne sont pas connus, on pourra toujours les remplacer par une estimation, ou encore leur distribution probabiliste (voir la Section 1.4.2). Le réseau synthétique ainsi construit sera alors plus proche du réseau originel, tout en restant aléatoire.

Le modèle consiste donc simplement, étant donné une séquence de degrés $d_1, \cdots, d_N$, en la génération d'un graphe aléatoire de N sommets ayants exactement les degrés donnés. Notons que ce modèle construit des *multigraphes*, c'est-à-dire des graphes où les arêtes multiples et les boucles sont autorisées.

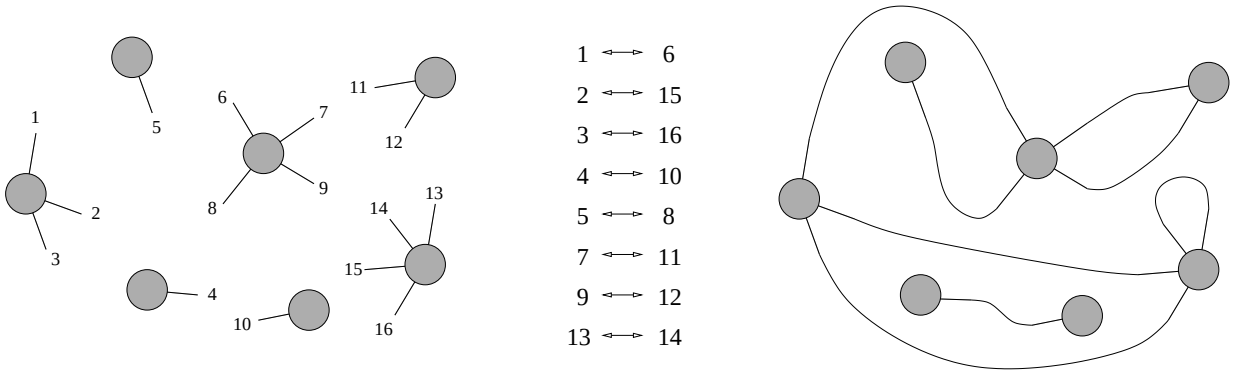

FIG. 2.1 – Génération de multigraphe aléatoire à degrés fixés par couplage aléatoire de demi-arêtes. À gauche : les sommets pourvus de leurs demi-arêtes, avant le couplage. Au centre : le couplage aléatoire. À droite : multigraphe généré après le couplage.

Algorithme de génération

En pratique, l'algorithme de génération se décrit simplement : on commence par attribuer à chaque sommet une demi-arête, puis on numérote chacune de ces demi-arêtes (il y en a exactement $\sum_{i=1}^{N} d_i$). Tant qu'il reste des demi-arêtes, on en choisit deux au hasard, qu'on connecte pour former une arête. Le processus s'arrête quand toutes les demi-arêtes ont disparu, remplacées par des arêtes : on obtient bien un multigraphe dont les degrés correspondent exactement aux valeurs voulues au départ. Le multigraphe est aléatoire, dans son mode de construction bien sûr, mais aussi dans un sens plus rigoureux : un multigraphe ainsi construit est équivalent au tirage aléatoire uniforme d'un multigraphe parmi l'ensemble des multigraphes ayant les degrés voulus. Notons que la somme des degrés $\sum_{i=1}^{N} deg(i)$ correspond exactement au double du nombre d'arêtes dans le multigraphe final (puisque chaque arête correspond à deux demi-arêtes). Il est en effet impossible de générer un multigraphe dont la somme des degrés est impaire, car un tel multigraphe n'existe pas. Par contre, si la somme des degrés est paire, le nombre de demi-arêtes l'est aussi et donc le processus décrit ci-dessus est sûr d'aboutir.

Nous montrons en Figure 2.1 la mise en pratique de cette technique pour un graphe de 7 sommets de degrés 3, 1, 2, 4, 5, 2, 4. Les demi-arêtes ont été numérotées en suivant l'ordre des sommets donné par la séquence de degrés.

Qualités et défauts

La première qualité de ce modèle est sa simplicité : en pratique, l'algorithme de génération est aisément applicable en temps et espace linéaire. Il possède également une propriété analytique puissante : dans un multigraphe ainsi généré, la probabilité que deux sommets a et b de degrés respectifs $deg(a)$ et $deg(b)$ soient connectés par au moins une arête est

donnée par l'équation suivante

$$P_{a \leftrightarrow b} = 1 - \left(1 - \frac{deg(a)deg(b)}{M(2M-1)}\right)^M \tag{2.1}$$

Cela permet de prédire statistiquement de nombreuses propriétés du multigraphe global, comme les tailles des composantes connexes principales, la distance moyenne, ou encore le clustering, comme décrit en [80]. Le fait de modéliser un graphe réel par un multigraphe aléatoire ayant des degrés fixés permet alors de mieux comprendre les corrélations entre les propriétés observées sur ces graphes et les degrés de leurs sommets. On a pu par exemple mieux comprendre l'effet *petit monde* ainsi [61]. En bref, le multigraphe aléatoire à degrés fixés n'est qu'une option parmi d'autres lorsqu'on veut travailler avec un modèle d'un réseau réel. Ce choix, qui peut sembler loin du cas concret, reste très pratique pour lancer des simulations et bénéficie d'une important base de connaissance quant à ses propriétés et son comportement [77, 76, 80, 4, 81].

Il comporte cependant un défaut souvent négligé : les arêtes multiples qu'il engendre. La plupart des réseaux rencontrés en pratique sont simples, et pour s'y conformer la pratique courante consiste à supprimer les arêtes multiples générées par le *configuration model*, en négligeant leur quantité puisqu'*a priori*, le branchement étant aléatoire, on peut espérer que deux même sommets ont assez peu de chance de se retrouver plusieurs fois connectés. Le problème est que le graphe ainsi obtenu ne respecte pas la séquence de degrés initiale, puisqu'on a enlevé des arêtes.

D'autres propriétés importantes du réseau réel peuvent être malmenées par le modèle. Nous nous sommes intéressés tout particulièrement à la connexité, car pour la quasi totalité des mesures de réseaux réels dont nous disposons, la connexité de l'ensemble du réseau est assurée, soit par la nature même du réseau, soit à cause de la méthode de mesure. Or le modèle de génération aléatoire à degrés fixés ne respecte pas la contrainte de connexité, comme on peut le voir sur la Figure 2.1. Là encore, on peut être tenté de contourner le problème en supprimant les petites composantes isolées et en gardant la composante connexe géante. Cela aura cette fois un impact sur le nombre de sommets, puisqu'on va en enlever, mais également sur la forme de la distribution de degrés.

Ces défauts sont particulièrement gênants quand on utilise le modèle dans des simulations, notamment quand ceux-ci requièrent la simplicité et/ou la connexité du graphe. Pourtant, sur le plan théorique, le modèle tel quel est beaucoup plus pratique à manipuler qu'un modèle de graphe *simple* ou *connexe* aléatoire à degrés fixés, car de telles contraintes imposent des corrélations entre les arêtes qui invalident la propriété fondamentale du modèle aléatoire pur (Équation 2.1). Cette divergence d'intérêt nous a poussé plus loin dans l'étude des différences entre ces deux modèles. Plus précisément, nous enquêtons dans la section suivante sur la légitimité de la démarche décrite plus haut : peut-on dire qu'un graphe aléatoire généré selon le *configuration model*, qu'on rend artificiellement simple et/ou connexe, est toujours aléatoire et surtout respecte la distribution de degrés initiale ?

Évaluation du biais

Nous étudions ici le biais induit sur le graphe lorsque l'on transforme un graphe généré par le *configuration model* en graphe simple et/ou connexe. Nous ne montrons que la déviation des deux paramètres extensifs principaux, les nombres de sommets N et d'arêtes M du graphe, et d'un paramètre intensif, le degré moyen z, mais celles-ci suffisent amplement à démontrer l'ampleur de la déviation subie. Dans le choix de nos graphes pour mener ces simulations, un paramètre crucial est la forme de la distribution de degrés, car c'est elle qui conditionne la déviation, comme nous le verrons dans la discussion des résultats à la fin de cette section. Dans un souci de réalisme et pour correspondre à notre propre usage du modèle, nous avons utilisé nos distributions de degrés en loi de puissance *décalées*, pour pouvoir obtenir des exposants et des moyennes arbitraires, comme expliqué à la fin de la Section 1.4.2.

Nous utiliserons les notations suivantes :

- G désignera un graphe généré selon le *configuration model*, ayant N sommets et M arêtes, de degré moyen $z = \frac{2M}{N}$.
- G_S désignera le graphe obtenu à partir de G en fusionnant toute ses multi-arêtes en arêtes simples et en enlevant les boucles.
- G_C désignera le graphe obtenu en ne gardant que la composante connexe principale de G.
- G_{CS} désignera le graphe obtenu en combinant les deux opérations : garder la composante connexe principale de G tout en fusionnant les multi-arêtes et en enlevant les boucles.

De même, les termes $N_C, M_C, z_C, N_S, M_S, z_S, N_{CS}, M_{CS}, z_{CS}$ correspondent aux nombres de sommets, d'arêtes, et aux degrés moyens des graphes G_C, G_S, G_{CS}. Notons que par définition, $N_S = N$ et $N_{CS} = N_C$ puisque le fait de rendre le graphe simple n'enlève aucun sommet.

Discussion

Avant d'examiner les figures en détail il convient de comprendre les phénomènes en jeu. La fusion des multi-arêtes et le retrait des boucles qui ont lieu lors du passage de G à G_S ont pour seul rôle le retrait d'arêtes, et enlèverons d'autant plus d'arêtes que la densité du graphe est grande (et donc que le degré moyen est grand). La restriction à la composante connexe principale enlève des petites composantes connexes, qui auront tendance à être moins denses que le reste du graphe car les sommets de fort degré se connectent plus facilement à la composante connexe géante. Elle induit donc une diminution du nombre de sommets, mais aussi des arêtes, et contribue globalement à une augmentation du degré moyen. L'effet se fera cette fois d'autant plus sentir que la composante géante est petite, ce qui correspond à une faible densité, et donc un petit degré moyen.

Nous observons les choses suivantes :

- Les tracés à gauche et au centre montrent clairement qu'une partie significative du graphe originel disparaît.

– La similitude entre les tracés du haut et du milieu montrent que la taille du graphe en elle seule a un très faible, voire aucun impact sur le biais provoqué qui semble donc dépendre uniquement de la distribution des degrés. La seule différence visible provient du fait que les tracés du milieu n'ont pas pu être moyennés sur autant d'instances que les autres à cause du temps de calcul trop important.
– Les tracés du bas sont plus proches de 1, montrant que le biais est moins significatif

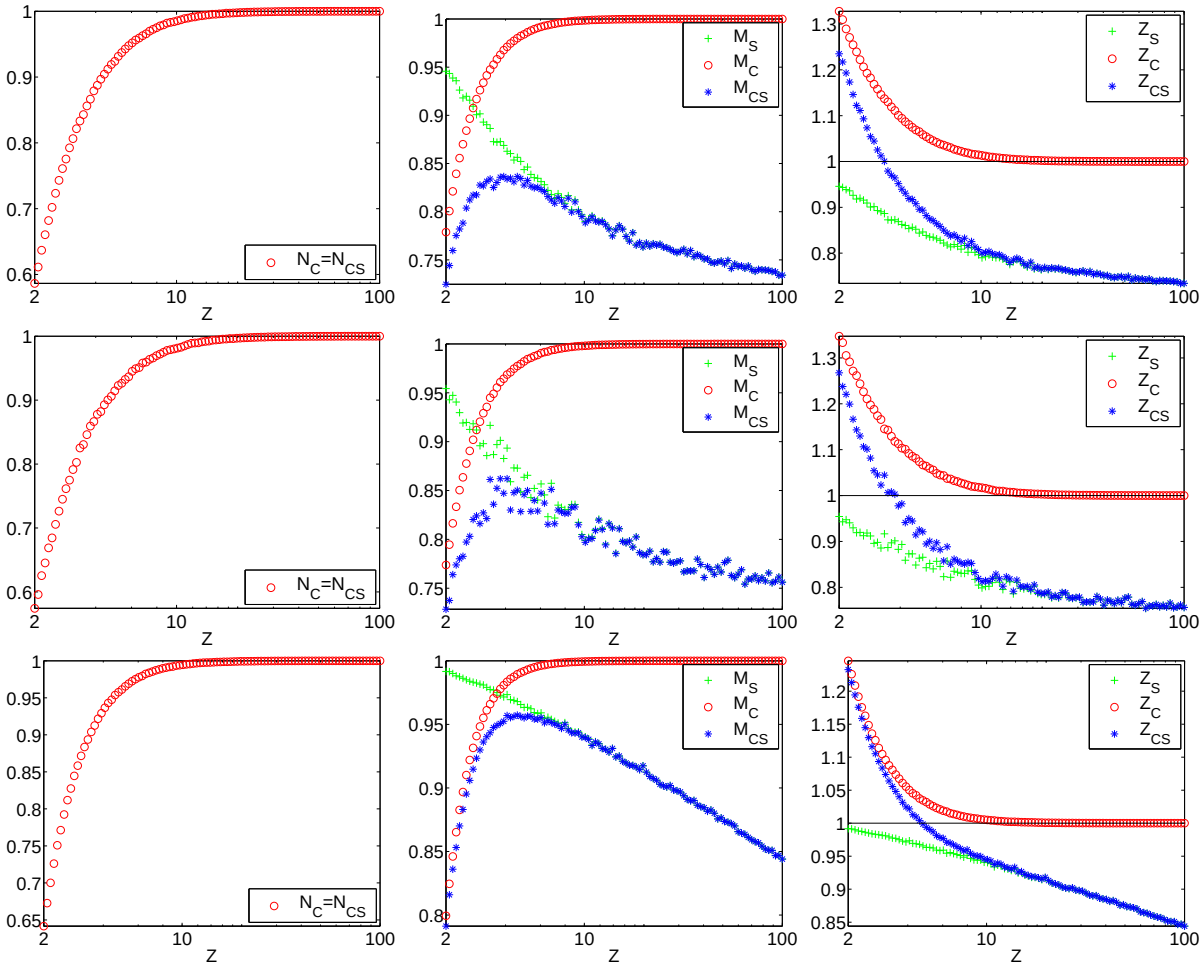

FIG. 2.2 – Comparaison du nombre de sommets N (à gauche), du nombre d'arêtes M (au centre) et du degré moyen z (à droite) dans les graphes G_S, G_C et G_{CS} par rapport au graphe d'origine G, pour différentes valeurs du degré moyen z dans le graphe d'origine. Les valeurs en ordonnées sont normalisées : les quantités initiales (respectivement N, M et z) sont ramenées à 1. Nous avons utilisé trois distributions de degrés : $N = 10^4, \alpha = 2.1$ (en haut), $N = 10^5, \alpha = 2.1$ (au milieu) et $N = 10^4, \alpha = 2.5$ (en bas). Les résultats sont moyennés sur 100 instances, sauf pour les tracés du milieu qui sont moyennés sur 10 instances à cause du coût de calcul trop élevé.

pour des distributions de degrés plus homogènes. Cela s'explique par le fait que moins de sommets ont un très faible degré et forment donc moins de petites composantes connexes isolées, et que moins de sommets ont un très fort degré ce qui engendre moins d'arêtes multiples.
– Les tracés de droite montrent bien les deux biais contradictoires qui modifient le degré moyen z du graphe, et leurs zones d'influence.

Pour des raisons essentiellement techniques, nous n'avons pas montré comment la distribution de degrés entière se voyait perturbée, nous concentrant sur le degré moyen. Il est clair qu'un biais provoquant une telle déviation du degré moyen n'est pas bénin. La déformation de la distribution de degrés peut se résumer à une homogénéisation des degrés puisqu'il y a à la fois moins de sommets de très faible degré et de très fort degré. Un tel biais se répercute sur la structure du graphe. Au final, une méthode basée sur l'utilisation d'un graphe généré de la sorte supprime le caractère rigoureusement aléatoire en introduisant des corrélations entre arêtes et provoque des changements importants dans la structure même du graphe. Pour ces raisons, nous avons jugé important de développer une variante du *configuration model* prenant en compte les contraintes de simplicité et/ou de connexité : la génération d'un graphe simple connexe aléatoire avec une séquence de degrés fixée.

2.1.2 Conventions techniques et notations

Nous introduisons ici les termes techniques et les structures nécessaires dans le reste du chapitre.

Échanges d'arêtes

Nous utilisons les conventions suivantes :

- Les graphes seront non dirigés ; N désigne le nombre de sommets d'un graphe et M son nombre d'arêtes.
- Étant donné 4 sommets a, b, c, d tels que les arêtes $(a-b)$ et $(c-d)$ existent, l'*échange d'arêtes* $(a-b),(c-d) \rightarrow (a-d),(b-c)$ consiste à remplacer ces deux arêtes par les arêtes $(a-d)$ et $(b-c)$, comme représenté en Figure 2.3.
- Étant donné un échange d'arêtes $(a-b),(c-d) \rightarrow (a-d),(b-c)$, son *dual* est l'échange d'arêtes $(b-a),(c-d) \rightarrow (b-d),(a-c)$ (voir Figure 2.3).
- Étant donné un échange d'arêtes $(a-b),(c-d) \rightarrow (a-d),(b-c)$, son *inverse* est l'échange d'arêtes $(b-c),(d-a) \rightarrow (b-a),(c-d)$. Appliquer un échange d'arêtes puis son inverse laisse le graphe inchangé.

FIG. 2.3 – Un échange d'arêtes et son dual.

Complexité des opérations élémentaires sur les graphes

Nous supposons que les graphes sont gérés dans des structures permettant d'obtenir les complexités suivantes :

1. L'*existence* d'une arête entre tout couple (a, b) de sommets est déterminée en temps constant $O(1)$.
2. Pour tout quadruplet de sommets a, b, c, d, l'échange d'arêtes $(a - b), (c - d) \rightarrow (a - d), (b - c)$ se fait en temps constant $O(1)$.
3. Pour tout sommet v, l'obtention de la liste Vois(v) de ses $deg(v)$ voisins se fait en temps linéaire $O(deg(v))$.
4. Choisir une arête aléatoire du graphe, c'est-à-dire un couple (a, b) de sommets choisi *uniformément* parmi l'ensemble des couples de sommets liés, se fait en temps moyen $O(1)$.
5. L'espace mémoire requis par la structure permettant de gérer le graphe est linéaire en $N + M$.

Nous décrivons ici schématiquement une structure de données permettant d'obtenir ces complexités[1].

Les sommets sont identifiés par des entiers de 0 à $N - 1$, et leurs degrés sont stockés dans un tableau $Deg[\cdot]$. Nous supposons que l'architecture utilisée effectue les opérations arithmétiques usuelles sur les entiers en temps constant. Pour chaque sommet i, nous maintenons une table de hachage décrivant la liste d'adjacence L_i qui contient les $Deg[i]$ voisins de i, et stockons l'adresse et la taille de cette table dans des tableaux, respectivement en $Adr[i]$ et $Taille[i]$. Les propriétés standard des tables de hachage assurent que les conditions 1 et 2 ci-dessus sont satisfaites. Nous nous assurons également que pour une constante K fixée, toutes nos table de hachage sont K-linéaires : $\forall i, Deg[i] \leq Taille[i] \leq K \cdot Deg[i]$, de manière à ce que le point 3 cité ci-dessus s'effectue en scannant simplement l'ensemble de la table de hachage, et en enlevant les $O((K - 1) \cdot Deg[i])$ trous.

Le quatrième point est plus délicat. Pour choisir une arête aléatoire de manière uniforme parmi les M arêtes du graphe, nous procédons en deux étapes : d'abord, nous tirons un sommet v parmi l'ensemble des sommets de manière à ce que la probabilité de choisir un sommet soit directement proportionnelle à son degré, puis nous tirons un élément aléatoire w de Vois(v) de manière uniforme. Il est alors facile de voir que l'arête $(v - w)$ obtenue est bien choisie aléatoirement uniformément parmi les M arêtes du graphe. La première étape peut être fait en temps moyen constant si nos listes d'adjacence L_i sont juxtaposées dans une zone de mémoire contiguë : chaque sommet i étant représenté exactement $Deg[i]$ fois dans cette zone mémoire (puisqu'il appartient à $Deg[i]$ listes d'adjacence), il suffit de tirer au hasard un emplacement dans cette zone mémoire et de continuer tant qu'on est tombé sur un trou d'une table de hachage. La deuxième étape est similaire, mais cette fois on

[1] La structure que nous utilisons dans notre programme disponible sur le Web est légèrement différente, ayant été optimisée pour obtenir de meilleurs temps d'éxecution et des besoins plus réduits en mémoire. Pour plus d'informations, consulter le code source disponible à [134].

restreint le tirage à L_v. Cela se fait bien en temps moyen constant grâce à la K-linéarité des tables de hachages.

Notons que les structures clés décrites ici, à savoir les tableaux $Deg[\cdot]$, $Adr[\cdot]$ et $Taille[\cdot]$, ne sont modifiés par aucune des opérations ci-dessus, ne dépendent que de la séquence de degrés et peuvent donc être créées une fois pour toutes lors de la génération du graphe. En particulier, un échange d'arêtes ne change pas les degrés des sommets concernés, et ne modifie donc pas la taille des listes d'adjacence correspondantes. Notons également que l'espace requis est bien linéaire en la taille du graphe $O(N + M)$.

2.1.3 Une génération en trois étapes

De nombreuses techniques ont été proposées pour la génération aléatoire de graphes simples et connexes à séquences de degrés fixée. Nous nous concentrons ici sur l'algorithme basé sur des chaînes de Markov [40], désigné par une étude récente [75] comme étant le plus performant.

L'algorithme de génération se subdivise en trois phases :

1. **Réalisation** de la séquence de degrés : génération d'un graphe simple ayant les degrés voulus.
2. **Connexion** du graphe à l'aide d'échanges d'arêtes, donc sans modifier les degrés de ses sommets, et en gardant le caractère simple du graphe.
3. **Brassage** du graphe à l'aide d'échanges d'arêtes aléatoires conservant la simplicité et la connexité pour obtenir un graphe aléatoire.

Nous traitons ici des deux premières étapes, et nous intéressons à la troisième dans le reste du chapitre.

Réalisation de la séquence de degrés

L'algorithme de Havel-Hakimi [51, 50] s'occupe de la première phase en temps et espace linéaire, et ressemble à la procédure de couplage aléatoire vue en Section 2.1.1. Chaque sommet i se voit d'abord assigner $deg(i)$ "demi-arêtes". Ensuite, l'algorithme choisit un sommet quelconque ayant au moins une demi-arête, et lie toutes ses demi-arêtes aux demi arêtes d'autres sommets, en choisissant ces derniers parmi les sommets ayant le plus de demi-arêtes encore disponibles. La liaison de deux demi-arêtes crée une arête, et implique donc la destruction des deux demi-arêtes concernées : le processus s'arrêtera donc au bout d'au plus M fusions de demi-arêtes. Un résultat d'Erdös et Gallai [32] montre que cet algorithme termine avec succès (avec aucune demi-arête restante) si et seulement si la séquence de degrés est réalisable, c'est-à-dire s'il existe un graphe simple avec cette séquence de degrés. Nous supposons implicitement que c'est le cas dans ce chapitre : si la séquence de degrés voulue n'est pas réalisable, le programme échouera à la première étape.

Pour obtenir une complexité linéaire, nous mettons simplement en place $O(d_{max})$ liste chaînées $L_1, \cdots, L_{d_{max}}$, où d_{max} est le degré maximal dans la séquence donnée. On aura vérifié au préalable que $d_{max} < N$ car sinon la séquence n'est pas réalisable. Chaque L_i

contiendra, au cours de l'algorithme, les sommets dont le nombre de demi-arêtes est égal à i (les sommets n'ayant aucune demi-arête sont considérés comme traités et n'ont plus besoin d'être considérés). On maintient également, au cours de l'algorithme, l'indice maximal K tel que L_K est non vide. L'initialisation se fait en temps linéaire : au début chaque L_i contient tous les sommets de futur degré i, et $K = d_{max}$. Le branchement va se faire en traitant les sommets un par un.

Lorsque l'algorithme traite un sommet v, il doit sélectionner les sommets ayant le plus de demi-arêtes pour les brancher aux demi-arêtes de v : il suffit de parcourir la liste non vide d'indice maximal L_K et de brancher une demi-arête de chaque sommet parcouru, en **marquant** les sommets parcourus. Si les sommets de L_K n'ont pas suffit à compléter les demi-arêtes de v, on continue avec L_{K-1}, et ainsi de suite. On décrémente ensuite K si L_K avait été complètement marquée. Avant le parcours, on aura pris soin de retirer v de sa liste pour ne pas le brancher à lui-même. Une fois v complètement branché, il faut ensuite transférer chaque sommet marqué dans la liste d'indice inférieur – puisqu'on leur a retiré une demi-arête – sauf dans le cas des sommets marqués appartenant à L_1, qui sont simplement supprimés. Le traitement d'un sommet de degré v se fait donc en temps $O(deg(v))$, d'où une complexité totale linéaire $O(M)$.

Connexion du graphe

La deuxième phase consiste d'abord à vérifier que le nombre d'arêtes $M = \frac{1}{2}\sum_{i=1}^{N} deg(i)$ vérifie $M \geq N-1$ et qu'aucun sommet n'a un degré nul : $\forall i, deg(i) > 0$. Ces deux conditions sont évidemment nécessaires à la connexion du graphe, et l'algorithme suivant montrera qu'elles sont suffisantes. Nous utilisons ensuite des échanges d'arêtes pour connecter entre elles les différentes composantes connexes, jusqu'à ce qu'il n'existe plus qu'une seule composante connexe. Le principe est le suivant : étant données deux composantes connexes C_x et C_y telles que C_x ne soit pas un arbre, nous choisissons une arête $(a-b)$ *superflue*[2] de C_x, où superflue signifie qu'elle peut être enlevée sans déconnecter C_x, et nous choisissons une arête $(c-d)$ quelconque[3] de C_y. L'échange d'arêtes $(a-b),(c-d) \rightarrow (a-d),(b-c)$ connecte clairement les deux composantes C_x et C_y, et ne crée pas d'arêtes multiples puisque C_x et C_y n'étaient pas connectées. La Figure 2.4 schématise ce processus. Nous pouvons ainsi répéter la fusion de deux composantes connexes jusqu'à ce qu'il ne reste qu'une seule composante, ou bien $k \geq 2$ composantes arborescentes. Ce dernier cas est impossible, puisqu'il impliquerait que le nombre total d'arêtes M est égal à $N-k$, ce qui est exclu par l'hypothèse de départ $M \geq N-1$.

Pour obtenir une complexité linéaire, cet algorithme procède de manière incrémentale, en maintenant une seule composante C_0 "de base" possédant une liste $L = \{(a_0 - b_0), \cdots, (a_p - b_p)\}$ d'arêtes superflues, et une liste de composantes arborescentes $T = \{C_1, \cdots, C_k\}$. Initialement, les listes sont vides et C_0 n'existe pas.

[2] L'existence d'une telle arête est garantie par le fait que C_x n'est pas un arbre, voir Section 1.4.3.

[3] C_y contient au moins deux sommets et une arête puisque par hypothèse aucun sommet n'a un degré nul

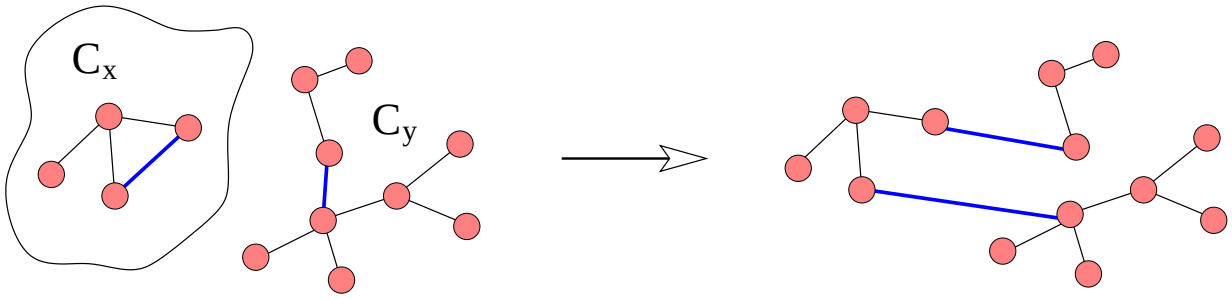

FIG. 2.4 – Fusion de deux composantes connexes, dont au moins une (ici C_x) n'est pas arborescente, à l'aide d'un échange d'arêtes.

L'algorithme lance un parcours en largeur d'un sommet non encore visité, et va donc explorer une nouvelle composante connexe C. Celle-ci va soit se révéler être arborescente et être ajoutée à T, soit posséder au moins une arête superflue, auquel cas, à la première arête superflue $(v - w)$ détectée pendant le parcours en largeur, deux cas de figures se présentent :

- Soit C_0 n'existe pas encore, on pose alors $C_0 = C$, ce qui se traduit en pratique par l'initialisation de L à $L \leftarrow \{(v - w)\}$.
- Soit C_0 existe déjà, on fusionne alors C avec C_0 avec l'échange d'arêtes $(a_0 - b_0), (v - w) \rightarrow (a_0 - w), (b_0 - v)$. Parmi les deux arêtes créées, on peut considérer l'une d'entre elles comme superflue[4], qu'on utilise pour remplacer $(a_0 - b_0)$ (qui n'existe plus) dans L.

Les arêtes superflues rencontrées subséquemment dans le parcours de C seront simplement ajoutées à L (puisque C a été fusionné avec C_0).

On poursuit cet algorithme jusqu'à ce que tous les sommets soient visités. À la fin, on aura l'ensemble L des arêtes superflues de C_0 et l'ensemble T des composantes arborescentes, qu'il suffit de connecter par échange d'arêtes comme décrit précédemment.

2.1.4 Un algorithme basé sur les chaînes de Markov

La troisième et dernière phase de l'algorithme de génération est constituée d'échanges aléatoires d'arêtes. On maintient le graphe simple et connexe pendant le processus. Le graphe initial G_0 est le graphe obtenu à la fin des deux premières phases, il est simple et connexe avec la bonne séquence de degrés. Étant donné le graphe G_t à l'étape t, on choisit deux arêtes aléatoirement parmi les M arêtes de G_t et on exécute l'échange d'arêtes associé, ce qui donne un autre graphe G' ayant les mêmes degrés. Si G' est simple et connexe, nous considérons l'échange d'arêtes comme *valide* et posons $G_{t+1} = G'$. Sinon, nous rejetons l'échange : $G_{t+1} = G_t$.

[4]La fusion de deux composantes ayant chacune une arête superflue engendre une composante ayant au moins une arête superflue : voir la Figure 2.4 en ajoutant une arête à C_y.

Cet algorithme est en fait une chaîne de Markov où l'**espace** est l'ensemble S_{CS} des graphes simples et connexes ayant les degrés désirés, l'**état initial** G_0 est le graphe obtenu à la fin de la deuxième phase, et les **transitions** $G_x \to G_y$ ont probabilité $1/M(M-1)$ s'il existe un échange d'arêtes transformant G_x en G_y, et 0 sinon. Ce dernier point vient des deux observations suivantes :

1. S'il existe au moins un échange d'arêtes transformant G_x en G_y, cet échange d'arêtes est *unique*.
2. Il y a exactement $M \cdot (M-1)$ échanges d'arêtes possibles (valides ou invalides), puisqu'il y a $M(M-1)/2$ paires d'arêtes, chacune correspondant à deux échanges possibles (voir Figure 2.3).

Notons que pour tout graphe $G_x \in S_{CS}$, la probabilité associée à la *transition fixe* $G_x \to G_x$ est donnée par le nombre d'échanges d'arêtes invalides sur G_x, divisé par $M(M-1)$. Pour montrer qu'il en existe au moins un, supposons d'abord que nous ne sommes pas dans le cas trivial $N \leq 2$, ou un seul graphe simple et connexe existe. Par connexité, G_x doit contenir au moins un sous-graphe de la forme a, b, c où les arêtes $(a-b)$ et $(b-c)$ existent. L'échange d'arêtes $(a-b),(b-c) \to (a-c),(b-b)$ est invalide, puisqu'il crée l'arête $(b-b)$. Par conséquent, pour tout $G_x \in S_{CS}$, la transition fixe $G_x \to G_x$ a une probabilité strictement positive au moins égale à $1/M(M-1)$.

Théorème 1. *Cette chaîne de Markov est irréductible [103], symétrique, et apériodique.*

Corollaire 2. *La chaîne de Markov converge vers la distribution uniforme sur son espace, c'est-à-dire tous les graphes possédant les propriétés voulues.*

Démonstration. Le corollaire vient des propriétés standard des chaînes de Markov.

- L'irréductibilité, autrement dit le fait que tout état $G_y \in S_{CS}$ soit atteignable depuis tout autre état $G_x \in S_{CS}$ par une suite finie de transitions, a été prouvé dans notre cas par Taylor [103].
- L'apériodicité est une conséquence, connue dans le cadre des chaînes de Markov, du fait que les transitions fixes $G_x \to G_x$ ont une probabilité strictement positive
- La symétrie vient de la réversibilité des échanges d'arêtes, déjà introduite en Section 2.1.2 : si l'échange $(a-b),(c-d) \to (a-d),(b-c)$ transforme G_x en G_y, alors l'échange inverse $(a-d),(c-b) \to (a-b),(d-c)$ transforme G_y en G_x. Par conséquent, pour tout couple d'états $(G_x, G_y) \in S_{CS} \times S_{CS}$, les transitions $G_x \to G_y$ et $G_y \to G_x$ ont même probabilité.

□

Ces résultats montrent que pour générer un graphe aléatoire il suffit de faire *suffisamment* de transitions. Cependant, à l'heure actuelle aucun résultat ne permet de connaître le nombre de transitions requis pour obtenir une convergence "satisfaisante". En termes plus précis, la convergence rapide (polynomiale) de cette chaîne de Markov n'est pas établie. Des travaux récents [59] permettent la démonstration d'une convergence au plus quadratique à l'avenir, ce qui reviendrait à dire que le nombre de transitions nécessaire dépend

du carré de la taille $N + M$ du graphe. Un autre article [36] montre même la convergence polynomiale de cette chaîne de Markov, mais sous réserve que la séquence de degrés vérifie certaines contraintes spécifiques. Notre contribution dans cette thèse étant axée sur une réduction du coût des transitions, ce qui n'intervient pas sur la durée du brassage en termes de nombre d'échanges d'arêtes, nous laissons une discussion sur la vitesse de convergence aux futurs travaux. Afin de pouvoir parler de la complexité du brassage, nous adoptons donc la convention ci-dessous. Notons que nous ne considérons pas le coût moyen par *transition*, mais par échange d'arêtes effectués : cela est dû aux modifications que nous apportons au processus de brassage, qui ne fera plus forcément intervenir les chaînes de Markov. L'expérience nous a également montré que le nombre d'échanges d'arêtes effectués est un bon quantificateur de l'état d'avancement du brassage.

Convention 1 (Coût)**.** *Le* **coût** *de nos algorithmes liés au brassage du graphe est défini comme le coût moyen par échanges d'arêtes* **validés**. *Dans la suite, nous utiliserons le terme* **coût unitaire** *pour rappeler notre situation particulière et éviter les confusions. En d'autres termes, si le brassage se termine après que x échanges d'arêtes aient été validés – les échanges annulés à cause de la clause obligeant le graphe à rester simple et connexe ne sont donc pas comptés – avec un coût total C_{total}, le* **coût unitaire** *du brassage sera de C_{total}/x.*

Nous restons cependant dépendants de la durée globale du brassage car nous nous intéressons *in fine* à une mise en pratique de ces algorithmes. Pour pouvoir parler de complexités globales, nous suivrons donc la convention suivante, non démontrée formellement mais vérifiée par des études empiriques [40, 75] et par notre propre expérience.

Convention 2. *[40, 75] La chaîne de Markov converge après $O(M)$ échanges d'arêtes.*

Ce résultat, qui n'est pas un théorème mais une observation, a été confirmé dans des simulations intensives où, partant de graphes initialement biaisés à l'extrême, $O(M)$ échanges d'arêtes suffisaient toujours à rendre le graphe "parfaitement" aléatoire en apparence. Plus précisément, les distributions d'un large panel de métriques non triviales – telles que le diamètre et le flot, entre autres – sur l'ensemble des graphes obtenus par ces brassages n'étaient pas différentes des distributions obtenues sur des graphes *réellement* aléatoires. Nous avons nous-mêmes essayés, en connaissance de cause, de trouver une métrique capable de falsifier cette assertion, sans succès.

Pour finir, notons que $\Omega(M)$ est une borne inférieure triviale pour le nombre d'échanges d'arêtes nécessaire au brassage, puisque chaque échange n'implique que 2 arêtes parmi un total de M.

2.1.5 Équivalence entre échanges d'arêtes et transitions

Pendant le brassage, une transition n'implique pas la validation d'un échange d'arête, puisque ceux-ci peuvent être annulés. Nous venons de mettre en place des conventions basées sur le calcul du coût unitaire de nos algorithmes. Dans l'algorithme tel qu'il est

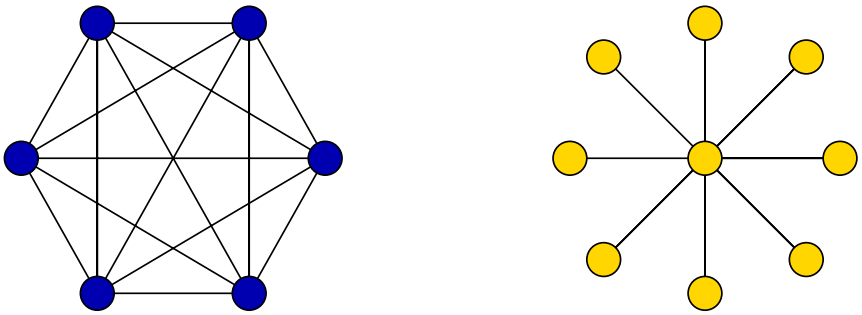

FIG. 2.5 – Exemples de graphes "pathologiques" où aucun échange d'arêtes n'est valide : sur le graphe de gauche (clique), tout échange d'arêtes crée des arêtes multiples. Sur le graphe de droite (étoile), tout échange d'arêtes laisse le graphe inchangé ou crée une boucle et déconnecte le graphe.

décrit, il est facile de calculer le coût d'une *transition*, qui consiste en une suite d'opérations bien définie, mais cela ne nous donne pas le coût unitaire. Nous fournissons ici une preuve concise de la dépendance linéaire entre le nombre de transitions effectuées et le nombre d'échanges d'arêtes validés, ce qui revient en fait à montrer qu'il existe une proportion non nulle (minorée par une constante strictement positive) d'échanges d'arêtes valides parmi tous les échanges possibles. Cela permettra d'obtenir aisément des bornes de complexité en terme de coût unitaire, pour l'algorithme de brassage décrit ici.

Considérons un graphe simple et connexe G, et deux arêtes $(a-b),(c-d) \in G$, comme montré en Figure 2.3, telles que a, b, c, d soient distincts deux a deux. Soit G^* le graphe obtenu par suppression de ces deux arêtes. Comme G était connexe, il est facile de voir que dans G^* au moins l'un des deux sommets a et b est toujours connecté (par un chemin dans le graphe) à c ou d. Supposons à présent que l'échange d'arêtes $(a-b),(c-d) \rightarrow (a-d),(b-c)$ déconnecte G. Il est alors clair que dans G^*, a n'est pas connecté à c et que b n'est pas connecté à d. Par conséquent, soit a et d sont connectés, soit b et c sont connectés (dans G^*), et donc l'échange d'arêtes dual $(b-a),(c-d) \rightarrow (b-d),(a-c)$ ne déconnecte pas G. De plus, il ne crée pas d'arêtes multiples, puisque, comme nous venons de le rappeler, a n'était pas connecté à c, ni b à d. Par conséquent l'existence de X échanges d'arêtes ne créant pas de boucles – ce qui revient a dire que les 4 sommets considérés sont distincts – mais déconnectant G implique par ailleurs l'existence d'au moins X échanges d'arêtes valides (gardant G simple et connexe). Le taux d'échanges d'arêtes invalidés par la déconnexion du graphe ne peut donc être supérieur à $1/2$.

Malheureusement on ne peut pas en dire autant pour les échanges d'arêtes invalidés par création d'arêtes multiples ou de boucles : l'existence de séquences de degrés pathologiques amenant à des graphes extrêmes tels les étoiles ou les cliques montrés en Figure 2.5 rendent impossible la minoration du taux d'échanges d'arêtes invalides, puisque certain graphes ne permettent aucun échange d'arêtes valide. Le cas des échanges d'arêtes mettant en jeu des quadruplets de sommets a, b, c, d non deux à deux distincts n'a pas été clairement

écarté : ces échanges sont en fait soit invalides (il créent des boucles), soit laissent le graphe inchangé. On pourrait choisir de les ignorer ou non lors du comptage des échanges validés. Pour éliminer ces cas extrêmes et ce type de considérations, et ainsi s'assurer du bon comportement de nos algorithmes, nous sommes forcés d'introduire des hypothèses supplémentaires sur nos graphes. L'hypothèse choisie ci-dessous n'écarte justement que ce type de cas extrême, et reste totalement valide dans les cas rencontrés en pratique. Nous utilisons ensuite le théorème suivant, qui ne fait *aucune* hypothèse sur le graphe :

Théorème 3. *Pour tout graphe simple et connexe, soit ρ la proportion des couples de sommets ayant une distance strictement supérieure à 2, alors la probabilité pour qu'un échange d'arêtes aléatoire soit valide est minorée par $\frac{\rho}{2z(z+1)}$, où z est le degré moyen.*

Démonstration. Si $\rho = 0$ le résultat est trivial. Si $\rho > 0$, considérons un couple (v, w) de sommets à distance $d(v, w) \geq 3$. Comme le graphe est connecté, il existe un chemin $(v, v_1, \cdots, v_{l-1}, w)$ de longueur $l \geq 3$ reliant v à w. L'échange d'arêtes $(v-v_1), (w-v_{l-1}) \rightarrow (v - v_{l-1}), (v_1 - w)$ est valide : il ne déconnecte pas le graphe, et les arêtes qu'il crée ne sont pas des boucles (les sommets considérés sont distincts) et ne peuvent préexister car sinon on aurait $d(v, w) \leq 2$.

Les $\rho N(N-1)$ couples de sommets à distance supérieure ou égale à 3 définissent donc au moins $\frac{\rho \cdot N(N-1)}{8}$ échanges d'arêtes valides, puisqu'un échange d'arêtes correspond au plus à 8 couples de sommets. Un échange d'arêtes aléatoire est donc valide avec probabilité supérieure ou égale à $\frac{\rho \cdot N(N-1)}{8M(M-1)}$. Le fait que $M = \frac{N \cdot z}{2}$ termine la preuve. □

Dans les topologies rencontrées en pratique, $\rho > 0$ car les graphes vérifiant $\rho = 0$ sont très particuliers, et sont en fait des hybrides entre les deux cas extrêmes montrés en Figure 2.5. Pour une distribution de degrés donnée, la valeur ρ tend en fait à grandir avec la taille du graphe, puisque les sommets ont tendance à être de plus en plus loin les uns des autres. L'hypothèse $\rho > 0$, même si elle est peu habituelle, devrait convaincre l'utilisateur familier des réseaux réels, ne serait-ce qu'en remarquant qu'elle découle de l'hypothèse plus forte $\overline{d} > 2$, elle-même vraie pour la (quasi ?) totalité des réseaux réels. La donnée d'une séquence de degrés non pathologique permet presque toujours de montrer directement cette affirmation sans que cela dépende du graphe. Cela nous permettra dans la suite d'affirmer que pour une séquence de degrés donnée – en supposant implicitement que celle-ci est non pathologique – la proportion d'échanges d'arêtes valides est minorée par une constante strictement positive, indépendante du graphe en particulier. La conséquence qui nous intéresse est que le nombre d'échanges d'arêtes validés est proportionnel au nombre de transitions.

2.1.6 Complexité

Comme nous l'avons décrit dans la Section 2.1.3, les deux premières phases de la génération se font en temps et espace linéaire, et la troisième phase nécessite au moins $\Omega(M)$ échanges d'arêtes aléatoires validés. La complexité asymptotique de l'ensemble du processus dépend donc uniquement de la complexité du brassage, qui dépend elle-même uniquement

– en appliquant les conventions vues en Section 2.1.4 – du *coût unitaire* C du brassage. Dans cette section, nous nous focalisons donc sur cette quantité.

L'approche naïve pour effectuer le brassage est d'effectuer les transitions de la chaîne de Markov une par une. Pour chaque transition :

- On effectue un échange d'arêtes aléatoire
- On vérifie que le graphe est toujours simple en s'assurant que les arêtes créées par l'échange d'arêtes n'existaient pas déjà et que les 4 sommets concernés par l'échange d'arêtes étaient distincts.
- On vérifie que le graphe est toujours connexe en lançant un parcours en largeur depuis un sommet du graphe.
- Si l'un des deux tests précédents a échoué, on annule l'échange d'arêtes.

Rappelons que d'après les résultats de la Section 2.1.5, le coût unitaire du brassage est équivalent au coût d'une transition. En utilisant la structure de graphe décrite en Section 2.1.2, les coûts d'un échange d'arêtes, du test de simplicité et du test de connexité sont respectivement $C_{échange} = O(1)$, $C_{simp} = O(1)$ et $C_{conn} = O(M)$, ce qui donne un coût unitaire linéaire :

$$C_{naive} = O(M)$$

On peut améliorer considérablement cette complexité en utilisant les structures de graphes décrites dans [53, 54, 105] qui maintiennent des informations sur la connexité de graphes dynamiques. Chaque test de connexité peut alors être effectué en temps sous logarithmique $O(\log N / \log\log\log N)$ et chaque test de simplicité en temps $O(1)$, alors que les échanges d'arêtes ont un coût $O(\log N(\log\log N)^3)$. Le coût unitaire du brassage devient donc :

$$C_{dynamic} = O\left(\log N(\log\log N)^3\right) \tag{2.2}$$

Notons cependant que ces structures sont complexes et impliquent des constantes plutôt grandes dans les complexités asymptotiques citées. L'algorithme naïf, malgré le fait qu'il implique un coût unitaire disproportionnellement élevé, reste le plus utilisé en pratique grâce à sa simplicité. Notre contribution sera de montrer comment améliorer l'algorithme naïf jusqu'à surpasser les performances obtenues par l'approche à base de graphes dynamiques, et ce en conservant une grande simplicité.

2.1.7 Accélération du brassage et heuristique (+1)(/2)

Gkantsidis et al. ont proposé un moyen simple d'accélérer le processus de brassage, dans le cas de la méthode naïve [40] : au lieu de lancer un test de connexité à chaque transition, ils n'en font que tous les T échanges d'arêtes, pour un entier T que nous appellerons la *période* d'accélération. Si le test de connexité échoue, les T derniers échanges d'arêtes auront besoin d'être annulés (plus précisément, ceux qui n'ont pas déjà été annulés à cause du test de simplicité doivent être annulés). En cas d'échec, le brassage est donc considérablement ralenti, mais en cas de réussite on aura économisé $T-1$ tests de connexité. Le processus de brassage ne peut alors plus être considéré comme une simple chaîne de Markov. Cependant, il a été prouvé [101, 40] que le Corollaire 2 reste valide, et que ce processus converge toujours

vers la distribution uniforme sur l'ensemble des graphes acceptables, même s'il est en fait composé d'une concaténation de chaînes de Markov [40] et que le graphe peut – en théorie – se déconnecter et se reconnecter sans que cela soit détecté.

Le coût unitaire des tests de connexités est réduit d'un facteur T, mais en même temps les échanges d'arêtes ont plus de chance d'être annulés : après T échanges d'arêtes à la suite, le graphe a plus de chances de se déconnecter qu'après un seul échange. Il est donc crucial ici de bien différencier les échanges d'arêtes validés *a posteriori* par un test de connexité de ceux qui seront annulés à cause des tests de connexité. L'équivalence entre échanges d'arêtes validés et transitions telle qu'on l'a vue en Section 2.1.5 n'est plus valide telle quelle : on ne connaît pas la dépendance entre le nombre de tentatives d'échanges d'arêtes et le nombre de réussites. Cela donne une autre raison de suivre la Convention 1 pour parler du coût du brassage : le *coût unitaire* correspond au coût *par échange d'arêtes validés* a posteriori, et le coût unitaire d'une opération correspond au coût réel par rapport à l'avancement du processus de brassage.

Pour plus de clarté dans la suite, nous décomposons le brassage en étapes, l'étape i étant constitué de T_i tentatives d'échanges d'arêtes avec tests de simplicité, suivies d'un test de connexité et de l'annulation de tous les échanges d'arêtes si celui-ci échoue. Le graphe obtenu à la fin de l'étape i sera noté G_i. Le test de simplicité étant une simple formalité en termes de complexité et grâce à la convention et aux résultats vus en Section 2.1.5, nous l'oublierons par la suite, considérant implicitement qu'un échange d'arêtes désigne en fait une tentative d'échange d'arêtes immédiatement annulée si le test de simplicité échoue. Pour quantifier le taux d'échange d'arêtes validés *a posteriori* par les tests de connexités, et ainsi pouvoir évaluer le coût unitaire du brassage avec l'algorithme de Gkantsidis et al., nous introduisons la grandeur suivante :

Définition 1 (Taux de réussite)**.** *Le taux de réussite $r_i = r(T_i)$ à une étape i donnée est la probabilité, connaissant G_{i-1}, que le graphe G_i obtenu après les T_i échanges d'arêtes soit toujours connexe.*

Évaluons a présent le coût unitaire du brassage : l'accélération divise le coût des tests de connexités par T sans augmenter le coût des échanges d'arêtes et des tests de simplicité, mais au final le coût unitaire est multiplié par $r(T)$ à cause des retours en arrière dus aux échecs des tests de connexité. Nous obtenons donc

$$C_{accel}(T) = \left(\frac{C_{attempt} + \frac{C_{conn}}{T}}{r(T)} \right) = \left(a \, \frac{b + \frac{M}{T}}{r(T)} \right) \tag{2.3}$$

où $C_{attempt}$ représente le coût moyen d'un échange d'arêtes avec un test de simplicité et l'annulation éventuelle de l'échange : $C_{simp} + C_{échange} \leq C_{attempt} \leq C_{simp} + 2C_{échange}$, et où $a = \frac{C_{conn}}{M}$ et $b = M\frac{C_{attempt}}{C_{conn}}$ sont des facteurs constants dépendant de la machine hôte, du code utilisé et des caractéristiques techniques de la structure de graphe choisie – rappelons que $C_{conn} = \Theta(M)$ et que $C_{attempt} = \Theta(1)$.

Pour $T = 1$, nous retrouvons le coût unitaire de l'algorithme naïf vu dans la section précédente :

$$C_{accel}(1) = O\left(a(b+M)\cdot r(T)^{-1}\right) = O\left(a\frac{b+M}{1-p}\right) = O\left(M\right) = C_{naive},$$

où p est la probabilité qu'un échange d'arêtes aléatoire déconnecte le graphe, qui est inférieure à $1/2$ d'après les résultats vus en Section 2.1.5.

Trouver une bonne valeur de T n'est pas facile, puisque le comportement de $r(T)$ est inconnu. Intuitivement, si T est trop grand le graphe sera déconnecté trop souvent et $r(T)$ sera trop petit, alors que si T est trop petit le taux de réussite $r(T)$ sera grand mais le facteur d'accélération sera réduit. Pour fixer une valeur de T "acceptable", Gkantsidis et al. ont proposé l'heuristique suivante (voir Figure 2.6), que nous désignerons par *heuristique (+1)(/2)*.

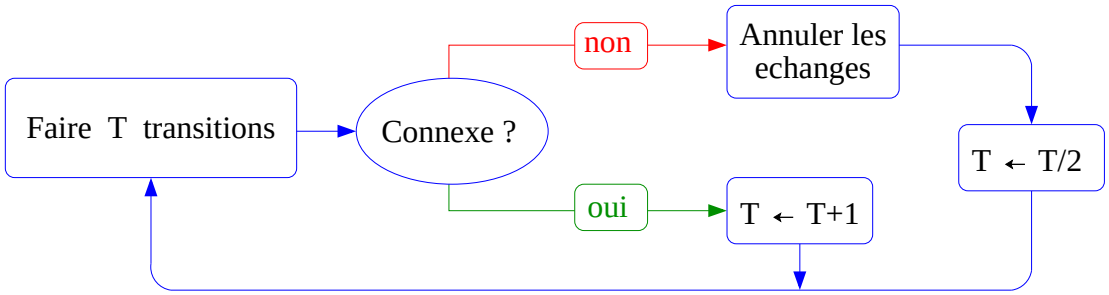

FIG. 2.6 – Heuristique (+1)(/2) [40]

Heuristique 1 (Heuristique (+1)(/2)). *À la fin de l'étape i :*
`SI` *le graphe obtenu à partir de G_{i-1} n'est pas connexe après les T_i échanges d'arêtes de l'étape i*
`ALORS` $T_{i+1} \leftarrow T_i/2$
`SINON` $T_{i+1} \leftarrow T_i + 1$

Ils s'attendent ainsi à voir T s'ajuster automatiquement vers un compromis entre un grand facteur d'accélération et un bon taux de réussite $r(T)$.

2.2 Étude de l'heuristique (+1)(/2)

Le problème auquel nous nous intéressons ici est l'estimation de l'efficacité de l'heuristique (+1)(/2). Nous commençons par introduire des notations et des outils permettant d'évaluer la valeur optimale de la période T, puis nous analysons le comportement de l'heuristique (+1)(/2) en comparant la période obtenue avec la période optimale, et enfin nous proposons une amélioration nous permettant d'atteindre cet optimum. Nous fournissons également une évaluation empirique des performances obtenues.

2.2.1 La période optimale

Commençons par définir la *probabilité de déconnexion* d'un graphe :

Définition 2 (Probabilité de déconnexion)**.** *Étant donné un graphe connexe G, la* **probabilité de déconnexion** *p est la probabilité que G se déconnecte après un échange d'arêtes aléatoire.*

Par la suite, nous allons admettre que les deux hypothèses suivantes sont vérifiées :

Hypothèse 1. *Pendant chaque étape i, la probabilité de déconnexion p reste constante pendant les T_i échanges d'arêtes effectués si le graphe reste connexe au cours de ces échanges. Cette probabilité est alors notée p_i.*

Hypothèse 2. *La probabilité qu'un graphe ayant été déconnecté se fasse reconnecter par un échange d'arêtes aléatoire est nulle.*

Ces deux hypothèses sont fausses en général, mais restent des approximations tout à fait raisonnables dans notre contexte et leur validité sera confirmée par la suite. La première hypothèse est discutée plus en profondeur dans [108], et la deuxième correspond à un pire cas destiné à faciliter les calculs mais restant empiriquement très proche de la réalité[5].

En admettant ces hypothèses, le taux de réussite r_i, qui est la probabilité pour que G_{i-1} reste connexe après T_i échanges d'arêtes, est donné par :

$$r_i = (1 - p_i)^{T_i} \tag{2.4}$$

Cette formule va nous permettre de donner une *approximation* (puisqu'on se base sur des hypothèses qui sont des vérités approchées) de la période optimale $T_{opt}(p)$.

Commençons par éliminer la cas trivial où p est de l'ordre de $\frac{1}{M}$ où plus petite : on peut alors poser $T = M$ et toujours obtenir un bon taux de réussite $r(T) = (1-p)^M = \Omega(1)$. L'Équation 2.3 nous donne alors $C_{accel} = O(1)$, qui est optimal a une constante près : le coût du brassage est alors linéaire par rapport au nombre d'échanges d'arêtes validés. Par la suite nous éliminons d'office ce cas particulier et supposons par conséquent que $p \gg \frac{1}{M}$, ce qui permettra de simplifier les formules et les études de cas, sans modifier la portée des résultats obtenus.

Définissons la période optimale T_i^{opt} à l'étape i comme la période T_i minimisant $C_{accel}(T_i)$. En se basant sur l'Équation 2.3 nous obtenons :

$$T_i^{opt} \cdot \left(\frac{b \cdot T_i^{opt}}{M} + 1 \right) = -log(1 - p_i)^{-1}$$

[5] La probabilité de reconnexion d'un graphe peu dense, à degré moyen borné, et ayant un nombre non négligeable de sommets de degré 1 était de l'ordre de $1/M$ voire inférieure sur toutes les topologies que nous avons pu tester. Cela correspond notamment à toutes les topologies issues de réseaux réels dont nous disposons.

ce qui implique notamment $T_i^{opt} < -log(1-p_i)^{-1} < \frac{1}{p_i}$, et donc $T_i^{opt} \ll M$ et $\frac{bT_i^{opt}}{M} + 1 \approx 1$, grâce à l'élimination du cas particulier $p = O(M^{-1})$ traité ci-dessus. Nous arrivons finalement à :

$$T_i^{opt} \approx -log(1-p_i)^{-1} \tag{2.5}$$

Le taux de réussite $r(T)$ étant une fonction bijective de la période T, la condition d'optimalité peut également être vue comme une contrainte sur r_i au lieu de T_i, ce qui permet de définir le *taux de réussite optimal* à l'étape i, obtenu à partir des Équations 2.4 et 2.5 :

$$r_i^{opt} \approx e^{-1} \tag{2.6}$$

Cette approximation revient simplement à négliger le coût $C_{attempt}$ par rapport à $\frac{C_{conn}}{T}$, et reste bien fondée tant que $p \gg \frac{1}{M}$. Notons que si $p_i \ll 1$ nous obtenons la valeur approchée simplifiée $T_i^{opt} \sim \frac{1}{p_i}$. Nous montrons en Figure 2.7 la valeur théorique de $C_{accel}(T)$ comme fonction de T ou de $r(T)$, pour un graphe ayant une probabilité de déconnexion $p = 1\%$. On voit bien que le coût minimal est atteint pour les valeurs $T \approx 1/p$ et $r \approx e^{-1} \approx 0.37$.

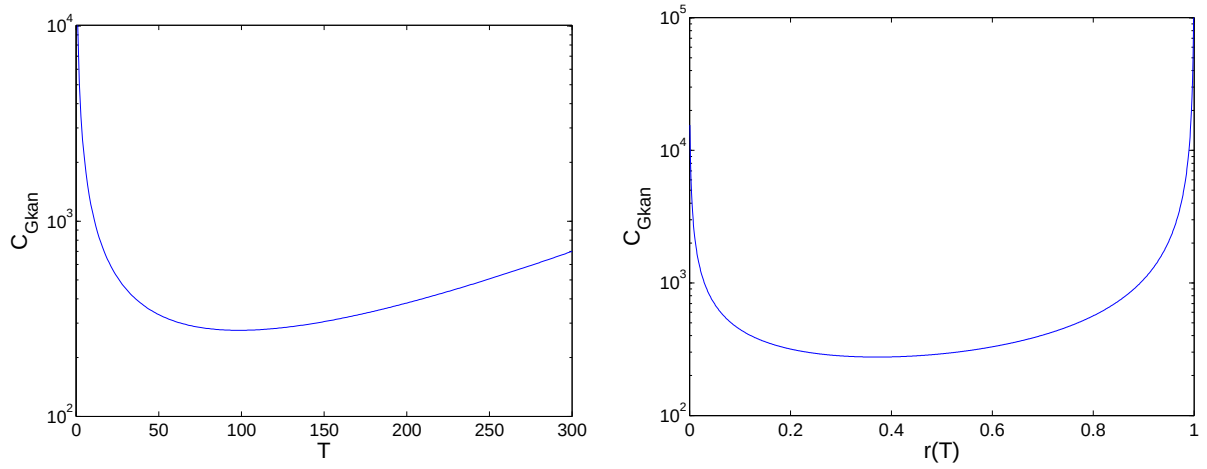

FIG. 2.7 – Valeur théorique du coût unitaire $C_{accel}(T)$ en fonction de la période T (à gauche) ou du taux de réussite $r(T)$ (à droite). Nous avons remplacé les constantes a et b de l'Équation 2.3 par 1 et posé $M = 10^4$ et $p = 10^{-2}$.

2.2.2 Analyse de l'heuristique (+1)(/2)

En se référant à la condition d'optimalité trouvée dans la section précédente, nous sommes désormais en mesure d'évaluer la performance de l'heuristique (+1)(/2). Notons $E_{(+1)(/2)}[\Delta T_i]$ l'espérance de $\Delta T_i = T_{i+1} - T_i$. Les règles d'évolution de la période dictées par l'heuristique impliquent :

$$E_{(+1)(/2)}[\Delta T_i] = r_i - (1-r_i) \cdot \frac{T_i}{2} = r_i \left(1 + \frac{T_i}{2}\right) - \frac{T_i}{2}$$

où r_i est toujours donné par $r_i = (1-p_i)^{T_i}$. Si nous considérons à présent $E_{(+1)(/2)}[\Delta T_i]$ comme une fonction de T_i, une simple étude des variations nous montre que cette fonction est strictement croissante, et donc bijective. Nous en profitons donc pour associer l'évènement $E_{(+1)(/2)}[\Delta T_i] = 0$ avec une certaine période, et introduisons la grandeur suivante :

Définition 3 (Période caractéristique)**.** *À une étape i de l'heuristique (+1)(/2), étant donnés le graphe G_i et sa probabilité de déconnexion p_i, La* **période caractéristique** $\widehat{T_{(+1)(/2)}}(p_i)$ *est l'entier positif vérifiant :*

$$T_i < \widehat{T_{(+1)(/2)}}(p_i) \iff E_{(+1)(/2)}[\Delta T_i] > 0$$

Intuitivement, cette période caractéristique correspond à la période d'*équilibre* de l'heuristique, celle pour laquelle la variation moyenne de la période est nulle. En supposant que p reste constante au cours du brassage afin que l'on puisse définir une période caractéristique globale $\widehat{T_{(+1)(/2)}}$, on peut voir que T_i est "attirée" par $\widehat{T_{(+1)(/2)}}$: si $T_i < \widehat{T_{(+1)(/2)}}$ alors T_i aura tendance à croître puisque $E[T_{i+1}] > T_i$, et inversement si $T_i > \widehat{T_{(+1)(/2)}}$. À cause de la nature stochastique de l'heuristique, mais aussi à cause de la variabilité de p_i pendant le brassage, une preuve formelle que la période T_i obtenue avec l'heuristique $(+1)(/2)$ reste *proche* de $\widehat{T_{(+1)(/2)}}(p_i)$ semble impossible à fournir. Nous avons cependant pu observer, pendant toutes nos expériences, la concordance de la théorie et des faits : la période T peut changer très vite avec cette heuristique (à cause de la division par deux) mais sa moyenne se stabilise autour d'une valeur correspondant effectivement à la période caractéristique $\widehat{T_{(+1)(/2)}}(< p >)$ où $< p >$ est la probabilité de déconnexion moyenne observée.

Afin d'estimer le degré d'optimalité de l'heuristique $(+1)(/2)$, nous pouvons à présent comparer $\widehat{T_{(+1)(/2)}}(p)$ avec la période optimale $T^{opt}(p)$, pour une valeur de p supposée constante. Nous avons déjà étudiée $T^{opt}(p)$ en Section 2.2.1, et pour $\widehat{T_{(+1)(/2)}}(p)$ nous obtenons la borne suivante :

Théorème 4. *La période caractéristique de l'heuristique (+1)(/2) est majorée par :*

$$\widehat{T_{(+1)(/2)}}(p) < \sqrt{\frac{2}{p}} \tag{2.7}$$

Démonstration. Rappelons que

$$E_{(+1)(/2)}[\Delta T] = r(T)\left(1 + \frac{T}{2}\right) - \frac{T}{2}$$

Comme $r(T) = (1-p)^T = e^{T\log(1-p)}$ et $\log(1-p) < -p$, on a :

$$E[\Delta T] < e^{-pT}(1 + \frac{T}{2}) - \frac{T}{2}$$

La convexité de la fonction exponentielle donne ensuite :

$$e^{-pT}(1+pT) < 1$$

Et enfin, l'équivalence

$$\frac{T/2+1}{1+pT} < \frac{T}{2} \iff T > \sqrt{\frac{2}{p}}$$

entraîne $T < \sqrt{\frac{2}{p}} \quad \Rightarrow \quad E[\Delta T] < 0$, ce qui termine la preuve. □

Quand p est petite la valeur $\sqrt{\frac{2}{p}}$ est beaucoup plus petite que l'optimale $T^{opt}(p) \sim \frac{1}{p}$. Cela montre que l'heuristique $(+1)(/2)$ est trop pessimiste : quand le graphe se déconnecte, T est divisée par 2, ce qui semble trop radical en comparaison avec la lente augmentation de T quand le graphe reste connexe. L'heuristique $(+1)(/2)$ a été conçue pour obtenir un taux de réussite très proche de 1, ce qui est trop loin de l'optimal $r^{opt} = e^{-1}$ (voir Figure 2.7).

2.2.3 Une dynamique optimale

Nous nous proposons de remplacer l'heuristique $(+1)(/2)$ sans changer le principe d'accélération : il s'agit donc simplement de redéfinir les règles d'évolution de T_i pendant le brassage.

Heuristique 2 (Heuristique q^+q^-)**.** *À la fin de l'étape i :*
`SI` *Le graphe obtenu à partir de G_{i-1} est déconnecté après les T_i échanges d'arêtes de l'étape i*
`ALORS` $T_{i+1} \leftarrow T_i \cdot (1 - q^-)$
`SINON` $T_{i+1} \leftarrow T_i \cdot (1 + q^+)$
`SI` $T_{i+1} > T_{limit}$
`ALORS` $T_{i+1} \leftarrow T_{limit}$

L'idée principale de cette heuristique est d'éviter une augmentation linéaire de T, trop lente, et d'autoriser plus de flexibilité grâce aux deux facteurs $1-q^-$ et $1+q^+$. Nous l'avons donc baptisée "heuristique q^+q^-".

La constante T_{limit} représente une valeur limite que T ne dépassera pas, correspondant en fait au cas trivial discuté plus haut où la probabilité de déconnexion est si faible que la complexité du brassage peut être optimale à une constante près en posant $T = \Theta(M)$. En pratique, nous utilisons $T_{limit} = 10\frac{C_{conn}}{C_{attempt}}$: en utilisant l'Équation 2.3 et le fait que $r(T)$ est décroissant avec T, on peut alors montrer que le coût unitaire est au plus 10% supérieur au coût unitaire minimal si $T_{opt} > T_{limit}$.

Étudions la dynamique de T loin du point où elle est influencée par la borne supérieure T_{limit}, en suivant la méthode déjà vue dans la section précédente pour l'étude de $T_{(+1)(/2)}$. L'espérance de ΔT_i devient :

$$E_{q^+q^-}[\Delta T_i] = q^+ r_i - q^-(1 - r_i) = r_i(q^+ + q^-) - q^- \tag{2.8}$$

Comme précédemment, notre nouvelle dynamique possède une période caractéristique $\widehat{T_{q^+q^-}}(p)$. Comme $r(T)$ et T sont bijectivement liés, nous pouvons également parler du *taux*

de réussite caractéristique $\widehat{r_{q^+q^-}}$, qui est le taux de réussite pour lequel $E_{q^+q^-}[\Delta T_i] = 0$. L'Équation 2.8 nous donne :

$$\widehat{r_{q^+q^-}} = \frac{1}{1 + \frac{q^+}{q^-}} \tag{2.9}$$

Reprenant la discussion développée sur l'analyse de l'heuristique $(+1)(/2)$, le taux de réussite caractéristique est un estimateur du taux de réussite obtenu par notre heuristique. À partir des conditions d'optimalité de l'Équation 2.6, et en supposant toujours qu'on est pas dans le cas trivial et donc que $p \gg 1/M$, nous obtenons le résultat suivant :

Théorème 5. *Si on suppose que p est constante durant le brassage et qu'elle vérifie $p \gg 1/M$, la période caractéristique de notre nouvelle dynamique est approximativement optimale* ssi *:*

$$\frac{q^+}{q^-} = e - 1 \ . \tag{2.10}$$

L'erreur relative de cette approximation, c'est-à-dire l'erreur relative entre la période caractéristique et la période optimale, est donnée par $\epsilon = \frac{b \cdot T_i^{opt}}{M}$.

Ce résultat montre l'optimalité de notre dynamique tant que $p \gg 1/M$. Notons que les cas où notre approximation devient illégitime, et notre optimalité non prouvée, correspondent au cas trivial où l'optimal est atteignable à une constante près en posant $T = M$, comme déjà discuté en Section 2.2.1. En pratique, l'heuristique se comporte bien pour des p de l'ordre de $1/M$, comme nous le montrons dans la section suivante.

Notons également que seul le *rapport* $\frac{q^+}{q^-}$ est contraint par la condition d'optimalité. L'*amplitude* $\sqrt{q^+q^-}$ peut être ajustée librement. Avec une grande amplitude, la période devient très instable, alors qu'avec une trop petite amplitude elle ne pourra s'adapter assez vite aux variations de p (qui induisent une variation de la période optimale $T_{opt}(p)$). En pratique, nous avons constaté que la qualité de notre dynamique est peu affectée par cette grandeur, pour peu qu'on la fixe dans un intervalle raisonnable, entre 10^-2 et 0.5 par exemple.

Nous montrons en Figure 2.8 un diagramme représentant, à simple fins démonstratives, la qualité de notre dynamique en fonction des grandeurs q^+ et q^-, évaluée empiriquement sur un ensemble de topologies représentatif du reste de nos expériences (voir la section suivante pour plus de détails). Sans rentrer dans les détails, le *rendement* utilisé dans ce diagramme est inversement proportionnel au coût unitaire, et est donc maximal quand le coût unitaire est minimal. Ce diagramme confirme – grossièrement – que le choix du rapport $\frac{q^+}{q^-} = e - 1$ semble optimal, et suggère d'utiliser une amplitude aux alentours de 0.1 (le pic de la courbe correspond précisément à ces deux valeurs).

2.2.4 Validation empirique

Pour évaluer la validité de nos résultats formels, basés sur les hypothèses 1 et 2 qui sont des approximations de la réalité, nous avons comparé les résultats obtenus avec les trois algorithmes suivants :

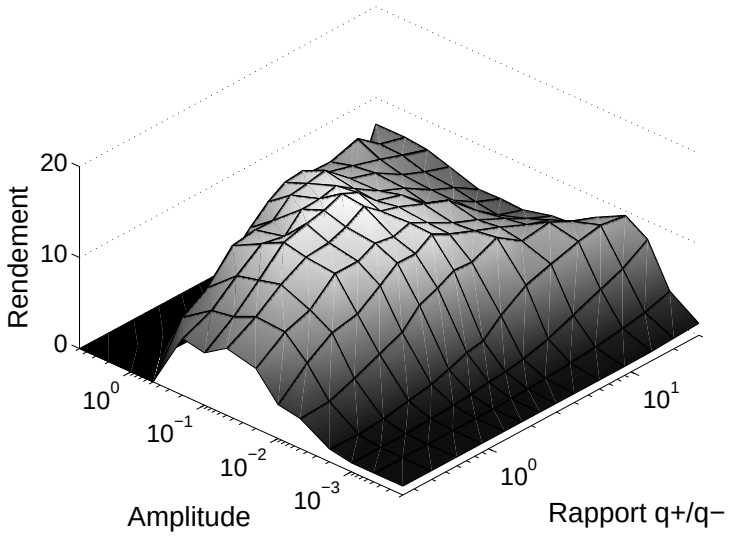

FIG. 2.8 – Diagramme expérimental récapitulant le *rendement*, valeur compilant les résultats de simulations sur plusieurs topologies et étant approximativement inversement proportionnelle au coût unitaire moyen, pour différentes valeurs de q^+ et q^-.

1. L'heuristique $(+1)(/2)$ (Figure 2.6)
2. Notre heuristique q^+q^-, en utilisant une amplitude $\sqrt{q^+q^-} = \frac{1}{10}$, comme le suggère l'étude vue en fin de Section 2.2.3.
3. La dynamique de T_i *optimale* : à chaque étape i, nous calculons la période T_i minimisant l'espérance du coût unitaire de l'étape i. Pour trouver cette valeur, nous simulons l'exécution de l'étape i un grand nombre de fois pour toutes les valeurs de T_i et gardons la valeur de T_i la plus performante. Le coût unitaire moyen dans le cadre de cette dynamique est noté C_{min}.

Le coût prohibitif du troisième algorithme empêche naturellement son utilisation en tant qu'heuristique d'évolution de T, mais il fournit le meilleur moyen que nous ayons trouvé pour simuler le comportement d'une heuristique *optimale*. L'optimalité est parfaite dans le sens où T_i ne pourrait être mieux ajustée. Notamment, cette optimalité ne dépend pas de la validité des approximations déjà utilisées.

Nous avons comparé le coût unitaire moyen obtenu par ces trois dynamiques (respectivement $C_{(+1)(/2)}$, $C_{q^+q^-}$ et C_{min}) pour la génération de graphes avec différentes séquences de degrés homogènes ou hétérogènes, ces dernières étant de la forme $P(X = k) = \beta(\mu + k)^{-\alpha}$ déjà vue en Section 1.4.2 pour imiter les réseaux réels. Pour le troisième algorithme, nous ignorons bien sûr le coût du calcul de T_i à chaque étape. Nous avons utilisé une large palette de paramètres, obtenant des résultats similaires que nous résumons pour quelques valeurs clés dans la table 2.1. La période moyenne $T_{(+1)(/2)}$ obtenue avec l'heuristique $(+1)(/2)$ se comporte bien comme la racine carrée de la période optimale, augmentant considérablement le coût unitaire pour les topologies ayant une faible probabilité de déconnexion.

Le coût unitaire obtenu avec notre nouvelle dynamique reste très proche du coût optimal (moins de 10% de différence), ce qui semble très encourageant et confirme la qualité de nos approximations.

Distributions de degrés binomiales

z	$C_{(+1)(/2)}$	$C_{q^+q^-}$	C_{min}
2.5	2606.12	1277.69	1272.13
3	1082.64	287.11	281.25
4	259.02	17.63	16.60
5	56.75	2.21	1.51

Distributions hétérogènes, $\alpha = 2.1$

z	$C_{(+1)(/2)}$	$C_{q^+q^-}$	C_{min}
2.1	16673.8	14787.7	14453.2
3	8587.6	5424.4	5271.7
6	4523.6	1143.4	1097.5
12	2576.2	242.3	239.4

TAB. 2.1 – Moyenne du coût unitaire des opérations effectuées pendant le brassage complet de graphe en cours de génération. Ce comparatif est résumé ici pour 8 topologies de graphes (8 séquences de degrés), 4 d'entre elles étant homogènes (à gauche) et correspondant au modèle aléatoire pur introduit en Section 1.5.1 et les autres étant fortement hétérogènes (a droite). Dans chaque table, on peut lire de gauche à droite le coût unitaire obtenu avec la dynamique $(+1)(/2)$, avec la nôtre, et avec la dynamique idéale. Nous nous sommes limités à des valeurs relativement faibles de N (10^4) à cause du coût démesuré du troisième algorithme.

Ces expériences confirment donc la qualité de notre heuristique et la validité de nos approximations. Nous pouvons donc exploiter les Équations 2.3, 2.5, 2.6 et 2.7 pour fournir une bonne estimation du coût unitaire de l'algorithme de brassage, pour l'heuristique $(+1)(/2)$ ou notre heuristique q^+q^- en fonction de la valeur moyenne $< p >$ de la probabilité de déconnexion pendant le brassage :

$$C_{(+1)(/2)} = O\left(1 + M \cdot \sqrt{< p >}\right) \tag{2.11}$$

$$C_{q^+q^-} = O\left(1 + M \cdot < p >\right). \tag{2.12}$$

D'autres comparatifs expérimentaux des deux heuristiques, fournissant notamment des temps de calcul sur machine, sont présentés plus loin dans ce chapitre dans la Table 2.2.

Le coût unitaire $C_{q^+q^-}$ obtenu avec notre heuristique q^+q^- peut s'avérer assez faible quand p est petite, malgré le fait que notre technique reste dépassée en termes de complexité asymptotique par les techniques de connexité dynamiques mentionnées auparavant (voir l'Équation 2.2). Pour la majorité des réseaux réels ayant un grand degré moyen – comme le World Wide Web, les réseaux de co-occurrence de mots ou de relations sociales – et donc une probabilité de déconnexion faible, notre approche peut tout de même s'avérer plus rapide en pratique, et conserve l'avantage d'être extrêmement simple à programmer. Il en va de même pour la génération de topologies homogènes comme celles présentées en Section 1.5, pour lesquelles les sommets de faible degrés sont rares et p peut devenir extrêmement faible (parfois inférieure à $1/M$). Cependant, notre méthode reste trop lente pour l'appliquer à de très grands graphes quand ceux-ci peuvent se déconnecter relativement facilement, comme c'est le cas notamment pour les mesures de la topologie de l'Internet dont nous disposons.

2.3 Prévenir la déconnexion à coût logarithmique

Nous allons à présent sortir du cadre restreint de l'algorithme de génération tel qu'il a été étudié dans la section précédente, et introduire un outil simple permettant de détecter les déconnexions du graphe à faible coût, ce qui réduit considérablement le recours aux tests de connexité, et donc le coût unitaire du brassage. Nous présentons d'abord l'idée directrice de ce test de déconnexion, puis nous adaptons l'algorithme naïf du brassage vu en Section 2.1.3. Nous analysons le coût unitaire de ce nouvel algorithme, et fournissons des coûts empiriques mesurés sur un large panel de topologies de graphes. Nous avançons enfin une conjecture, basée sur des intuitions fortes et des résultats expérimentaux solides, donnant une borne logarithmique au coût unitaire de notre algorithme, surpassant ainsi les meilleures complexités connues.

2.3.1 Idée directrice

Dans cette sous-section, nous ne fournissons pas d'argument formel ou d'étude détaillée, mais illustrons simplement l'idée générale d'un test de connexité partiel.

Observons tout d'abord que les graphes arborescents sont rares parmi les topologies réalistes, (sauf dans certains réseaux ou le caractère arborescent est justement requis). Les graphes ont en pratique un certain nombre d'arêtes superflues – que l'on peut enlever sans déconnecter le graphe – que l'on peut quantifier par $1 + \frac{z-2}{2}N$. Dans toute cette section, nous nous restreignons donc aux séquences de degrés vérifiant $\frac{M}{N} > 1 + \mu$ pour un réel μ strictement positif, ce qui nous permettra de s'assurer de la présence d'un nombre au moins linéaire d'arêtes superflues.

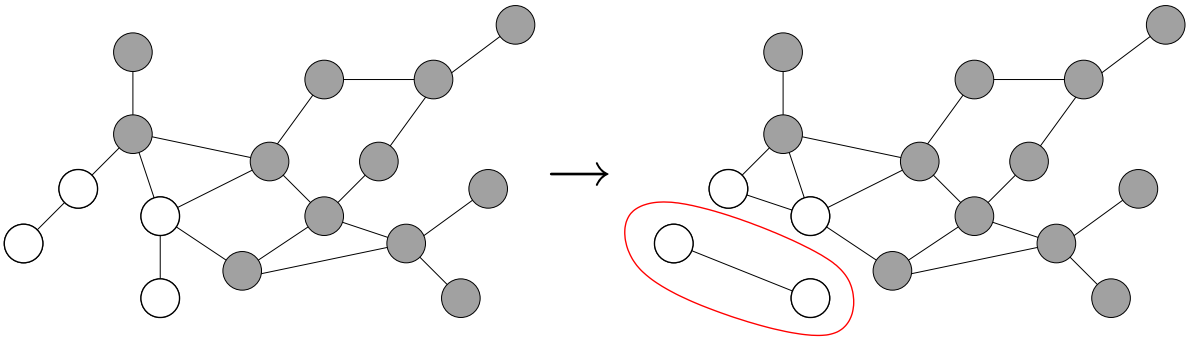

FIG. 2.9 – Déconnexion d'une paire de sommets isolée par un échange d'arêtes.

Nous avons également observé pendant le brassage que la majorité des déconnexions étaient causées par de *petites* composantes qui se séparaient du reste du graphe (principalement des paires de sommets isolés). Cela peut intuitivement être expliqué grâce à notre hypothèse $\frac{M}{N} > 1 + \mu$. Soit $C_K = v_1, \cdots, v_K$ un ensemble de K sommets connectés entre eux. Comme $\frac{M}{N} > 1 + \mu$, le degré moyen des sommets est supérieur à $2 + 2\mu$. Supposons à présent que le graphe est aléatoire dans le sens où une arête attachée à v peut être liée à

n'importe quel autre sommet du graphe (ce qui est une approximation de l'état de notre graphe aléatoire pendant la génération). Si nous isolons $K-1$ arêtes formant un arbre couvrant nos K sommets dans C_K, nous pouvons espérer avoir en moyenne $2\mu K$ arêtes superflues liées à nos K sommets et non impliquées dans l'arbre. Si nous supposons a présent que la composante connexe géante du graphe contient au moins $N/2$ sommets, il semble naturel de minorer la probabilité pour qu'une de ces arêtes soit connectée à un sommet de la composante géante par $1/2$. En suivant ce raisonnement, la probabilité que C_K ne soit pas connectée à la composante connexe géante semble minorée par $2^{-2\mu K}$. En d'autres termes, un ensemble de K sommets connectés a une probabilité K-exponentiellement faible d'être séparé de la composante connexe géante.

Cette intuition résulte d'une grossière approximation, cependant le caractère exponentiellement faible en K de la probabilité de séparation d'une composante de taille K semble valide. Nous ne prétendons pas fournir de preuve d'une telle assertion, mais espérons convaincre le lecteur du bien-fondé de cette intuition.

2.3.2 Un nouvel espace pour la chaîne de Markov

Suivant l'intuition décrite précédemment, nous introduisons l'opération suivante :

Définition 4. *Pour tout entier $K \geq 3$, un* **test d'isolement de portée** K *centré sur le sommet v* **réussit** *ssi la composante connexe contenant v possède au moins K sommets, et* **échoue** *sinon.*

Effectuer un test d'isolement de portée K se fait simplement en lançant un parcours en largeur depuis v et en l'arrêtant dès que l'on a visité K sommets. Le coût d'une telle opération est $O(K)$.

Rappelons que l'algorithme de brassage naïf vu en Section 2.1.3 est une chaîne de Markov sur l'espace S_{CS} des graphes simples et connexes ayant les degrés voulus, dont chaque transition peut se résumer à :

1. Effectuer un échange d'arêtes aléatoire
2. Tester si le graphe est toujours simple
3. Tester si le graphe est toujours connexe
4. Annuler l'échange si un des deux tests échoue

Nous proposons de garder le même schéma, mais en remplaçant le test de connexité complet par deux tests d'isolement de portée K centrés sur les sommets potentiellement séparés par l'échange d'arêtes, pour un entier K fixé. Par exemple, pour l'échange d'arêtes $(a-b),(c-d) \rightarrow (a-d),(b-c)$, on lancerait le test sur a ou d, au choix (puisqu'ils appartiennent a la même composante), puis sur b ou c, au choix. Il est aisé de montrer que le graphe ainsi obtenu ne possède *jamais* de composantes de taille strictement inférieure à K, même après la répétition de plusieurs transitions de ce type. En d'autres termes, notre algorithme modifié est – comme l'était l'algorithme naïf original – une chaîne de Markov. L'espace n'est plus S_{CS} mais est remplacé par l'ensemble S_K des graphes simples ayant les degrés

voulus et n'ayant pas de composante connexe de taille strictement inférieure à K. On a bien sûr $S_{CS} \subset S_K$.

Notons que le graphe obtenu à la fin n'est pas nécessairement connexe, puisque la seule garantie est qu'il appartienne à S_K. Pour pallier à ce problème, nous proposons une simple approche de rejet : si le graphe obtenu à la fin du brassage n'est pas connexe, nous recommençons à zéro avec une plus grande valeur pour K, en espérant que plus K est grand, plus la probabilité d'avoir un graphe connexe à la fin sera élevée. L'algorithme global est esquissé en Figure 2.10. Précisons que la chaîne de Markov sur S_K reste symétrique et apériodique, et que l'irréductibilité de sa restriction à S_{CS} implique clairement que tout état dans S_{CS} est atteignable puisque l'état initial G_0 est dans S_{CS}. Les graphes obtenus par notre algorithme restent donc uniformément distribués sur S_{CS}.

1. Créer un graphe G_0 simple et connexe réalisant la séquence de degrés
2. Créer une copie de sauvegarde de G_0
3. Initialiser K à $K = 2$
4. Initialiser $N_{transitions}$ au nombre de transitions voulu
5. `RÉPÉTER`
   . Restaurer le graphe à son état inital G_0
   . `FAIRE` $N_{transitions}$ fois :
      . Effectuer un échange d'arêtes aléatoire
      . Vérifier la simplicité
      . Effectuer un test d'isolement de portée K
      . `SI` un des deux tests échoue
        `ALORS` Annuler l'échange d'arêtes
   . Multiplier K par 2 : $K \leftarrow 2K$
   `TANT QUE` le graphe n'est pas connexe

FIG. 2.10 – Algorithme de génération final

Montrons que cet algorithme termine : si $K > \frac{N}{2}$ il est facile de montrer que tout graphe de S_K est connexe, ce qui implique que l'algorithme s'arrêtera au bout d'au plus $\lfloor \log_2 N \rfloor$ itérations de la boucle `RÉPETER` principale. Cela correspond bien sûr au cas le plus pessimiste. Notons que la suite $(S_K)_{K \in \mathbb{N}}$ est décroissante et converge vers S_{CS} :

$$S_2 \supset S_3 \supset \cdots \supset S_{\lfloor N/2 \rfloor + 1} = S_{CS}$$

Nous pouvons donc espérer que pour K assez grand, S_K devienne suffisamment proche de S_{CS} pour assurer la connexité du graphe final avec forte probabilité. Nous étudions en détail la valeur attendue de K (qui nous donnera le nombre d'itérations attendu de notre algorithme et donc sa complexité) dans la prochaine section.

2.3.3 Complexité

Nous tentons ici d'analyser la complexité de l'algorithme décrit dans la section précédente. Les opérations initiales (lignes 1,2,3,4) ont toutes un coût linéaire $\Theta(M)$ (voir Section 2.1.3). À chaque étape de la boucle `RÉPETER` principale (ligne 5), un test de connexité complet et une copie du graphe initial sont effectués : ces opérations ont également un coût linéaire $\Theta(M)$. Le coût total des échanges d'arêtes, des tests de simplicité et des tests d'isolement de portée K est $\Theta((K+1)N_{transitions})$.

Soit X le nombre total d'itérations de notre algorithme principal. La portée K des tests d'isolement suit une loi géométrique simple $2, 4, 8, \cdots, 2^X$. Le coût total est la somme des coûts de chaque exécution de la boucle `RÉPETER`, et s'écrit :

$$\Theta(M) + X \cdot \Theta(M) + \sum_{x=1}^{X} \Theta\left((2^x+1)N_{transitions}\right) \;=\; \Theta\left(M(X+1) + 2^X N_{transitions}\right)$$

Nous avons vu en Section 2.1.3 que le nombre de transitions $N_{transitions}$ à effectuer doit être au moins linéaire, et d'après notre discussion en Section 2.1.5 le nombre d'échanges d'arêtes validés sera proportionnel au nombre de transitions. Par conséquent, le coût unitaire de notre algorithme peut se résumer à

$$C_{final} = \Theta(2^X)$$

Étudions à présent le nombre moyen X d'itérations de notre algorithme. Rappelons que les espaces S_K diminuent avec K, et atteignent leur limite S_{CS} pour $K > N/2$.

Définition 5 (Taux de non connexion)**.** *Pour une séquence de degrés et une portée de test d'isolement K données, nous définissons le taux de non connexion $\epsilon(K)$ comme la proportion de graphes dans S_K qui ne sont pas connexes :*

$$\epsilon(K) = \frac{|S_K \setminus S_{CS}|}{|S_K|}$$

En d'autres termes, si $\epsilon(K)$ est petit, alors la plupart des graphes de S_K sont dans S_{CS}, et un graphe aléatoire de S_K aura une forte probabilité d'être connexe. Il est clair que $\epsilon(K)$ décroît avec K. Considérons à présent notre algorithme à la fin de la $x^{\text{ième}}$ itération, quand les $N_{transitions}$ transitions ont été effectuées avec des tests d'isolement de portée 2^x, engendrant un graphe $G \in S_{2^x}$. Si nous supposons que $N_{transitions}$ est assez grand pour que G puisse être considéré comme un élément aléatoire de S_{2^x}, la probabilité que G ne soit pas connexe est simplement $\epsilon(2^x)$.

À présent, définissons une portée de test d'isolement garantissant un taux de non connexion $\epsilon(K)$ suffisamment bas :

Définition 6. *Pour une séquence de degrés fixée, la portée caractéristique $\widehat{K}$ des tests d'isolement est le plus petit entier K tel que $\epsilon(K) < \frac{1}{3}$.*

La valeur $\frac{1}{3}$ utilisée ici est arbitraire, et tout réel de $]0, \frac{1}{2}[$ aurait pu faire l'affaire. Soit $P_{fin}(x)$ la probabilité que notre algorithme finisse après exactement x itérations de la boucle principale. Comme $\epsilon(K)$ décroît avec K et $1 - \epsilon(2^x)$ représente la probabilité que la $x^{\text{ième}}$ itération termine l'algorithme, nous obtenons :

$$\forall x \mid 2^x \geq \widehat{K}, \quad P_{fin}(x+1) < \frac{1}{3} P_{fin}(x)$$

Appelons $\widehat{x}$ le plus petit entier tel que $2^{\widehat{x}} \geq \widehat{K}$. Nous obtenons alors, pour le coût unitaire attendu $E[C_{final}] = \Theta(E[2^X])$ de notre algorithme, l'expression suivante :

$$E[C_{final}] = \Theta\left(\sum_{x=1}^{\infty} 2^x \cdot P_{fin}(x)\right) = \Theta\left(\sum_{x=\widehat{x}}^{\infty} 2^x \left(\frac{1}{3}\right)^{x-\widehat{x}}\right)$$

ce qui donne :

$$E[C_{final}] = \Theta\left(2^{\widehat{x}}\right) = \Theta(\widehat{K})$$

Le coût unitaire est donc proportionnel à $\widehat{K}$. Notons que l'algorithme choisi dans cette section est loin d'être optimal en pratique, et ne correspond pas à la version disponible dans le programme que nous fournissons sur le Web [134]. Nous l'avons cependant exposé ici dans un souci de simplicité et de rigueur, car les bornes de complexité sont beaucoup plus faciles à obtenir et à démontrer avec cette version. Reste que le coût unitaire $\Theta(\widehat{K})$ peut *a priori* se situer entre $\Theta(1)$ et $\Theta(N)$: il est crucial de pouvoir borner $\widehat{K}$ pour garantir l'efficacité de notre approche. Nous nous attaquons à ce problème dans la section suivante.

2.3.4 Portée caractéristique

Les bornes naturelles de $\widehat{K}$ sont données par la valeur minimale $K = 2$ et par l'identité $S_{\lfloor N/2 \rfloor + 1} = S_{CS}$, et sont donc $2 \leq \widehat{K} \leq 1 + \lfloor N/2 \rfloor$, ce qui reste bien trop vague pour estimer l'ordre de grandeur de $\widehat{K}$. Cependant, l'idée exposée en Section 2.3.1 s'accorde avec la totalité de nos expériences pour affirmer que $\widehat{K}$ est en pratique bien plus petit que $\frac{N}{2}$ pour les topologies non arborescentes (rappelons qu'en Section 2.3.1 nous nous sommes restreints au séquence de degrés telles que $\frac{M}{N} > 1 + \mu$, pour un $\mu > 0$). Ajoutons, et c'est un fait crucial, que nos mesures expérimentales indiquent que $\widehat{K}$ semble dépendre essentiellement de la séquence de degrés, mais qu'à distribution de degrés constante il croît lentement avec la taille du graphe.

Nous présentons en Figure 2.11 la variation de la portée caractéristique en fonction de la taille du graphe, pour différentes distributions de degrés. Le tracé de gauche montre les résultats obtenus avec des topologies homogènes : nous avons tiré des séquences de degrés réalisables[6] en utilisant des distribution binomiales avec des moyennes z variées. Un graphe

[6] Pour qu'une séquence de degré corresponde à un graphe simple et connexe, il faut notamment qu'aucun sommet n'ait un degré nul. En pratique, lorsqu'on tire des degrés selon une distribution, on élimine donc d'office degrés nuls, et on vérifie à la fin que la séquence est réalisable, quitte à recommencer jusqu'à obtenir un succès.

aléatoire tiré avec une distribution de degrés de ce type est extrêmement similaire au modèle classique de graphe aléatoire pur [33] déjà vu en Section 1.5.1. Sur le tracé de droite, nous montrons les résultats obtenus sur des topologies très hétérogènes, basées sur une loi de puissance d'exposant $\alpha = 2.5$, et modifiées pour s'ajuster à des degrés moyens voulus. Ces dernières distributions sont les **pires cas que nous ayons trouvé** : d'autres exposants – plus petits ou plus grands – et d'autres types de distributions résultaient systématiquement, à degré moyen égal, en des portées caractéristiques $\widehat{K}$ plus faibles et donc en des coûts unitaires moindres. Nous avons pourtant exploré de nombreuses séquences de degrés qui semblaient pathologiques, comme des séquence ne comportant presque que des sommets de degré 1 et quelques sommets de fort degré, des séquence avec le moins possible de sommets de degré supérieur à 2, et beaucoup d'autres.

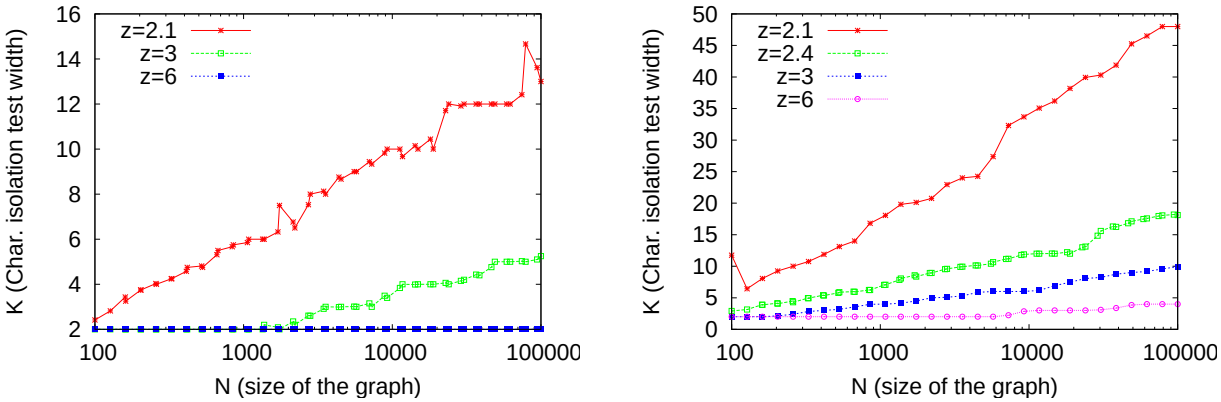

FIG. 2.11 – Variation de la portée caractéristique en fonction de la taille du graphe, pour une distribution de degrés fixée. À gauche : distribution de degrés binomiale. À droite, le pire cas que nous ayons trouvé à degré moyen fixé, à savoir des distributions de type loi de puissance d'exposant $\alpha = 2.5$ modifiées pour obtenir un degré moyen z voulu. Sur les deux tracés, on montre les résultats obtenus pour différentes valeurs du degré moyen z. L'échelle est semi-logarithmique.

La cohérence forte des résultats empiriques que nous avons obtenus (et dont nous n'avons montré que l'essentiel dans la Figure 2.11), combinée avec l'intuition décrite en Section 2.3.1 qu'un graphe n'ayant que des composantes de taille supérieure à K a une probabilité K-exponentiellement faible d'être déconnecté, nous amène à avancer la conjecture suivante :

Conjecture 1. *Il existe une fonction réelle η décroissante définie sur $]2, \infty[$ telle que, pour toute séquence de degrés de taille N et de moyenne z, la portée caractéristique $\widehat{K}$ soit majorée par :*

$$\widehat{K} \leq \eta(z) \log N$$

Selon cette conjecture, et si l'on se restreint – comme déjà discuté – aux topologies vérifiant $\frac{M}{N} > 1+\mu$, notre algorithme de génération a un coût unitaire logarithmique $O(\log N)$.

Cela dépasse même les performances des meilleurs algorithmes basés sur l'utilisation des structures de graphe dynamiques maintenant la connexité (voir l'Équation 2.2).

Distributions de degrés binomiales

z	$C_{(+1)(/2)}$	$C_{q^+q^-}$	C_{final}
2.5	2606.12	1277.69	16.49
3	1082.64	287.11	10.40
4	259.02	17.63	5.81
5	56.75	2.21	4.23

Distributions hétérogènes, $\alpha = 2.1$

z	$C_{(+1)(/2)}$	$C_{q^+q^-}$	C_{final}
2.1	16673.8	14787.7	87.71
3	8587.6	5424.4	19.84
6	4523.6	1143.4	14.19
12	2576.2	242.3	12.13

TAB. 2.2 – Résumé empirique du coût unitaire des différents algorithmes présentés jusqu'ici. Chaque donnée représente une moyenne sur 100 essais. La table de gauche correspond à une distribution de degrés homogène, celle de droite à une distribution hétérogène. Dans chaque table, on peut voir de gauche à droite : le coût unitaire de l'algorithme accéléré avec l'heuristique $(+1)(/2)$, avec notre heuristique q^+q^-, et le coût unitaire C_{final} de notre algorithme de génération final. Nous avons comptabilisé le coût des opérations de la manière suivante : un échange d'arêtes coûte 1, un test de connexité complet coûte M, et un test d'isolement de portée K coûte K. À cause du coût élevé $C_{(+1)(/2)}$, nous n'avons pas été plus loin que $N = 10^4$.

Les performances sont résumées dans les Tables 2.2 et 2.3 et montrent clairement l'avancée réalisée par notre algorithme. Précisons que la version de notre algorithme disponible en ligne permet d'utiliser les trois types de génération (le dernier algorithme est le comportement par défaut), et met en œuvre une version optimisée de l'algorithme final que nous ne détaillons pas ici. Cette version est notamment *toujours* plus performante que la version accélérée par notre heuristique vue en Section 2.2.3, et combine les idées de ces deux derniers algorithmes pour une efficacité optimale en pratique. Nous schématisons son fonctionnement sur la Figure 2.12 : on part d'un graphe G_0 obtenu par les réalisation de la séquence de degrés et la connexion du graphe (voir Section 2.1.3), qu'on injecte dans l'heuristique, et on continue tant qu'on a pas effectué le nombre voulu d'échanges d'arêtes.

À titre d'exemple, nous montrons un tableau comparatif des performances en termes de temps de calcul dans la Table 2.3. Le temps de calcul affiché est le temps correspondant au processus entier de génération. Chaque mesure a été moyennée sur une dizaine d'essais, sauf la mesure de $10600s$, trop longue, et les mesures précédées d'un $\approx$ pour lesquelles une estimation du temps de calcul a été faite, basée sur la complexité asymptotique et sur la progression du processus de brassage au bout d'une dizaine de minutes. Les temps sont montrés pour 4 méthodes : l'algorithme de génération naïf, puis la version accélérée avec les heuristiques $(+1)(/2)$ et q^+q^-, et enfin notre algorithme final tel que nous l'avons optimisé et programmé en [134].

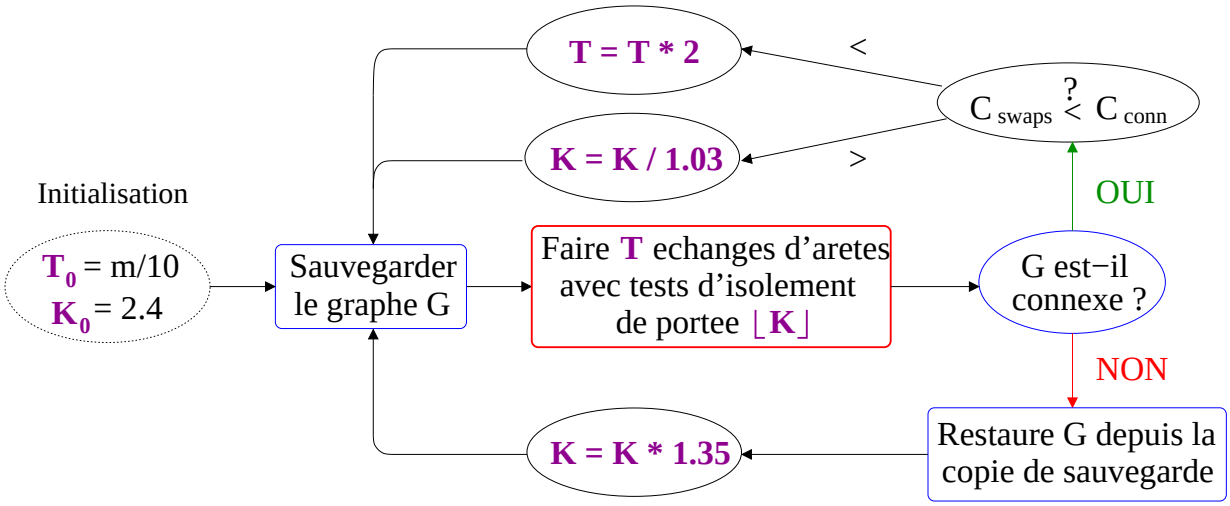

FIG. 2.12 – Heuristique utilisée pour l'algorithme de brassage optimisé utilisé dans la version publiée sur le Web.

M	Algo. naïf	Heur. $(+1)(/2)$	Heur. q^+q^-	Algo. final
10^3	0.51s	0.02s	0.02s	0.02s
10^4	26.9s	1.15s	0.47s	0.08s
10^5	3200s	142s	48s	1.1s
10^6	≈400000s	≈30000s	10600s	25.9s
10^7	≈40000000s	≈3000000s	≈1000000s	420s

TAB. 2.3 – Temps de calcul moyens pour la génération de graphes de différentes tailles et pour une distribution de degrés hétérogène fixée d'exposant $\alpha = 2.5$ et de moyenne $z = 6.7$. La machine utilisée est un DELL Inspiron 8600 avec un Intel Centrino 1.5GHz et 512MB de RAM.

2.4 Bilan

En nous intéressant au *configuration model*, la nécessité d'utiliser un modèle plus rigoureux afin de respecter strictement la séquence de degrés fixée nous a paru importante au vu des déviations constatée dans notre étude préliminaire. Par faute d'implantation publiquement disponible d'un programme permettant de générer des graphes selon un tel modèle, nous avons dû concevoir nous-mêmes un tel programme. Les algorithmes existants étant soit trop lents soit trop complexes à implanter, nous avons étudié puis amélioré un algorithme existant au point de découvrir une technique totalement neuve et très simple à décrire et programmer. Le projet final est disponible en open-source [134]. Nous résumons les contributions scientifiques de ce chapitre en les regroupant par thématiques.

Conception et implantation d'algorithmes

Nous avons d'abord implanté et décrit en détail les algorithmes linéaires permettant de générer un graphe simple et connexe (mais non aléatoire). Nous avons également implanté la méthode de brassage accélérée de Gkantsidis et al., puis conçu une heuristique visant à supplanter l'heuristique $(+1)(/2)$, trop peu performante. Les caractéristiques de notre heuristique sont un comportement quasi optimal et une grande adaptabilité à l'évolution des paramètres lors du brassage. Notre majeure contribution en termes d'algorithmique reste l'élaboration d'un concept novateur permettant de maintenir un certificat de connexité dans un graphe dynamique à coût amorti logarithmique, dans le cas d'un graphe *censé* rester connexe.

Analyse théorique et optimalité

Nous avons d'abord changé la convention utilisée dans la littérature antérieure à propos de la convergence des chaînes de Markov réalisant le brassage du graphe : nous quantifions l'avancement du brassage en nombres d'échanges d'arêtes *validés* et pas simplement en nombre d'échanges *tentés*.

Nous avons établi un système d'approximation et d'analyse formelle pour estimer la période optimale dans le processus d'accélération, et obtenu un critère d'optimalité (taux de réussite égal à e^{-1}). Nous avons ensuite analysé le comportement de l'heuristique $(+1)(/2)$ et montré sa non optimalité.

Une autre contribution importante est la mise en place d'une base formelle autour du test de connexité partiel introduit dans la dernière section. Nous avons pu ainsi proposer une conjecture propre dont une des implications serait la complexité logarithmique de notre approche finale.

Validation expérimentale

Notre travail dans ce chapitre, s'il se fonde sur de nombreuses considérations théoriques, garde une vocation appliquée. La validation empirique est pour nous cruciale. Nous avons utilisé des topologies de graphe variées, homogènes et hétérogènes. Nous avons vérifié et quantifié l'optimalité approchée de notre heuristique q^+q^- en la confrontant aux résultats que donnerait l'heuristique *optimale* au sens strict. Nous avons également mis nos algorithmes à l'épreuve grâce à des compteurs d'opérations et en mesurant les temps d'exécution sur machine. Nous avons cherché à contredire nos résultats en cherchant les contre-exemples les plus à même d'invalider nos résultats, ce qui nous a permis d'introduire les quelques restrictions vues dans ce chapitre (graphe non dégénéré et degré moyen strictement supérieur à 2).

En résumé, nous avons fait un effort de rigueur particulier dans nos expériences : si nos validations empiriques ne peuvent rivaliser avec des preuves formelles – que nous n'avons pas – des résultats approchés et de la conjecture que nous avançons, nous avons tout fait pour fournir des éléments convaincants.

Applications

L'article décrivant notre méthode de génération [110] et le site Web dédié à l'hébergement de notre application open-source [134] ont étés cités à ce jour dans un peu plus d'une dizaine d'articles scientifiques (hors articles écrits par des collaborateurs directs). La moitié d'entre eux ont utilisé notre application pour générer les graphes utilisés dans leurs travaux, les autres citent notre méthode de génération pour son intérêt algorithmique.

La plupart des utilisateurs ont choisi notre générateur de graphes grâce au hasard d'une recherche sur le Web, et certains d'entre eux n'auraient pas eu l'idée ou le besoin d'utiliser notre générateur, ce pour diverses raisons, comme par exemple est la petite taille des graphes générés, rendant possible la génération naïve en temps quadratique. Nous pensons cependant qu'une bonne partie de nos utilisateurs auraient – comme il nous semble être trop souvent le cas – naturellement choisi le *configuration model* avec les modifications discutées en Section 2.1.1 s'ils n'étaient pas tombés par hasard sur notre page Web. Nous espérons de cette manière contribuer à un effort de rigueur dans le domaine de la simulation sur graphes aléatoires. En effet, les implantations précédentes du *configuration model* rigoureux (c'est-à-dire tel que nous l'avons implanté dans ce chapitre), à cause de leur lenteur, incitaient fortement à utiliser la version très rapide mais biaisée décrite en Section 2.1.1.

Ajoutons enfin que notre méthode se généralise très aisément à des contraintes plus faibles : si on n'exige pas d'obtenir un graphe simple, il suffit d'enlever les tests de simplicité et tout le reste tient. De même si on enlève la contrainte de connexité, notre algorithme devenant alors trivialement linéaire en le nombre d'échanges d'arêtes. D'autres généralisations plus hardies sont envisageables, comme la génération de graphes dirigés avec double séquence de degrés (sortants et entrants), et pourraient facilement être mises en place, mais rien ne garantit à ce jour leur validité sans un apport substantiel au niveau théorique.

Chapitre 3

Inférence des propriétés de l'Internet

Dans ce chapitre, nous traitons d'un problème tout a fait différent de celui abordé dans le chapitre précédent : nous tentons de *quantifier* la qualité des mesures de la topologie `IP` basées sur l'outil `traceroute`.

3.1 Définition du problème traité

3.1.1 Contexte

Nous nous intéressons ici aux mesures basées sur le relevé (partiel) des *chemins* empruntés par les données transitant sur l'Internet, comme le permet l'outil `traceroute`. La donnée fournie par la mesure consiste donc en un ensemble de chemins correspondant à des suites de routeurs adjacents dans le réseau. Nous avons déjà introduit en Section 1.6.2 le fonctionnement de l'outil `traceroute`. Nous avons vu en Section 1.6.3 la manière dont la donnée des chemins permet d'obtenir un *graphe* représentant la topologie telle qu'on la voit. Ce graphe, que nous appellerons graphe image, image du réseau ou encore graphe échantillon, est l'objet central d'étude dans ce chapitre. Nous avons également montré en Section 1.6.3 les différentes manières possibles d'organiser les mesures à grande échelle. Nous nous intéressons ici au schéma de type Skitter représenté en bas à gauche de la Figure 1.8 où un ensemble relativement restreint de sources contrôlées par l'utilisateur relèvent les chemins empruntées par des sondes vers un ensemble de destinations commun à toutes les sources.

Nous avons vu en Section 1.6.4 que les données recueillies au cours d'une telle mesure sont par nature incomplètes et imparfaites, et donc biaisées d'un point de vue statistique. Ce biais a été reconnu par la communauté scientifique [64], puis étudié, ce qui a permis d'atteindre le consensus suivant [18, 86, 20, 1, 48] : si les assertions faites à propos de la topologie `IP` (comme par exemple l'hétérogénéité des degrés des nœuds du réseau) basées sur les mesures `traceroute` sont fiables *qualitativement*, les mesures *quantitatives* sont souvent trop biaisées pour être significatives. En d'autres termes, la mesure d'une *quantité* – comme le degré moyen, le coefficient de clustering, etc – sur une image de la topologie `IP`

aura *a priori* de grandes chances de dévier significativement par rapport à la valeur qu'on obtiendrait sur le réseau entier.

D'un point de vue statistique, la correction de ce type de biais se fait à l'aide d'*estimateurs* des quantités que l'on veut mesurer. La conception de tels estimateurs dans le cadre d'un schéma d'échantillonnage bien défini – comme c'est le cas ici, la collecte d'un image partielle de la topologie `IP` pouvant être vue comme un *échantillonnage* du graphe global – est une tâche courante dans le domaine des statistiques (voir par exemple [104]). De nombreux travaux ont été consacrés au contexte particulier de l'échantillonnage de graphes, principalement dans le domaine des sciences sociales (voir [38] et sa bibliographie). Ceux-ci font apparaître les difficultés liées à ce type de problème, et suggèrent que la performance des estimateurs dépend fortement du type d'échantillonnage, les effets dus aux particularités de la topologie du réseau sondé, et de la nature des quantités estimées. Notons que ces travaux ne traitent pas du cas particulier d'un échantillonnage basé sur la collecte de chemins comme c'est le cas dans les mesures faites avec `traceroute`.

Notre contribution dans ce chapitre sera d'établir une base de travail sur notre sujet particulier : l'estimation de grandeurs topologiques à partir de cartes `IP` obtenues par `traceroute`. Notre positionnement sera en réalité plus généraliste, et nous travaillerons essentiellement sur le plan théorique, en gardant à l'idée que notre travail doit principalement s'appliquer au cas de l'Internet.

Nous commençons par introduire les notations et rattachons notre problème à une catégorie bien connue d'estimation statistique, le problème de dénombrement des espèces, ce qui aura notamment des implications sur la difficulté de notre tâche. Nous nous focalisons ensuite sur l'estimation de la plus simple des grandeurs : la taille du réseau, que nous quantifions par le nombre de nœuds. Nous explorons ensuite les difficultés analytiques liées à notre problème, qui nous incitent à nous tourner vers des estimateurs non paramétrés, dont nous détaillons la conception. Nous présentons enfin des résultats empiriques basés sur des simulations quantifiant la qualité de ces estimateurs, ainsi que les résultats de leur application à l'Internet en se basant sur les mesures du projet Skitter [126]. Nous discutons enfin la qualité des résultats obtenus et la faisabilité d'estimateurs pour d'autres quantités – telles que le nombre de liens – basés sur les mêmes principes.

3.1.2 Notations

Reprenant les notations de graphes introduites dans la Section 1.4, nous représenterons le réseau mesuré comme un graphe $G = (V, E)$ où V est l'ensemble des sommets – ou nœuds – et $E \subset V \times V$ est l'ensemble des arêtes – ou liens. Nous notons respectivement $N = |V|$ et $M = |E|$ le nombre de sommets et d'arêtes. Pour modéliser une mesure du réseau, nous considérons un ensemble de n_S sources $S = \{s_1, \cdots, s_{n_S}\}$ situées dans le réseau, envoyant chacune des sondes vers un ensemble de n_T destinations $T = \{t_1, \cdots, t_{n_S}\}$ commun à toutes les sources. On a bien sûr $S, T \subset V$. Nous définissons également les *densités* de sources et de destinations par $q_S = n_S/N$ et $q_T = n_T/N$.

Pour chaque couple $(s, t) \in S \times T$, le chemin parcouru par les sondes entre s et t est collecté, et correspond à une suite de routeurs $s, r_1, \cdots, r_k, t$. Ce chemin comporte

une information topologique simple, à savoir l'existence des nœuds $r_1, \cdots, r_k$ traversés, et l'existence des liens $(r_1 - r_2), \cdots, (r_{k-1} - r_k)$. L'agrégation de tous les chemins collectés entre chaque couple (s, t) fournit donc un sous-graphe de G, que nous appellerons graphe échantillon, désigné par $G^* = (V^*, E^*)$. Les grandeurs relatives à ce graphe échantillon seront similairement accolées d'une étoile : N^* et M^* désigneront respectivement le nombre de sommets et d'arêtes de G^*.

3.1.3 Choix du modèle de routage

Nous n'avons pas spécifié la manière dont le chemin entre une source et une destination est choisi parmi les multitudes de possibilités : dans le cas de l'Internet le chemin suivi dépend du routage qui, comme nous l'avons déjà discuté en Section 1.3.2, est non trivial et ne peut être modélisé simplement de manière réaliste.

Insistons au contraire sur le fait que dans ce chapitre, les résultats et idées présentés ne dépendent pas du modèle de routage choisi, et peuvent aisément se généraliser à tout type d'échantillonnage de graphe basé sur la collecte de chemins. Cependant, lors des simulations menées plus loin en Section 3.4, nous avons bien dû fixer un modèle de routage pour pouvoir appliquer cet échantillonnage en pratique. Nous avons alors choisi la simplicité et modélisé les chemins entre les nœuds du réseau par les *plus courts chemins*. Pour plus de précision quand au choix du plus court chemin, nous renvoyons le lecteur à la Section 3.4 dans laquelle nous détaillons les conventions utilisées pour nos simulations.

Il est important de réaliser dès maintenant que notre étude ne prend pas en compte les phénomènes complexes et les anomalies rencontrées avec l'outil `traceroute`, dus par exemple à l'instabilité du routages ou aux congestions temporaires du réseau. Par exemple, nous ignorons complètement l'existence (pourtant bien établie) de chemins incomplets reportés par `traceroute` à cause de routeurs "invisibles" ne répondant pas aux sondes. Nous supposons donc que `traceroute` se comporte de manière idéale et fournit un chemin réaliste et unique entre chaque couple de nœuds du réseau. Cette approximation peut sembler grossière, mais reste logique dans le cadre de notre étude puisque le problème posé est difficile et très peu traité dans la littérature, et que nous avons favorisé une approche généraliste plutôt que trop spécifique à un modèle de l'Internet particulier – qui serait par ailleurs forcément imparfait. Nous enquêterons sur les imperfections des mesures `traceroute` de manière approfondie dans le Chapitre 4.

3.1.4 Liens avec le problème de dénombrement des espèces

Abordons le problème de manière statistique et considérons une quantité topologique caractéristique $\nu(G)$ du réseau sondé G, comme N, M, où encore la séquence de degrés $(deg(i))_{i \in V}$. Les valeurs observées de ces quantités (par exemple $N^* = |V^*|$ ou $M^* = |E^*|$) sous-estiment les valeurs réelles, sauf si l'échantillonnage est exhaustif auquel cas $G^* = G$. De nombreux travaux [64, 18, 86, 20, 1, 48] ont même montré que les valeurs observées peuvent être considérablement inférieures aux valeurs réelles (et donc que le graphe échantillon G^* est considérablement plus petit que G), induisant des biais significatifs et invali-

dant l'approximation $G \approx G^*$. Cela vient du fait que les explorations à l'aide de `traceroute` ne sondent pas le graphe uniformément, les `traceroute` ayant beaucoup plus tendance à traverser les nœuds "centraux" du graphe, par lesquels la majorité des chemins passent, que les nœuds de faible degré de la périphérie [20].

D'un point de vue statistique, l'estimation des quantités N, M et $(deg(i))_{i\in V}$ fait partie du plus général problème de dénombrement des espèces. Dans sa forme générale, ce problème fait référence à la situation où, ayant observé un certain nombre d'individus d'une population séparée en C classes (ou *espèces*), et connaissant les classes des individus observés, on désire estimer ce nombre C. Un exemple typique est celui du biologiste tentant d'estimer le (très grand) nombre d'espèces d'insectes dans la forêt amazonienne. Même s'il a pu observer un très grand nombre d'insectes, il peut rester loin d'avoir observé toutes les espèces, certaines étant très rares. Ce problème intervient dans de nombreux contextes et a reçu beaucoup d'attention en statistiques (voir [14] pour une revue du problème et une bibliographie conséquente). Citons pour mémoire le problème de l'estimation de la taille du vocabulaire d'un auteur, étant donné l'ensemble de ses œuvres : chaque mot correspond alors à une espèce et chaque occurrence de mot à une observation.

Alors que l'estimation de la *répartition* des individus, à savoir les tailles relatives des classes, est immédiate étant donnée une telle observation, l'estimation du *nombre* C de classes est souvent difficile à cause du potentiel grand nombre de classes (où espèces) présentes en très faible proportions, et ayant donc de grandes chances de ne pas observées.

Dans le cadre de notre échantillonnage de graphe, l'estimation de N, M et des degrés $(deg(i))_{i\in V}$ peut être reformulée en tant que problème de dénombrement des espèces. Considérons l'estimation de N : chaque nœud sera alors une espèce, et chaque occurrence d'un nœud dans un chemin correspondra à une observation : si un nœud $v \in V$ apparaît K fois parmi les $n_S n_T$ chemins relevés, on dira que l'espèce v a été observée K fois. De même, pour l'estimation de M on considère que chaque lien correspond à une espèce, et pour les degrés une espèce v correspondra à l'ensemble des arêtes adjacentes au sommet v.

On peut imaginer d'autre quantités liées ainsi au problème de dénombrement des espèces, mais dans la suite nous restons focalisés sur la plus simple de ces quantité :N. Il semble en effet illusoire d'espérer produire des estimateurs de M, voire des $(deg(i))_{i\in V}$, si on ne peut d'abord estimer convenablement N.

3.1.5 Description synthétique

Pour plus de clarté, nous résumons ici brièvement le problème traité : connaissant le résultat de mesures `traceroute` à grande échelle, nous cherchons à déterminer le nombre de nœuds du réseau sondé. Plus précisément, les données sont :

- Le graphe G^* obtenu par la mesure
- L'ensemble $S = \{s_1, \cdots, s_{n_S}\}$ des sources à partir desquelles `traceroute` a été lancé.
- L'ensemble $T = \{t_1, \cdots, t_{n_T}\}$ des destinations vers desquelles `traceroute` a été lancé.

Le but est alors de produire un estimateur $\hat{N}$ de N, le nombre de sommets de G.

Dans une approche plus généraliste du problème, considérant un graphe G et deux sous-ensemble $S, T \subset G$, il suffit de définir G^* comme l'agrégation des chemins obtenus

entre tous les couples $(s,t) \in S \times T$. La définition d'un *chemin* est alors ce qui fait la spécificité du problème traité.

3.2 Méthode analytique pour l'inférence de N

La première approche pour l'estimation de N à partir de S, T et G^* est d'analyser la manière dont G^* dépend de G, S et T, en se focalisant bien entendu sur le nombre de sommets observés. Une première intuition simple est en effet de chercher une relation liant N^*, n_S, n_T et N afin de pouvoir estimer N en fonction des trois autres. Nous montrons dans cette section que la tâche est plus difficile qu'il n'y paraît au premier abord, justifiant ainsi l'approche non-paramétrique adoptée par la suite.

Afin de cadrer les choses et de simplifier au maximum notre analyse, nous restreignons notre propos dans cette Section en considérant que les chemins sont déterminés de manière unique et correspondent au plus courts chemins dans un graphe (voir Section 1.4.4). Il existe en général plusieurs plus courts chemins de longueur identiques entre chaque couple de sommets, mais il suffit d'en privilégier un au hasard (pour chaque couple) pour obtenir l'unicité dans la mesure. Voir la Section 3.1.3 pour plus de détails.

3.2.1 Centralité de plus court chemin

Une quantité cruciale dans l'échantillonnage basé sur la collecte de plus courts chemins est la *centralité de plus court chemin*, qui compte précisément pour chaque nœud v le nombre total de plus courts chemins passant par v (sans compter les chemins commençant ou terminant par v). Si nous appelons D_{hj} le nombre total de plus courts chemins du nœud h au nœud j, et que $D_{hj}(i)$ est le nombre de ces chemins passant par le nœud i, la centralité[1] b_i de i est définie [2] par

$$b_i = \sum_{\substack{h,j \in V^2 \\ j \neq h \neq i}} D_{hj}(i)/D_{hj}$$

La quantité b_i est directement proportionnelle au nombre *attendu* d'observations du nœud i, puisque la mesure consiste en une collecte de plus courts chemins. Notons que la formule fait apparaître une normalisation de $D_{hj}(i)$ par D_{hj}, cela est dû au fait qu'un seul chemin sera choisi – au hasard – entre h et j parmi les D_{hj} plus courts chemins possibles.

On peut facilement montrer [41] que la distance moyenne $\bar{d}$ (comme définie en Section 1.4.4) est liée aux b_i par la relation

$$\sum_{i \in V} b_i = N(N-1)(\bar{d}-1)$$

[1] Nous utilisons la lettre b pour désigner la centralité à cause du terme anglo-saxon plus répandu : *betweenness centrality*, ou tout simplement betweenness.

[2] La définition de centralité donnée ici peut s'étendre à des modèles de routage qui ne considèrent pas forcément les plus courts chemins. Voir par exemple [82] pour une mesure basée sur des marches aléatoires.

qui peut être reformulée sous la forme

$$N = 1 + \frac{E[b]}{\bar{d} - 1} \tag{3.1}$$

où $E[b]$ désigne la moyenne de b_i sur l'ensemble des nœuds.

Nos résultats empiriques suggèrent que la distance moyenne $\bar{d}$ peut être estimée assez précisément lors d'un échantillonnage basé sur les plus courts chemins, ce qui n'est pas étonnant puisqu'il suffit d'évaluer directement la distance moyenne observée entre les couples de sommets considérés (en supposant que le choix des sources S et des destinations T ne biaise pas la distance moyenne entre nœuds du réseau). On peut donc lier l'estimation de N directement à l'estimation de $E[b]$.

Il s'avère que de nombreux réseaux réels, et en particulier l'Internet, présentent une distribution de centralité très hétérogène parmi les sommets, approchée asymptotiquement par une loi de puissance [85]. Nos simulations, dont nous montrons un exemple en Figure 3.1, ont confirmées cette assertion, et montrent même la stabilité de l'exposant de la loi de puissance entre graphe original et graphe échantillonné : on peut donc raisonnablement estimer l'exposant donnant le comportement asymptotique de la distribution de b à partir de la mesure de cet exposant sur les distributions de b^* dans les graphes échantillonnés G^*.

Jusqu'ici, tout semble s'accorder pour permettre une bonne estimation de N grâce notamment aux bonnes estimations de $\bar{d}$ et de l'exposant β de la loi de puissance approchant la distribution de b. Cependant, il est important de savoir que la distribution de b n'est en pratique pas exactement une loi de puissance, et en dévie même assez fortement pour les valeurs faibles, comme souligné sur la Figure 3.1. Il est plus réaliste de modéliser la loi distribution de b comme une loi mixte [74] :

$$P(b) = \nu P_1(b) + (1 - \nu) P_2(b) \tag{3.2}$$

P_1 est la loi de distribution pour les faibles valeurs $b \in [1, b_{min}[$, pour un b_{min} petit, et $P_2(b)$ est la distribution pour les grandes valeurs $b \in [b_{min}, b_{max}], b_{max} >> b_{min}$ qui suit une loi de puissance $P_2(b) = b^{-\beta}/K$, K étant un terme de normalisation. On considère que b n'est jamais supérieure à b_{max}, qui est la centralité maximale dans le graphe. Enfin, ν est le facteur de normalisation entre les deux distributions.

3.2.2 Une approche analytique basée sur la centralité

Supposons à présent que l'Équation 3.2 est vérifiée, et concentrons-nous sur l'estimation de $E[b]$, qui revient comme nous l'avons déjà vu à estimer N. L'espérance $E[b]$ est une pondération des espérances des deux distributions considérées : $E[b] = \nu E_1[b] + (1-\nu)E_2[b]$. Il nous faut donc estimer les deux espérances $E_1[b]$ et $E_2[b]$, ainsi que la pondération ν. Les difficultés du problème apparaissent alors de manière bien cernée : l'estimation de la première moyenne $E_1[b]$ est difficile car elle concerne les nœuds sur lesquels on reçoit justement le moins d'informations, et de même la pondération ν correspond à la proportion de ces

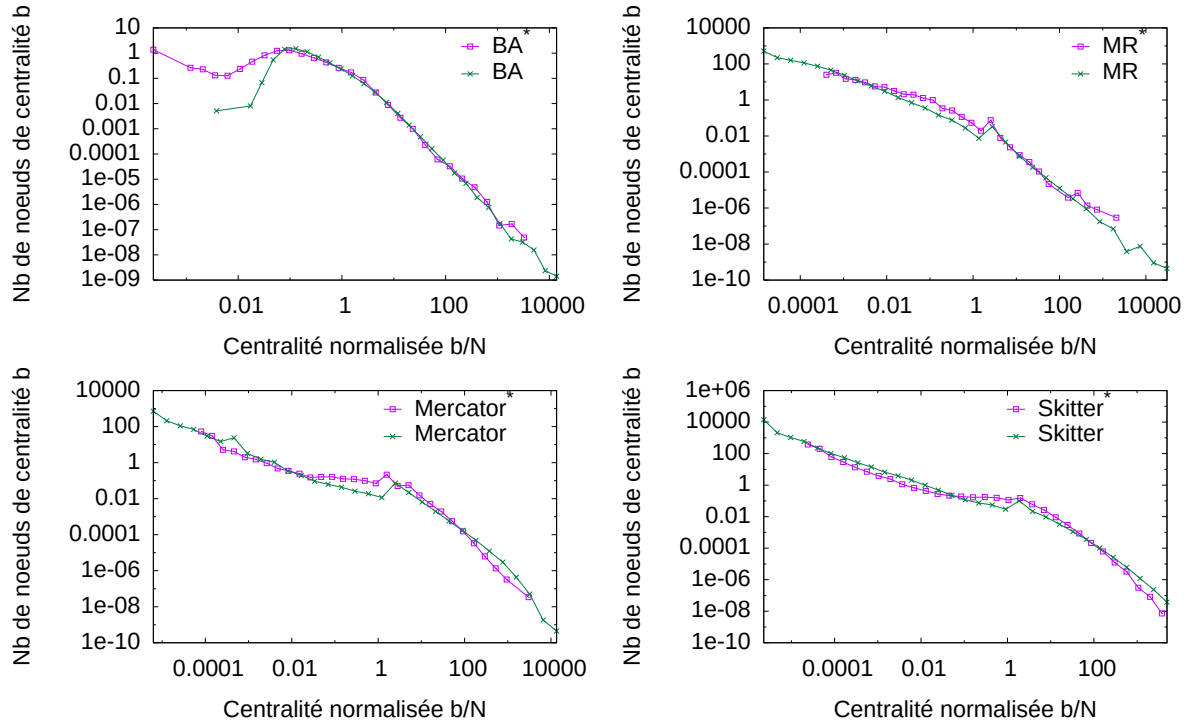

FIG. 3.1 – Distribution de centralité dans plusieurs graphes G et dans leurs échantillons G^* respectifs. Sur chaque tracé, les centralités ont été calculées sur le graphe originel G et sur un graphe échantillon G^* obtenu par collecte de chemins avec $n_S = 10$ sources et une densité de destinations de $q_T = 10\%$. Les tracés correspondent à la centralité normalisée b/N, plus facile à comparer entre graphes de tailles différentes (la moyenne de b/N est de $\bar{d}$, voir Equation 3.1). Les graphes utilisés correspondent en haut à deux graphes synthétiques : à gauche, BA désigne un graphe issu du modèle d'attachement préférentiel (voir Section 1.5) et à droite MR correspond à un graphe aléatoire dont les degrés évoluent selon une loi de puissance ajustée, généré selon le modèle décrit dans le Chapitre 2. Ces deux graphes ont pour paramètres $N = 10^6$ sommets et un degré moyen $z = 6$. En bas, les graphes originels sont eux-mêmes des cartes (partielles) de la topologie `IP`, à gauche celle obtenue par le projet MERCATOR [44] et à droite une autre carte obtenue par le projet Skitter [126], ayant des tailles et degrés moyens respectifs $N = 228263, z = 2.81$ et $N = 624324, z = 3.77$.

nœuds rarement observés, qui semble similairement difficile à estimer. Ces difficultés sont caractéristiques du problème de dénombrement des espèces et du manque d'informations sur les espèces rares. Intéressons nous tout de même à l'espérance $E_2[b]$: nous obtenons, puisque le facteur de normalisation K vaut $K = \int_{b_{min}}^{b_{max}} b^{-\beta} db$:

$$E_2[b] = \frac{1}{K} \int_{b_{min}}^{b_{max}} b^{1-\beta} db = \frac{{b_{min}}^{2-\beta} - {b_{max}}^{2-\beta}}{{b_{min}}^{1-\beta} - {b_{max}}^{1-\beta}} \frac{\beta - 1}{\beta - 2}$$

Ce qui revient à

$$E_2[b] \;=\; \frac{\beta-1}{\beta-2}\cdot b_{min}\cdot\frac{1-\delta^{\beta-2}}{1-\delta^{\beta-1}}\;,\quad \text{où } \delta=\frac{b_{min}}{b_{max}}.$$

On peut ensuite approcher b_{max} par une méthode de majoration [29] revenant à dire que le nombre attendu de sommets ayant une centralité supérieure à b_{max} est inférieur à 1, en supposant que la loi de probabilité $P(b)$ reste asymptotiquement conforme quand $b \to \infty$ avec son comportement sur $[b_{min}, b_{max}]$. Cela donne :

$$N \times \int_{b_{max}}^{\infty} P(b)db \sim 1 \quad \rightarrow \quad b_{max} \sim \left(\frac{(1-\nu)N}{(\beta-1)K}\right)^{\frac{1}{\beta-1}}. \tag{3.3}$$

En supposant $b_{min} << b_{max}$ et donc $K \sim {b_{min}}^{1-\beta}/(\beta-1)$, nous obtenons

$$b_{max} \sim b_{min}\,((1-\nu)N)^{\frac{1}{\beta-1}} \tag{3.4}$$

ce qui donne pour δ la valeur approchée

$$\delta \sim ((1-\nu)N)^{\frac{1}{\beta-1}} \tag{3.5}$$

et, en supposant (ce qui semble tout à fait légitime dans notre cas) que $1 << (1-\nu)N$, et donc que $\delta << 1$, et aussi que β est proche de 2 ou supérieur à 2, et donc que $1-\delta^{\beta-1} \sim 1$, nous obtenons finalement l'approximation :

$$E_2[b] \sim \frac{\beta-1}{\beta-2}\cdot b_{min}\cdot\left(1-((1-\nu)N)^{-\frac{\beta-2}{\beta-1}}\right). \tag{3.6}$$

$E_2[b]$ est donc fortement dépendante du rapport $\frac{\beta-1}{\beta-2}$. Malheureusement, comme nous l'avons vu sur la Figure 3.1, les valeurs de β dans le cas de l'Internet, en se basant sur les mesures effectuées sur les cartes `IP` actuelles, semblent très proches de 2 – comme c'est le cas dans de nombreux autres réseaux réels [12, 41]. Cela implique une instabilité considérable lors de l'estimation de $E_2[b]$ par rapport à l'incertitude dans l'estimation de β.

Pour résumer tout cela, malgré le bon comportement attendu de l'estimation de quantités comme $\bar{d}$ ou β à partir d'échantillonnage de chemins, l'estimation de $E[b]$ semble compromise par deux facteurs principaux : le manque d'informations sur les nœuds de faible centralité (affectant l'estimation de $E_1[b]$ et ν) et l'instabilité pathologique de la valeur de $E_2[b]$ par rapport à l'estimation de β. N'ayant pas pu trouver d'autre approche analytique que celle basée sur la centralité de plus court chemin, nous nous sommes naturellement tournés vers une approche non-paramétrique, que nous détaillons dans la prochaine section.

3.3 Estimateurs de la taille du réseau

Nous présentons dans cette section deux estimateurs du nombre N de nœuds dans le réseau, se basant essentiellement sur le nombre de nœuds observé N^* dans G^* et sur les quantités n_S et n_T. Nos estimateurs sont de la forme $\hat{N} = N^*/\hat{\alpha}$, où $\alpha \in]0,1[$ est le *taux de découverte* N^*/N, et $\hat{\alpha}$ est un estimateur de α. Dans les deux cas, nos estimateurs se basent sur des méthodes statistiques de *sous-échantillonnage*, où l'on part de G^* et tente d'extraire de l'information à partir de sous-graphes de G^*.

3.3.1 Re-échantillonnage

Une méthode populaire de sous-échantillonnage est la méthode de *re-échantillonnage*, qui est à la base de la technique "bootstrap" bien connue en statistiques [31]. Étant donné un échantillon X^* d'une population X, le principe du re-échantillonnage consiste à échantillonner X^* *de la même manière* qu'on a échantillonné X, obtenant ainsi X^{**}, échantillon de l'échantillon. L'idée est ensuite de supposer que les relations entre X^{**} et X^* sont les mêmes que celles entre X^* et X, et donc d'estimer une quantité $\nu(X)$ par extrapolation à partir de $\nu(X^*)$ et $\nu(X^{**})$.

Dans notre contexte, la "relation" entre l'original et l'échantillon sera simplement le taux de découverte $\alpha = N^*/N$, on estimera donc N grâce à la connaissance de N^* et du taux de découverte $\alpha^* = N^{**}/N^*$ obtenu lors du re-échantillonnage : $\hat{N} = N^*/\alpha^*$. Toute la difficulté réside dans le fait qu'il faille re-échantillonner dans des conditions similaires à l'échantillonnage original.

Les caractéristiques du re-échantillonnage que nous avons retenu sont simples : on choisit un ensemble S^* de ${n_S}^*$ sources et un ensemble T^* de ${n_T}^*$ destinations au hasard parmi les N^* nœuds de G^*, et on applique notre échantillonnage basé sur la collecte des ${n_S}^*{n_T}^*$ plus courts chemins entre les couples $(s,t) \in S^* \times T^*$. Dans ce cadre, reproduire les caractéristiques de l'échantillonnage originel revient donc simplement à choisir les *bons* nombres de sources et de destinations ${n_S}^*$ et ${n_T}^*$. Notons que nous avons également exploré la possibilité de restreindre le choix des sources du re-échantillonnage aux sources utilisées pour l'échantillonnage originel et de même pour les destinations, ce qui revient à poser $S^* \subset S$ et $T^* \subset T$, mais avons obtenu de moins bons résultats. Nous garderons donc notre convention de choix aléatoire $S^*, T^* \subset V^*$ par la suite.

Définissons la densité de destinations du re-échantillonnage ${q_T}^* = {n_T}^*/N^*$. Nous avons observé lors de nombreuses simulations que pour des paramètres réalistes (du même ordre de grandeur que ceux du projet Skitter [126]), c'est-à-dire pour un nombre de sources relativement petit par rapport à la taille du réseau et un nombre de destinations grand, le taux de découverte attendu $E[\alpha] = E[N^*]/N$ s'approche d'une fonction continue de q_T, pour un nombre de sources n_S fixé. De plus, nous avons observé que pour des échantillons G^* suffisamment représentatifs (typiquement, tels que $N^*/N > 1\%$), le taux de découverte attendu lors du re-échantillonnage correspondait assez bien au taux de découverte obtenu lors de l'échantillonnage pour peu que ${n_S}^* = n_S$ et ${q_T}^* = q_T$. En d'autres termes,

connaissant n_S, n_T et N^*, nous obtenons l'équation :

$$\left.\begin{array}{r} {n_S}^* = n_S \\ {q_T}^* = q_T \end{array}\right\} \Rightarrow \frac{E[N^{**}]}{N^*} = \frac{N^*}{N}, \tag{3.7}$$

où l'espérance $E[N^{**}]$ signale que l'on est à même de simuler le re-échantillonnage plusieurs fois avec des choix de sources et de destinations aléatoires et de moyenner les observations pour obtenir des résultats plus stables et fiables. Cette équation est essentiellement fondée sur des résultats théoriques connus [20] et sur notre propre expérience, comme nous le montrons en Figure 3.2. Cette Figure présente une étude comparative des taux de découverte α et α^* obtenus lors de l'échantillonnage et du re-échantillonnage de plusieurs graphes, synthétiques ou issus d'une mesure de la topologie `IP`. Les graphes synthétiques choisis correspondent à des modèles bien connus introduits en Section 1.5, à savoir le modèle de croissance et le modèle aléatoire pur, notés respectivement BA et ER en références aux articles de Barabási et Albert [9] et d'Erdös et Renyi [33] qui les ont rendus populaires. L'idée était d'éprouver la validité de notre assertion sur des familles de graphes de natures très différentes, les graphes ER étant homogènes alors que les graphes BA sont fortement hétérogènes. Pour le graphe censé représenter un réseau réel, nous avons choisi le graphe de la topologie `IP` collecté par Skitter (voir la Section 3.4.1 pour plus de détails). Sur la Figure 3.2, on peut voir que l'égalité entre le taux de découverte obtenu lors de l'échantillonnage et celui obtenu lors du re-échantillonnage n'est pas parfaite, mais qu'elle reste relativement bien approchée.

L'estimation de N se produit alors de la manière suivante : on re-échantillonne le graphe G^* un certain nombre B de fois, avec des paramètres ${n_S}^*$ et ${q_T}^*$ correspondant à n_S et q_T, obtenant ainsi des graphes $G_1^{**}, \cdots, G_B^{**}$ de tailles $N_1^{**}, \cdots, N_B^{**}$, puis on calcule la moyenne de ces quantités $\bar{N}^{**} = (1/B)\sum_{k=1}^{B} N_k^{**}$, et on obtient enfin l'estimateur de re-échantillonnage $\hat{N_{RE}}$:

$$\hat{N_{RE}} = N^* \cdot \frac{N^*}{\bar{N}^{**}} \tag{3.8}$$

.

Le problème de cette méthode est qu'elle exige la connaissance de q_T puisque l'on doit ajuster ${n_T}^*$ de manière à ce que ${q_T}^* = q_T$. Or on connaît n_T mais pas N, et donc $q_T = n_T/N$ non plus. Pour résoudre ce problème, nous utilisons une astuce basée sur une reformulation de l'Équation 3.7, en remarquant que l'égalité ${q_T}^* = q_T$ implique $\frac{{n_T}^*}{N^*} = \frac{n_T}{N}$ et donc $\frac{{n_T}^*}{n_T} = \frac{N^*}{N}$:

$$\left.\begin{array}{r} {n_S}^* = n_S \\ {q_T}^* = q_T \end{array}\right\} \Rightarrow \frac{E[N^{**}]}{N^*} = \frac{{n_T}^*}{n_T}. \tag{3.9}$$

Cette fois, l'observation $\frac{E[N^{**}]}{N^*} = \frac{{n_T}^*}{n_T}$ n'implique que des *quantités mesurables*. Or, la concavité et la sous-linéarité en q_T – observées empiriquement et justifiées par des arguments théoriques simples[3] – de la fonction $E[N^*]/N = E[\alpha](q_T)$ à n_S fixé implique l'unicité (à

[3]La fonction $E[\alpha](q_T)$ varie exactement comme la fonction $E[N^*](n_T)$. En étudiant $\Delta(n_T) = E[N^*](n_T+1) - E[N^*](n_T)$, qui correspond au nombre moyen de *nouveaux* sommets découverts en ajou-

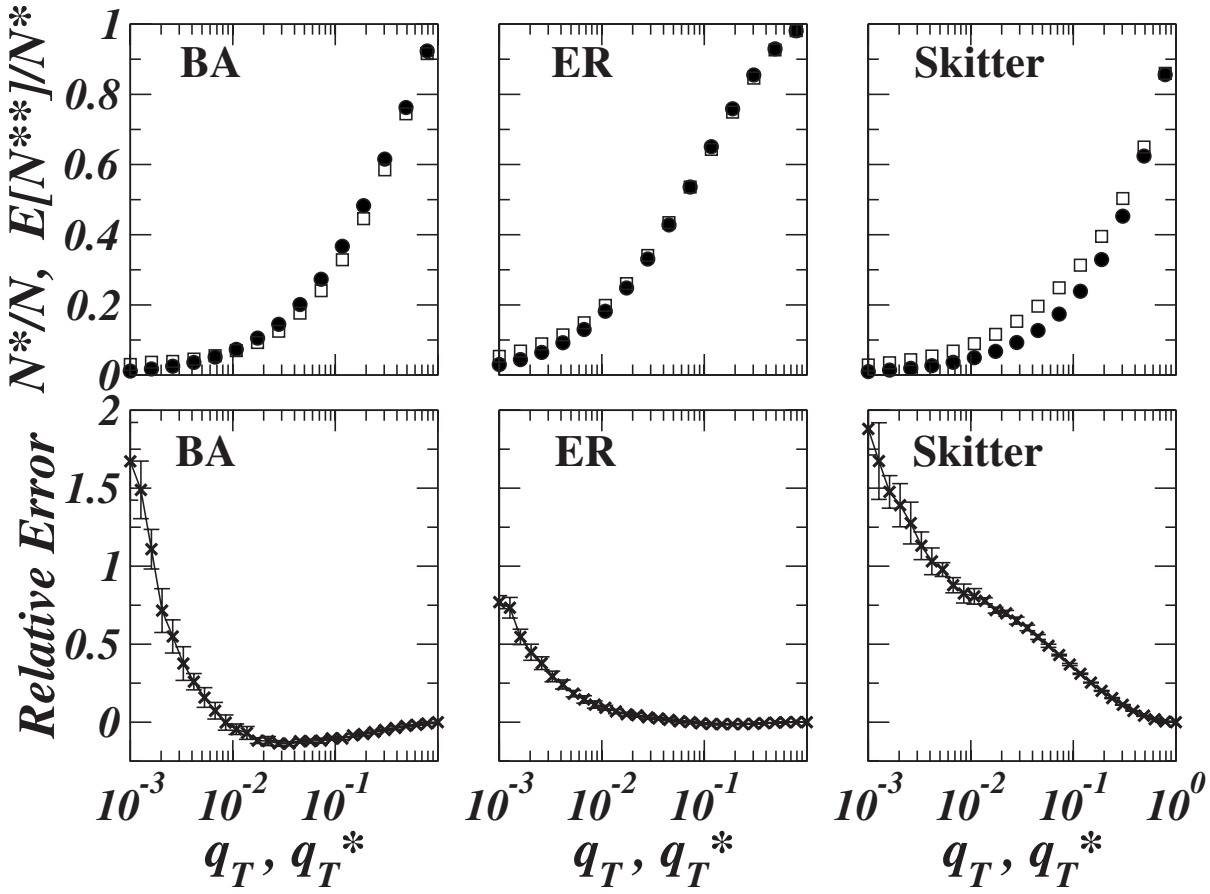

FIG. 3.2 – Comparaison des quantités N^*/N (ronds noirs) et $E[N^{**}/N^*]$ (carrés blancs) en fonction de $q_T = {q_T}^*$, pour les trois topologies décrites en Section 3.4.1. On a posé ici $n_S = {n_S}^* = 10$. Les tracés du haut montrent les moyennes de $\alpha = N^*/N$ et de $E[\alpha^*] = E[N^{**}]/N^*$ sur 10 échantillonnages de G^* (pour chaque G^*, 10 re-échantillonnages sont lancés pour évaluer $E[N^{**}]$). Les tracés du bas montrent l'erreur relative moyenne $(E[\alpha^*] - \alpha)/\alpha$ entre ces deux quantités, ainsi que des barres d'erreurs correspondant à l'écart-type de cette erreur. La forme des courbes en haut corrobore les approximations posées dans l'Equation 3.7 et la partie basse met en évidence une erreur d'autant plus faible que q_T est élevé. On remarque également que $E[N^{**}]/N^*$ à tendance à surestimer N^*/N – sauf dans le cas du graphe BA pour les q_T élevés.

$n_S = {n_S}^*$ fixé) du point ${q_T}^*$ où l'observation $\alpha^*({q_T}^*) = \alpha(q_T)$ sera réalisée. En d'autres

tant une destination, il est clair que ce nombre est d'autant plus faible que N^* – le nombre de sommets déjà découverts – est grand. Comme N^* est une fonction croissante de n_T, $\Delta(n_T)$ est décroissante, d'où la concavité de $E[\alpha](q_T)$. La sous-linéarité découle alors de l'observation que $\Delta(n_T) < n_S \, \forall n_T$.

termes, l'Équation 3.9 devient une *équivalence* :

$$n_S{}^* = n_S \;\Rightarrow \left(q_T{}^* = q_T \iff \frac{E[N^{**}]}{N^*} = \frac{n_T{}^*}{n_T}. \right) \tag{3.10}$$

En schématisant comme en Figure 3.3 l'évolution de $\frac{E[N^*]}{N^*}$ en fonction de $\frac{n_T{}^*}{n_T}$ lors du re-échantillonnage, il suffit simplement de procéder à une dichotomie pour retrouver le point où $\frac{E[N^{**}}{N^*} = \frac{n_T{}^*}{n_T}$. Pour chaque $n_T{}^*$ rencontré lors de la dichotomie, on calcule donc ces deux quantités avec d'autant plus de précision – en augmentant B on augmente la précision du calcul de $E[N^{**}]$ – qu'on est proche du point d'égalité. Au niveau du point d'égalité, on est en mesure de produire l'estimateur voulu puisque la condition $q_T{}^* = q_T$ est vérifiée.

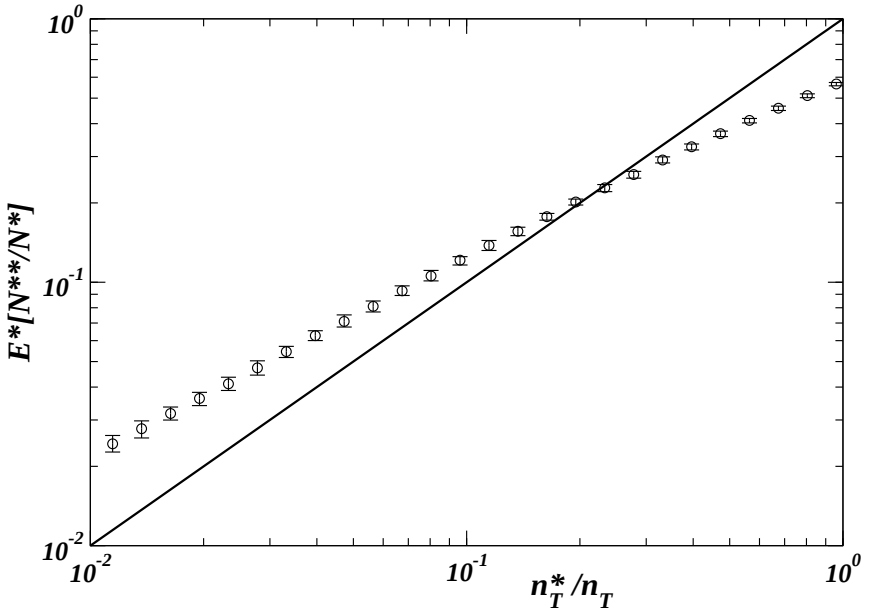

FIG. 3.3 – Illustration graphique de la méthode d'obtention du point vérifiant $q_T{}^* = q_T$: superposition des courbes $y = x$ et $y = E[N^{**}]/N^*$, $x = n_T{}^*/n_T$. Le graphe G choisi est synthétique, généré selon le modèle d'attachement préférentiel, et correspond au graphe BA utilisé et décrit en Section 3.4.1. On a ici $N = 10^5$, $z = 6$, et l'échantillonnage est fait avec $n_S = 10$ sources et $n_T = 10^4$ destinations (soit $q_T = 0.1$). Le graphe échantillon G^* obtenu vérifie $N^* = 33178$. La courbe montre donc la moyenne de N^{**}/N^* obtenue en échantillonnant G^* avec $n_S{}^* = n_S = 10$ sources, pour un $n_T{}^*$ variable. Les barres d'erreurs montrent l'écart-type de cette grandeur qui varie en fonction du choix des sources et des destinations utilisés dans le re-échantillonnage. L'intersection avec $y = x$ est bien unique, et peut donc se déterminer par dichotomie.

En réalité, et comme nous l'avons montré en Figure 3.2, l'Équation 3.10 n'est vraie qu'en valeur approchée, et notre technique présente l'inconvénient de l'exploiter deux fois, une fois dans chaque sens, afin de localiser le point $q_T{}^* = q_T$ puis de produire l'estimateur,

ce qui multiplie les imprécisions. Un autre facteur aggravant est la faible différence de pente (voir Figure 3.3) entre la courbe $y = \frac{E[N^{**}]}{N^*}, x = \frac{{n_T}^*}{n_T}$ et la droite $y = x$, qui multiplie également le facteur d'imprécision dans la détection du point d'intersection de ces courbes. Ces problèmes d'imprécision sont réels et leur impact est démontré dans la Section 3.4, mais nous n'avons pu trouver un moyen de contourner ces difficultés.

3.3.2 Méthode du Delta

Une autre technique populaire de sous-échantillonnage est la stratégie "leave one out", que nous désignerons par méthode du Delta et qui est à la base de la méthode dite *jack-knifing* ou encore la méthode de validation croisée [31]. Ce principe peut être appliqué à notre problème d'estimation de N, et ce sans imposer les conditions contraignantes posées en [112].

Rappelons que l'ensemble des nœuds V^* collectés lors de l'échantillonnage contient notamment $S = \{s_1, \cdots, s_{n_S}\}$ et $T = \{t_1, \cdots, t_{n_T}\}$, les sources et destinations d'origine. Notre approche consiste à lier la quantité N à la probabilité avec laquelle les destinations t_j sont présentes *au milieu d'un chemin* de l'échantillonnage – on ne compte donc pas la présence triviale de t_j à l'extrémité des chemins vers t_j. Pour formaliser les choses, désignons par $T \setminus \{t_j\}$ l'ensemble des destinations privé de t_j, et $V^*_{(-j)}$ l'ensemble des sommets découverts par l'échantillonnage des chemins entre S et $T \setminus \{t_j\}$. Définissons $\delta_j = I\left(t_j \in V^*_{(-j)}\right)$ l'indicateur de l'évènement "t_j est présent sur au moins l'un des chemins collecté vers les autres destinations". Le nombre total de destinations visibles parmi les chemins collectés vers les autres destinations peut s'écrire $X = \sum_{j=1}^{n_T} \delta_j$. Notre idée est d'établir une relation entre X et N. La mesure de X pendant l'échantillonnage est facile à réaliser : il suffit de faire abstraction des chemins vers une destination t_j pour détecter la présence de t_j parmi les chemins vers d'autres destinations, et cela pour chaque destination. La mesure de X devrait ensuite permettre l'estimation de N grâce à la relation établie. Notons que cette mesure peut se faire sans problème *après* l'échantillonnage à condition d'avoir gardé la donnée des chemins collectés, et pas seulement G^*.

Une assertion cruciale pour la validité de notre estimateur est le caractère *aléatoire* du choix des destinations : nous supposons que les destinations $t_1, \cdots, t_{n_T}$ correspondent à un choix aléatoire uniforme de n_T nœuds parmi les $N - n_S$ nœuds de $V \setminus S$. Nous aurions également pu supposer que les destinations et les sources peuvent coïncider (et seraient alors choisies parmi les N nœuds de V), cela ne changerait ni les arguments présentés ici, ni la forme des résultats, et aurait en réalité très peu d'incidence étant donné le faible nombre de sources utilisé en pratique. Étant donné une telle contrainte, la probabilité que la destination t_j soit présente dans $V^*_{(-j)}$ est simplement donné par le rapport entre le cardinal de $V^*_{(-j)} \setminus (S \cup T \setminus \{t_j\})$ et de celui l'ensemble des nœuds parmi lesquels on tire t_j aléatoirement, à savoir $V \setminus (S \cup T \setminus \{t_j\})$. La soustraction de $S \cup T \setminus \{t_j\}$ vient de la contrainte de choisir des sources et destinations *distinctes* deux à deux. Notons que

$(S \cup T \setminus \{t_j\}) \subset V^*_{(-j)}$. On obtient donc :

$$Pr\left(\delta_j = 1 | V^*_{(-j)}\right) = \frac{N^*_{(-j)} - (n_S + n_T - 1)}{N - (n_S + n_T - 1)} \ , \tag{3.11}$$

où $N^*_{(-j)} = \left|V^*_{(-j)}\right|$. Notons que par symétrie l'espérance $E\left[N^*_{(-j)}\right]$ est la même pour tous les j, on peut donc noter cette quantité $E\left[N^*_{(-)}\right]$. On obtient alors l'identité suivante :

$$E[X] = \sum_{j=1}^{n_T} Pr\left(\delta_j = 1 | V^*_{(-j)}\right) = \frac{n_T \cdot \left(E\left[N^*_{(-)}\right] - (n_S + n_T - 1)\right)}{N - (n_S + n_T - 1)} \tag{3.12}$$

Ce qui revient, en isolant N, à

$$N = n_S + n_T - 1 + \frac{n_T \cdot \left(E\left[N^*_{(-)}\right] - (n_S + n_T - 1)\right)}{E[X]} \tag{3.13}$$

On obtient alors un estimateur de N en estimant les quantités $E[X]$ et $E\left[N^*_{(-)}\right]$. Pour la deuxième il semble naturel d'utiliser l'estimateur $\bar{N^*_{(-)}} = (1/n_T)\sum_{j=1}^{n_T} N^*_{(-j)}$, qui est non biaisé, ne pose aucun problème arithmétique et se mesure directement. Pour $E[X]$ on peut être tenté d'utiliser X, qui est également mesurable, mais le cas $X = 0$ pose problème puisqu'il provoquerait une division par zéro.

Le cas $X = 0$ intervient quand aucune des destinations t_j n'est observée dans $V^*_{(-j)}$, ce qui peut arriver notamment si l'échantillon G^* est trop petit par rapport au graphe originel. Typiquement, on peut atteindre ce type de comportement dès que N^* est de l'ordre de ou plus petit que N/n_T, car alors l'espérance de X devient inférieure à 1. Dans ce cas, nous n'avons aucune information sur une possible majoration de N, qui peut être *a priori* infiniment grand. Le fait que $X = 0$ transporte néanmoins l'information que N est grand, aussi préférons-nous modifier légèrement notre estimateur pour renvoyer un résultat dans le cas $X = 0$. En l'occurrence, notre stratégie est d'estimer directement la quantité $1/X$. Pour cela, rappelons que X est la somme des δ_j qui sont des variables de Bernoulli, et supposons dans un premier temps que ces variables soient indépendantes et identiquement distribuées avec probabilité $p = Pr(\delta_j = 1) = E[N^*_{(-)}]/N$. Dans ce cas, X obéit à une loi binomiale de paramètres n_T et p, ce qui nous permet d'obtenir la relation suivante, en posant $q = (1 - p)$:

$$p \cdot E\left[\frac{n_T + 1}{X + 1}\right] = 1 - q^{n_T+1}.$$

On voir alors que la quantité $\frac{n_T+1}{X+1}$ a pour espérance $p^{-1}(1 - q^{n_T+1}) \approx p^{-1}$, et constitue donc un estimateur approximativement non biaisé de p^{-1}, ce qui nous donne à un facteur n_T près un estimateur de X^{-1}. Finalement, nous remplaçons $n_T/E[X]$ par $(n_T+1)/(X+1)$

et l'estimateur devient donc :

$$\hat{N_\Delta} = n_S + n_T - 1 + \frac{(n_T+1)\left(\bar{N^*_{(-)}} - (n_S+n_T-1)\right)}{X+1} \tag{3.14}$$

Justifions à présent notre approximation δ_j comme des variables de Bernoulli de probabilité $E[N^*_{(-)}]/N$ au lieu de $E[N^*_{(-j)}]/N$: cela vient du fait que les $N^*_{(-j)}$ sont en pratique tous très proche de N^* quand n_T est grand et quand le nombre $N^* - N^*_{(-j)}$ de nouveaux nœuds découverts en ajoutant la destination n_T reste très faible par rapport au nombre de nœuds déjà découverts[4]. L'approximation $E[N^*_{(-j)}] \approx E[N^*_{(-)}]$ est donc très bonne.

En incorporant ce dernier argument $N^*_{(-)} \approx N^*$, qui suggère également le remplacement de $\bar{N^*_{(-)}}$ par $N^* - 1$, notre formule devient :

$$\hat{N_\Delta} = n_S + n_T + \frac{N^* - (n_S+n_T)}{1-w^*}, \tag{3.15}$$

où $w^* = (n_T - X)/(n_T+1)$. En d'autres termes, $\hat{N_\Delta}$ s'obtient en comptant les n_S+n_T sources et destinations, puis en multipliant le nombre $N^* - (n_S+n_T)$ de sommets observés restants par un facteur dilatant $(1-w^*)^{-1}$. Cette forme est analogue à une méthode classique dans la littérature statistique sur le problème de dénombrement des espèces, due à Good [42], dans laquelle le nombre d'espèces observées est ajusté par un facteur estimant la proportion de la population qui n'a pas été observée. Les estimateurs de ce type sont typiquement désignés comme des estimateurs de couverture, dont les qualités – relativement aux autres alternatives – ont été reconnues [14].

Pour finir, avançons que la qualité de notre estimateur dépend essentiellement de X : plus celui-ci sera grand, plus l'estimation sera fidèle car basée sur plus d'information. Une garantie intéressante de qualité apparaît quand on s'aperçoit que pour $n_T > \sqrt{N}$, comme $N^* \geq n_T$, on obtient forcément $\delta_j = E[N^*_{(-)}]/N \geq 1/\sqrt{N}$ d'où $E[X] \geq 1$. On peut donc s'attendre à des estimations de qualité à partir du moment où q_T dépasse largement $1/\sqrt{N}$, ce qui semble tout à fait réalisable en pratique.

3.4 Validation empirique

Les estimateurs développés dans la Section 3.3 reposent sur des approximations, et nécessitent donc une validation pratique. Afin de s'assurer de leur validité, nous avons donc lancé un grand nombre de simulations où le graphe de base G est connu. Un ensemble S de sources et un ensemble T de destinations sont extraits de G et permettent de l'échantillonner pour obtenir un graphe échantillon G^*. Ces mesures sont fournis à nos estimateurs, et nous comparons leurs estimations $\hat{N_{RE}}$ et $\hat{N_\Delta}$ avec N.

[4] Dans le projet Skitter par exemple, on a reporté que l'ajout d'une nouvelle destination apportait en moyenne 3 nouvelles adresses, à comparer avec les millions d'adresses couvertes par leurs cartes de la topologie `IP`

3.4.1 Graphes

Nous avons choisi de soumettre nos estimateurs à plusieurs types de topologies bien différentes. Trois d'entre elles nous ont semblé représentative de la variété des résultats obtenus :

1. La topologie aléatoire pure, issue du modèle aléatoire simple décrit en Section 1.5 où un nombre prédéfini d'arêtes sont distribuées au hasard parmi les $\binom{N}{2}$ arêtes du graphe, sera désignée par le sigle ER, suivant les initiales d'Erdös et Renyi qui l'ont rendu populaire [33]. Cette topologie produit des graphes fortement homogènes, où les sommets ont tous un degré proche du degré moyen.
2. Les graphes obtenus par le modèle d'attachement préférentiel, désignés par BA selon les initiales de Barabási et Albert qui l'ont rendu populaire [9], sont au contraire très hétérogènes. Leur degré moyen est également spécifiable en paramètre.
3. Le dernier graphe utilisé est issu des mesures de la topologie `IP` du projet Skitter. Les mesures que nous avons considérées ont été effectuées en Mai 2006 depuis 18 sources et vers 445768 destinations[5] à travers le monde.

Pour les deux types de topologies synthétiques, nous avons fixé le degré moyen à $z = 6$ car cela correspondait au degré moyen des cartes de l'Internet dont nous disposions au début de notre travail. D'autres valeurs ont été essayées également, et produisent des résultats qualitativement similaires. Les tailles N utilisées pour la génération de nos graphes BA et ER allaient de 10^3 à 10^6. Le graphe issu des mesures d'Internet par Skitter possédait quant à lui 624324 nœuds et 1191525 arêtes, pour un degré moyen de 3.82.

Les caractéristiques de ces topologies sont volontairement différentes : les graphes BA et ER sont loin de la réalité mais restent utiles pour observer le comportement de nos estimateurs. Leur aspect synthétique et dénué des caractéristiques complexes de la topologie de l'Internet (comme le fort clustering, l'organisation hiérarchique, etc) nous a poussé à utiliser des graphes issus de la réalité. Celui que nous utilisons ici est lui-même un échantillon et ne prétend pas imiter parfaitement l'Internet, mais comporte de nombreuses structures complexes similaires à l'objet initial, et présente donc un intérêt majeur.

3.4.2 Échantillonnage de type `traceroute`

Étant donné un graphe G, pour simuler un échantillonnage comparable à celui du projet Skitter, nous choisissons d'abord les n_S sources $s_1, \cdots, s_{n_S}$ et les n_T destinations $t_1, \cdots, t_{n_T}$ distinctes par tirage aléatoire parmi les N sommets de G. Nous déterminons ensuite l'ensemble des plus courts chemins depuis chaque source vers chaque destination. Trois conventions algorithmiques sont alors envisageables [48] pour modéliser la découverte des chemins entre chaque couple $(s, t) \in S \times T$:

[5] Le nombre total de destinations utilisé par le projet Skitter à ce moment-là était deux fois plus grand, mais nous nous sommes volontairement restreints aux destinations *atteintes* par les `traceroute` pour s'approcher des conditions idéales posées par nos modèles

1. La convention USP pour *Unique Shortest Path* fait en sorte que chaque couple (s,t) se voit attribuer un unique plus court chemin, et ce de manière cohérente à un routage centré sur s : si le plus court chemin de s à t s'écrit $s, a_1, \cdots, a_k, t$, alors pour tout $i \leq k$, le plus court chemin de s à a_i s'écrira $s, a_1, \cdots, a_i$. Pour chaque source, le routage est décrit par un arbre des plus courts chemins choisi aléatoirement parmi tous les arbres possibles.
2. La convention RSP pour *Random Shortest Path* attribue simplement à chaque couple (s,t) un plus court chemin unique, choisi aléatoirement parmi l'ensemble des plus courts chemin entre s et t.
3. La convention ASP pour *All Shortest Paths* attribue à chaque couple l'ensemble des plus courts chemins les reliant : ainsi, un "sondage" entre s et t permet d'observer la topologie de tous les plus courts chemins possibles entre s et t.

Pour plus de réalisme, nous avons opté pour la convention USP, qui s'avère de plus être la pire des trois en termes de couverture, puisqu'elle engendre les échantillons G^* les plus petits. Nous avons également essayé les autres conventions et obtenu des résultats très similaires, et avons donc préféré nous conformer a notre choix originel.

3.4.3 Résultats

Pour comparer les performances de nos estimateurs, nous avons tracé leurs performances respectives pour des paramètres n_S, q_T identiques. Pour faire office de comparaison, nous avons également inclus la performance de l'estimateur trivial $\hat{N} = N^*$, qui se résume à considérer que G^* est une bonne estimation de G. Nous quantifions la performance de ces trois estimateurs N^*, $\hat{N_{RE}}$ et $\hat{N}_\Delta$ par les rapports N^*/N, $\hat{N_{RE}}/N$ et $\hat{N}_\Delta/N$.

En fonction de q_T

Les tracés montrés en Figure 3.4 montrent une comparaison à n_S fixé et q_T variable de 10^{-3} à 1, pour les 3 valeurs $n_S = 1$, $n_S = 10$ et $n_S = 100$, et pour les trois topologies introduites en Section 3.4.1. Notons que le cas $n_S = 1$ est un cas très pessimiste mais permet de tester la performance de nos estimateurs dans des cas extrêmes. Des valeurs de 10 et 100 sont plus réalistes dans le cadre de projets de type Skitter.

Dans les tracés, nous désirons donc obtenir des rapports proches de 1 indiquant que nos estimateurs sont proches de la réalité. Au vu des tracés, il semble clair que l'amélioration apportée par nos estimateurs en comparaison à N^* est significative, même si une bonne estimation de N n'est pas toujours possible. Les estimateurs, quels qu'ils soient, sont d'autant plus fidèles que n_S est grand et que q_T est grand.

Entre la méthode du Delta et le re-échantillonnage, le premier s'en sort beaucoup mieux. Cela est dû à l'imperfection des approximations et à la dilatation des erreurs concernant le re-échantillonnage, comme déjà discuté dans la Section 3.3.1, alors que la méthode du Delta ne souffre *a priori* d'aucun biais, sauf pour les cas extrêmes où G^* est trop petit par rapport à G, ce qui provoque sa grande variabilité pour les très faibles valeurs de q_T ainsi que des valeurs sous-estimées de N dans les cas où n_S et q_T sont très petits.

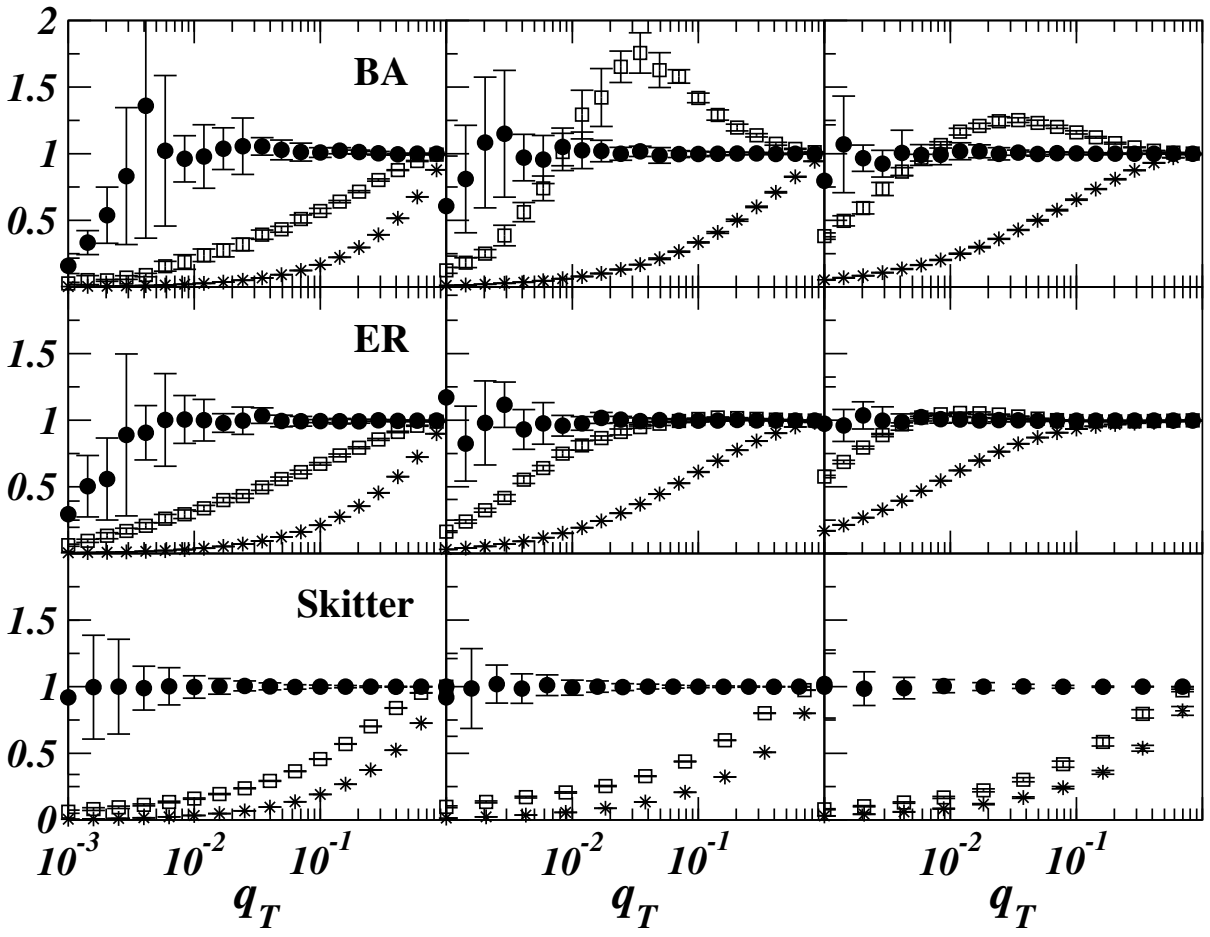

FIG. 3.4 – Comparaisons des performances des trois estimateurs en fonction de q_T, à n_S fixé et pour des graphes fixés. Les ronds noirs correspondent à $\hat{N_\Delta}/N$, les carrés blancs à $\hat{N_{RE}}/N$ et les étoiles à N^*/N. Les topologies utilisées sont, de haut en bas : BA, ER, et Skitter. Le nombre de sources utilisé est, de gauche à droite : $n_S = 1$, $n_S = 10$, $n_S = 100$. Les valeurs ainsi que les barres d'erreurs correspondant à l'écart-type sont basées sur 100 échantillons G^* collectés pour chaque valeur de q_T.

En termes de topologie, et en ce qui concerne le re-échantillonnage, l'estimation semble plus facile pour les graphes ER, et plus difficile pour le graphe Skitter. Pour N^* et $\hat{N_\Delta}$, l'estimation semble relativement stable quel que soit le type de topologie, avec un léger mieux dans le cas ER. La meilleure qualité de $\hat{N_\Delta}$ dans le cas de Skitter s'explique simplement par le fait que ce graphe est de taille bien supérieure au deux autres, et que $\hat{N_\Delta}$ se comporte d'autant mieux que N est grand pour un q_T fixé (voir la fin de la Section 3.3.2). Notons enfin le comportement étrange de $\hat{N_{RE}}$ sur le graphe BA pour des valeurs intermédiaires de q_T. Nous n'avons pas trouvé d'explication bien cernée, mais suspectons fortement la nature très spéciale du graphe BA d'être en cause, n'ayant observe de tels effets sur aucun

autre type de topologie.

En fonction de N

Nous partons ici du principe que les mesures réelles se font sur l'Internet, qui reste un réseau évolutif dont la croissance est significative au cours du temps, alors que les infrastructures de mesures sont fixées ou ont du moins des limitations ne pouvant pas forcément s'adapter à la croissance du réseau. Nous nous intéressons donc à la qualité de nos estimateurs en fonction de N, la taille du réseau, dans deux scénarios :

1. À n_T fixé
2. À q_T fixé

Nous montrons les résultats de ces deux simulations dans les Figures 3.5 et 3.6. Ne disposant pas de cartes `IP` de l'Internet relevées pendant une période de temps suffisamment longue pour traduire une croissance significative du réseau, nous avons restreint nos simulations aux topologies synthétiques ER et BA déjà utilisées, pour lesquelles la taille peut évoluer librement. Nous avons donc gardé constant le degré moyen $z = 6$ pour ces deux topologies.

Dans le premier cas, la qualité des estimations décroît naturellement avec N, puisque la quantité d'information n'augmente pas (ou très peu) alors que la taille du graphe augmente. La nature étrange des résultats dans le cas des graphes BA vient du pic observé précédemment. On remarque cependant que la qualité de nos estimateurs $\hat{N_{RE}}$ et $\hat{N}_\Delta$ décroît bien plus lentement que celle de N^*.

Dans le deuxième cas, nos estimateurs se comportent cette fois d'autant mieux que N est grand, contrairement à N^*/N qui diminue quasi imperceptiblement. Cela est naturellement dû au fait qu'avec une densité de sondage fixée, la représentativité de G^* par rapport à G reste approximativement constante, mais que l'information transportée par G^* devient de plus en plus riche, fait qu'exploitent nos estimateurs.

Application au sondage de l'Internet

Nous avons utilisé nos estimateurs sur des données réelles. Le re-échantillonnage a pu être testé sur la donnée fournie par Skitter dont nous avons déjà parlé, qui est un graphe échantillon de l'Internet relevé avec 18 sources et 445768 destinations. La méthode du Delta n'a pu être testée que sur une mesure plus petite décrite ci-dessous. Les résultats ne prétendent pas encore être des mesures fiables de la taille de l'Internet à cause de l'écart entre théorie (un `traceroute` parfait) et pratique, mais sont fournis pour simple démonstration.

Notons que dans le cas particulier de l'estimation de N dans l'Internet, une expérience bien plus simple que nos estimateurs permet de fournir une réponse assez précise du nombre d'hôtes "vivants" à un instant donné, la définition du terme *vivant* étant alors "qui répond au *ping*". Pour cela, en remarquant que l'espace des $2^{32} \approx 4 \times 10^9$ adresses `IP`v4 possibles est relativement bien peuplé, on peut simplement tirer K adresses `IP` aléatoires dans l'espace d'adressage et compter le nombre K' d'adresses *valides* répondant au *ping* parmi ces K

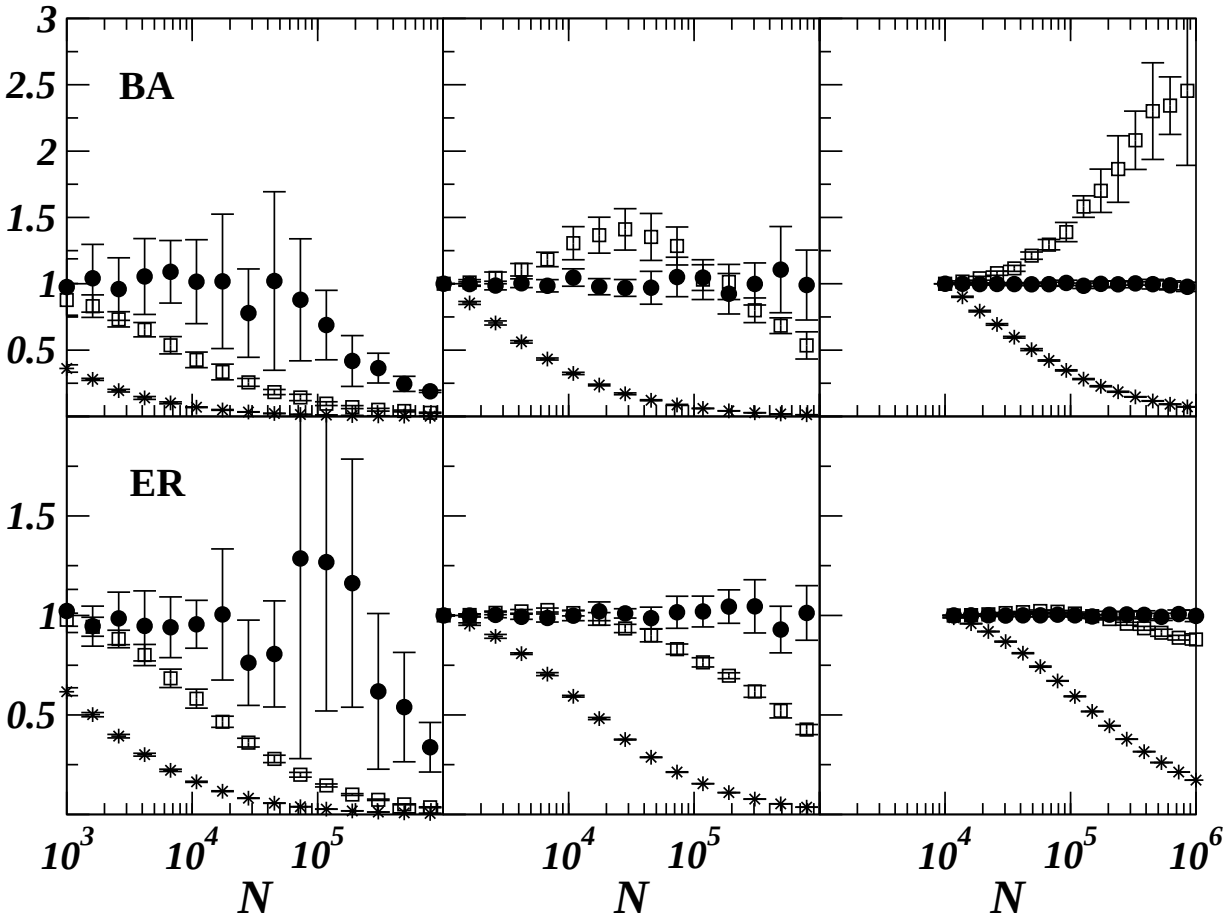

FIG. 3.5 – Effet de la taille N sur la qualité des estimateurs à n_T fixé, pour les graphes BA (en haut) et ER (en bas). Les ronds noirs correspondent à $\hat{N_\Delta}/N$, les carrés blancs à $\hat{N_{RE}}/N$ et les étoiles à N^*/N. Le nombre de sources est fixé à $n_S = 10$, et le nombre de destinations est de $n_T = 10^2$ à gauche, $n_T = 10^3$ au milieu et $n_T = 10^4$ à droite. Les valeurs ainsi que les barres d'erreurs correspondant à l'écart-type sont basées sur 100 échantillons G^* collectés pour chaque valeur de N.

adresses. L'estimation du nombre total d'hôtes répondant au *ping* sur l'Internet au moment de l'expérience est alors immédiatement donnée par $2^{32} * K'/K$. nous avons mené cette expérience afin d'évaluer la qualité de nos estimateurs dans le cas de l'Internet, même si les grandeurs mesurées ne sont pas rigoureusement les mêmes (nos estimateurs auront tendance à estimer le nombre d'hôtes répondant à `traceroute`, ce qui ne revient pas forcément au même que les hôtes répondant au *ping*).

Nous avons donc choisi $K = 3726773$ adresses `IP` aléatoires et envoyé des *ping* vers chacune d'entre elles, recevant $K' = 61246$ réponses valides, soit un taux de réponse de 1.64%. L'estimation de N correspondante est alors de $\hat{N_{ping}} = 70583737$. Nous avons ensuite lancé des `traceroute` vers ces 61246 destinations pour appliquer la méthode du Delta, et

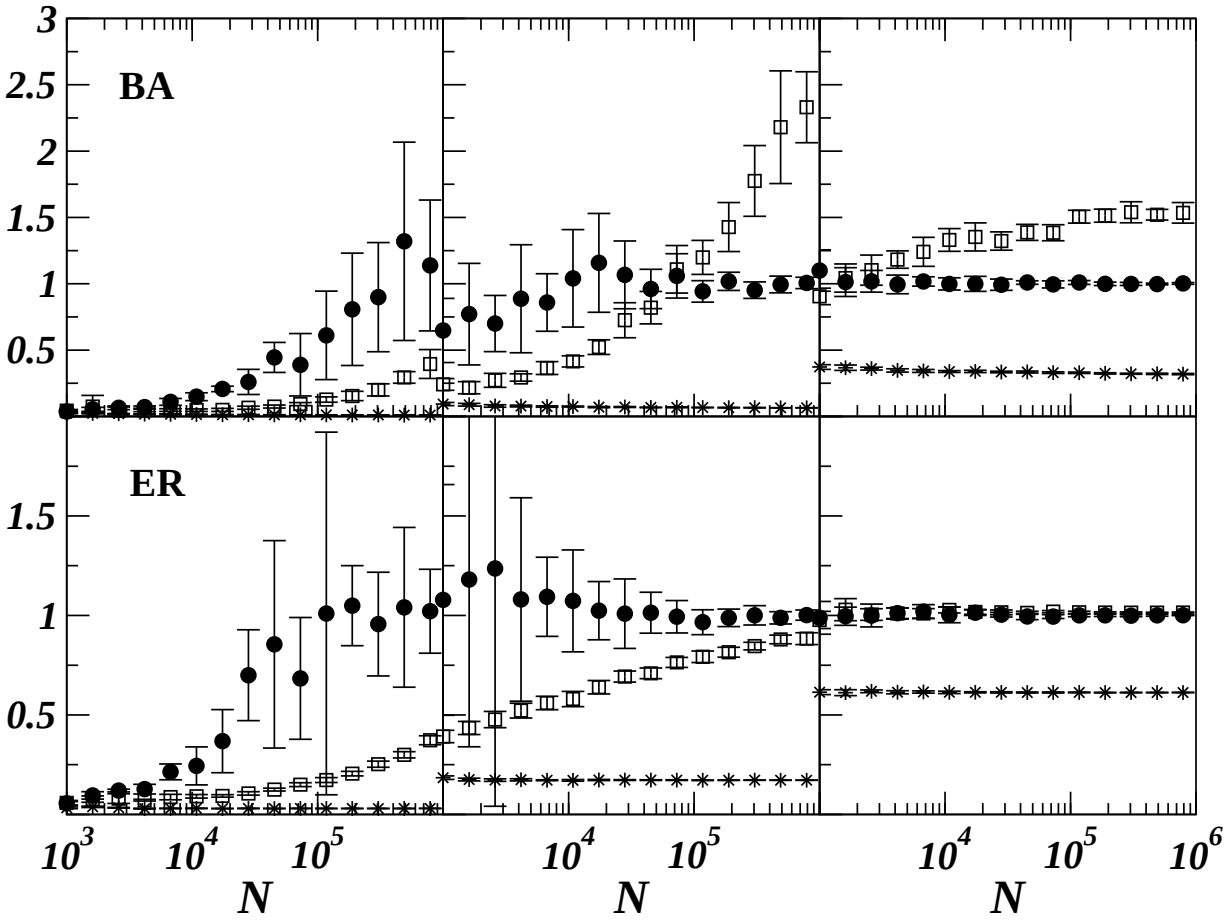

FIG. 3.6 – Effet de la taille N sur la qualité des estimateurs à q_T fixé, pour les graphes BA (en haut) et ER (en bas). Les ronds noirs correspondent à $\hat{N_\Delta}/N$, les carrés blancs à $\hat{N_{RE}}/N$ et les étoiles à N^*/N. Le nombre de sources est fixé à $n_S = 10$, et la densité de destinations est de $q_T = 10^{-3}$ à gauche, $q_T = 10^{-2}$ au milieu et $q_T = 10^{-1}$ à droite. Les valeurs ainsi que les barres d'erreurs correspondant à l'écart-type sont basées sur 100 échantillons G^* collectés pour chaque valeur de N.

obtenu l'estimation $\hat{N_\Delta} = 72296221$. De son coté, l'estimation par re-échantillonnage sur les données de Skitter à donné $\hat{N_{RE}} = 1051295$. Comme on l'attendait, le résultat de la méthode du Delta est proche de celui donné par notre estimation à base de *ping* et représente un nombre sans doute bien plus significatif de la réalité que l'estimation fournie par le re-échantillonnage, qui souffre de défauts déjà discutés. Il faut cependant garder à l'esprit que de meilleures estimations seraient possibles en conservant ce principe mais en l'optimisant pour le cas particulier de l'Internet, notamment en prenant en compte les phénomènes négligés par simplicité, comme discuté en Section 3.1.3. Si l'estimation par *ping* semble fiable, elle dispose néanmoins d'un potentiel très réduit par rapport à nos approches, qui exploitent une plus grande quantité d'informations.

3.5 Discussion et perspectives

Dans ce chapitre, nous avons étudié la correction de la partialité et des biais des mesures de l'Internet basées sur `traceroute`. Notre travail se situe en réalité dans le cadre plus général de toute mesure d'un graphe basée sur la collecte de chemin. Nous perdons ainsi la possibilité de nombreux choix spécifiques à la mesure de l'Internet : par exemple, nous avons ignoré les défauts inhérents à `traceroute`, qui – bien que ce soit là le rôle de cet outil – ne donne pas de mesures rigoureuses des chemins dans le graphe `IP`, à cause des divers problèmes abordés en Section 1.6.2. À cause de cela, nos résultats peuvent paraître contestables, et ne sont pas applicables tels quels dans le cas de l'Internet sans appliquer de sérieuses corrections, comme le montre les chiffres obtenus en Section 3.4.3.

Néanmoins, nos travaux constituent un premier effort dans la direction voulue, qui reste encore peu explorée dans le contexte scientifique actuel.

Nous nous sommes restreints à l'estimation du nombre N de nœuds présents dans le réseau sondé, et avons conçus deux estimateurs de natures assez différentes mais basés tous deux sur un principe de sous-échantillonnage. Nos résultats numériques montrent clairement la faisabilité et l'intérêt d'une telle approche. Comme prévu, nos estimateurs se comportent d'autant mieux qu'on augmente la densité de destinations et le nombre de sources dans le sondage. Il est également intéressant de voir que leur qualité augmente avec la taille du réseau, à nombre de sources et densité de destinations fixés : même si dans ces cas-là le rapport N^*/N tend à diminuer très légèrement, nos estimateurs tirent parti de la plus grande richesse de l'information disponible dans G^* et produisent de meilleurs résultats.

De futurs travaux dans cette direction sont fort souhaitables. De nombreuses directions apparaissent déjà :

- Appliquer nos estimateurs $\hat{N_\Delta}$ et $\hat{N_{RE}}$ à l'Internet de manière *adaptée*. Il s'agit de prendre en compte les phénomènes ayant des impacts forts sur notre mesure et/ou la validité de nos approximations, comme par exemple l'imperfection des mesures `traceroute`. On pourrait par exemple choisir des modèles de routage plus proches de la réalité que le routage par plus courts chemins. L'hypothèse fondamentale du re-échantillonnage souffre elle aussi de déviation semble-t-il plus grave dans le cas de l'Internet que dans les topologies synthétiques ou mêmes les cartes de l'Internet dont nous disposons. Une piste prometteuse pour corriger cela serait de différencier les nœuds extrémaux du graphe de ceux du cœur (les routeurs), et donc de concevoir une variante du re-échantillonnage permettant de se libérer de ces contraintes.
- Améliorer la technique utilisée dans le re-échantillonnage pour détecter le point d'égalité $q_T{}^* = q_T$, trop approximative et coupable de larges déviations. L'équation fondamentale 3.7 mérite aussi d'être remplacée par une version plus précise. Ces améliorations sont à faire dans le cadre spécifique d'une mesure de l'Internet : le schéma global du re-échantillonnage peut être conservée, mais des optimisations importantes sont envisageables selon les contexte d'application.
- Adapter la technique de re-échantillonnage à la mesure d'autres quantités, comme M ou les degrés $(deg(i))_{i \in V}$.

- Chercher des moyens d'adapter la méthode du Delta à la mesure d'autres quantités. Contrairement au re-échantillonnage, cette méthode a été conçue spécifiquement pour N et n'est pas facile à généraliser.

Nous présentons d'ores et déjà un avant-goût de l'estimation de M avec le re-échantillonnage en Figure 3.7. Les résultats présentés sont le fruit d'une adaptation sommaire à l'estimation de M en remplaçant la condition sous-jacente à l'équation de re-échantillonnage 3.7 par le couple $(n_S{}^* = n_S, n_T{}^*/M^* = n_T/M)$. Une adaptation plus avisée serait sans doute nécessaire, puisque les résultats pour l'estimation de M semblent moins bons que pour l'estimation de N. Les ressources nous ont manqué dans cette direction, mais nous sommes convaincus que des travaux futurs pourront y remédier.

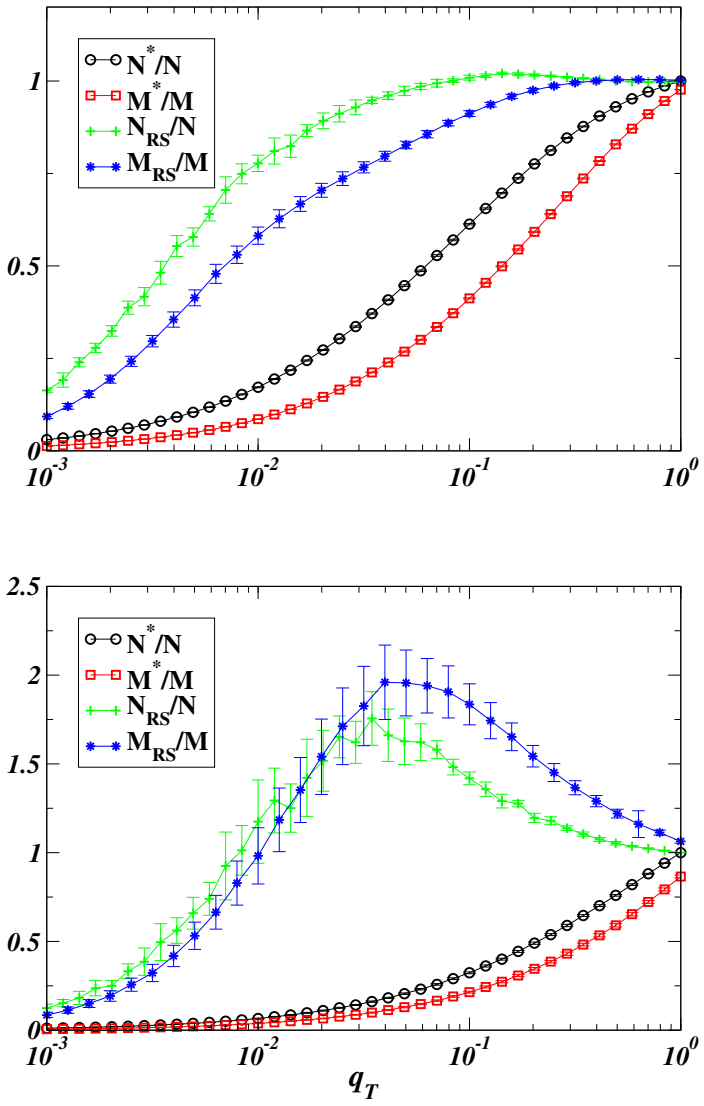

FIG. 3.7 – Performance comparée des estimateurs basés sur le re-échantillonnage pour les graphes ER (en haut) et BA (en bas), avec dans les deux cas $N = 10^5$ et $z = 6$, en utilisant $n_S = 10$ sources.

Chapitre 4

Correction de l'outil traceroute

Nous avons vus précédemment comment l'agrégation de mesures `traceroute` permettait de relever des cartes de la topologie `IP`, et parlé des imprécisions et de la partialité de ces mesures dues à diverses raisons (voir la Section 1.6.2 et le Chapitre 3). Si certaines de ces raisons sont connues et bien documentées, la multitude d'effets pervers et d'anomalies dans les traces expérimentales reste peu documentée : il est difficile de trouver une étude quantitative des erreurs de mesure dans les cartes de la topologie `IP`, même sur celles disponibles publiquement. Ce chapitre présente un effort de classification des anomalies visant à clarifier la situation.

Par rapport au contexte général de mesures à grande échelle, nous nous restreignons dans ce chapitre aux mesures depuis une *source unique*. L'essentiel de notre propos sera même centré sur les mesures individuelles : nous nous intéressons aux erreurs susceptibles d'apparaître lors de l'appel d'un `traceroute` isolé, comme le ferait un utilisateur désirant sonder le chemin vers une seule destination. Nous l'étendrons parfois au cas de mesures répétitives, ou au cas de mesures vers un ensemble de destinations.

Ce chapitre est organisé de la manière suivante. Nous commençons par détailler les spécificités techniques de `traceroute` qui seront utilisées dans la suite, puis expliquons comment la répartition de charge – qui est la première cause d'erreur que nous ayons identifiée – affecte la validité de `traceroute` en tant que mesure de la topologie `IP`. Nous proposons alors un moyen technique simple de contourner la répartition de charge, et d'éliminer ainsi une bonne partie des erreurs de mesure. Cette correction prend la forme d'un nouvel outil, *Paris traceroute*, projet open-source disponible sur [135]. Nous identifions ensuite un certain nombre de structures anormales (artefacts) dans les cartes de la topologie `IP` : les *boucles*, *cycles* et *diamants*. Nous montrons comment ces structures apparaissent à cause de la répartition de charge, et quantifions leur disparition lors de l'utilisation de notre outil. Celle-ci n'étant pas totale, nous enquêtons ensuite sur les autres phénomènes susceptibles de provoquer l'apparition de nos artefacts, et identifions les principaux d'entre eux. Nous fournissons également des moyens de détecter la présence de tels phénomènes et donc de corriger la mesure rétrospectivement. Au final, l'utilisation combinée de notre outil *Paris traceroute* et des diagnostics présentés dans ce chapitre permet d'éliminer la grande majorité des artefacts étudiés ici.

Notons que d'autres erreurs de mesures, comme discuté en Section 1.6.3, apparaissent uniquement lors d'études mettant en jeu des mesures `traceroute` depuis de *multiples* sources. Ce n'est pas l'objet ici.

4.1 Vers un `traceroute` amélioré

4.1.1 L'outil `traceroute`

Le fonctionnement de l'outil `traceroute` a déjà été résumé en Section 1.6.2 (voir notamment la Figure 1.7). Nous ajoutons ici quelques détails. Le terme "`traceroute`" lui-même peut être sujet à ambiguïté, puisque de nombreuses versions existent. Leur comportement est en fait très similaire, et le choix de versions différentes de celles que nous avons utilisées ne change absolument rien dans nos travaux – nous avons d'ailleurs utilisé plusieurs versions différentes sans problème particulier. Par souci de rigueur, notre description de l'outil `traceroute` restera générique, et conforme à la description donnée par Jacobson [58], comme en Section 1.6.2. Nous restreignons notamment le champ des paramètres possibles aux seuls qui nous intéressent.

Fonctionnement général

L'invocation d'un `traceroute` se fait depuis une source donnée (un ordinateur connecté à l'Internet) et nécessite trois paramètres : l'adresse `IP` destination, le protocole P utilisé, et le nombre de sondes à envoyer à chaque étape. Le protocole est UDP (par défaut) ou ICMP, ou encore TCP si l'outil `traceroute` concerné le permet. Le sondage par `traceroute` se fait en partant de la source vers la destination par succession d'étapes. Les étapes sont numérotées à partir de 1 en augmentant de 1 à chaque fois. Cet entier correspond au TTL que l'on donne aux sondes, et donne donc la distance du routeur sondé (en nombre de routeurs traversés depuis la source, routeur sondé inclus) pendant cette étape. Chaque étape est en général constituée de l'envoi d'un certain nombre de sondes (le plus souvent 3 sondes par défaut). Le routeur sondé, en voyant passer une sonde dont le TTL sera tombé à 1 pendant le transfert, supprime la sonde et envoie un message ICMP *Time Exceeded* à la source [88].

L'adresse source de cette réponse permettra d'identifier le routeur puisque celui-ci est censé utiliser l'adresse `IP` d'une de ses interfaces, précisément celle qui envoie le message ICMP [8, Sec. 4.3.2.4] [1]. Quand le routage est symétrique (localement, au niveau du routeur sondé), cette adresse sera typiquement celle de l'interface qui a reçu la sonde. On peut donc comprendre que la collecte de `traceroute` issus de plusieurs sources, comme discuté en Section 1.6.2, présente l'inconvénient d'obtenir des adresses `IP` différentes bien que correspondant au même routeur. Dans ce chapitre, nous ignorons ce problème puisque nous réduisons notre champ de recherche aux mesures effectuées depuis une seule source.

[1] Pour plus de détails, voir [70] et sa bibliographie.

Au final, la sortie de `traceroute` peut typiquement ressembler à l'exemple montré en Figure 4.1 : il s'agit d'une suite d'adresses `IP` annotée du numéro d'étape (à gauche) et du temps écoulé entre l'envoi de la sonde et la réception de la réponse. On peut voir les temps d'aller-retour des trois sondes envoyées. Dans certaines étapes, plusieurs adresses `IP` sont présentes, ce qui indique que des adresses `IP` différentes ont répondu aux trois sondes.

```
traceroute to www.l.google.com (66.249.93.147), 30 hops max, 40 byte packets
 1  129.199.1.10    10.428 ms  2.350 ms   1.198 ms
 2  195.221.127.61  3.198 ms   2.116 ms   2.246 ms
 3  195.221.125.57  3.144 ms   2.243 ms   9.423 ms
 4  193.51.186.102  1.529 ms   1.838 ms   1.493 ms
 5  193.51.187.18   7.284 ms   2.426 ms   1.823 ms
 6  194.68.129.242  2.468 ms   2.332 ms   1.842 ms
 7  80.231.79.14    2.136 ms   1.909 ms   1.994 ms
 8  80.231.65.65    18.934 ms  11.683 ms  11.550 ms
 9  80.231.66.4     11.044 ms  11.613 ms  11.241 ms
10  80.231.66.50    11.042 ms  11.550 ms  11.018 ms
11  66.249.94.138   11.008 ms  11.108 ms  11.455 ms
12  72.14.232.208   15.595 ms  72.14.238.119 16.683 ms  17.352 ms
13  64.233.175.246  27.463 ms  18.798 ms  72.14.232.141  17.835 ms
14  72.14.233.79    24.550 ms  72.14.233.77  17.618 ms  17.557 ms
15  216.239.47.229  18.844 ms  66.249.94.46  29.517 ms  216.239.47.229  23.950 ms
16  66.249.93.147   19.956 ms  17.960 ms  26.694 ms
```

FIG. 4.1 – Exemple de sortie obtenue par le programme `traceroute`.

Identification des réponses ICMP

L'outil `traceroute` nécessite un moyen d'identifier de manière unique les sondes qu'il envoie, car une même machine est susceptible de lancer plusieurs `traceroute` de manière simultanée vers une même destination, dans des processus différents. Il est également important de pouvoir identifier *au sein d'un même* `traceroute` le numéro d'étape des sondes, celles-ci pouvant éventuellement être envoyées sans attendre la réception des réponses des étapes précédentes. Sur l'outil `traceroute` classique, cette identification se fait en examinant les données transportées dans le paquet ICMP *Time Exceeded*, qui contient les 28 premiers octets du paquet sonde tel qu'il était lorsque le routeur l'a reçu. L'en-tête `IP`, long de 20 octets, est donc contenu dans ces 28 octets, ainsi que les 8 premiers octets qui suivent l'en-tête [89, p.5]. Si l'en-tête `IP` est sans options, comme c'est le cas pour les sondes `traceroute`, ces 8 octets contiennent – selon le protocole utilisé par `traceroute` : UDP, ICMP ou TCP – l'intégralité de l'en-tête UDP, l'intégralité de l'en-tête ICMP ou une partie de l'en-tête TCP. Si cette portion du paquet sonde comprend un identifiant unique, `traceroute` peut déterminer à quelle sonde correspond chaque réponse ICMP *Time Exceeded* reçue.

Pour le protocole UDP, le comportement par défaut de `traceroute` consiste à choisir un port source correspondant à l'*identifiant du processus* invoquant `traceroute` (PID) plus 32768, ce qui permet d'associer chaque sonde au processus `traceroute` qui l'a envoyée, et à choisir un port destination égal au départ à 33435 et augmentant de 1 à chaque étape, ce qui permet d'associer chaque sonde au numéro d'étape correspondant. Pour les sondes ICMP, ces deux champs sont remplacés par le champ d'identifiant (*Identifier*) et le champ de numéro de séquence (*Sequence Number*). Dans les deux cas, chaque champ occupe 4 octets, et les deux champs sont juxtaposés dans les 8 premiers octets suivant l'en-tête `IP`.

Pour diverses raisons (comme des routeurs éliminant silencieusement des paquets ayant un TTL de 1 sans émettre de paquet ICMP *Time Exceeded*), certaines sondes peuvent ne pas engendrer de réponses. Le comportement par défaut consiste alors à considérer l'échec de la sonde après un temps d'attente limite, et à afficher une étoile en lieu et place de l'adresse `IP` attendue.

4.1.2 Impact de la répartition de charge sur la mesure

Les administrateurs réseau emploient la répartition de charge pour augmenter la fiabilité et équilibrer l'utilisation des ressources. Cela consiste essentiellement à répartir des flux de données sur plusieurs routes : on ne peut alors parler de *la* route entre une source et une destination mais d'*une* des routes *possibles*. Une organisation possible est montrée en Figure 4.2.

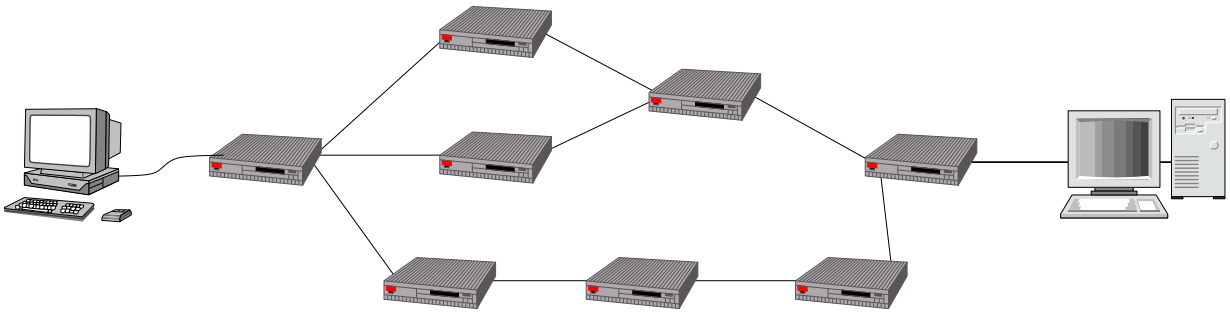

FIG. 4.2 – Répartition de charge sur l'Internet : existence de plusieurs routes entre un même couple (source, destination).

En pratique, la répartition de charge se fait essentiellement au niveau intra-domaine (au sein d'un AS), à travers les protocoles OSPF [79] et IS-IS [15] qui appliquent le principe de répartition selon des chemins de même coût (*equal cost multipath* ou *ECMP*). Un AS peut également répartir la charge entre ses AS voisins [90].

Les différents modes de répartition de charge

Au niveau des routeurs, la répartition de charge peut se faire selon trois modes principaux : par paquet, par flux ou par destination [17, 60].

- Dans la répartition de charge *par flux*, le routeur peut envoyer des données provenant d'une même source et allant vers la même destination sur des chemins différents, mais tente de laisser les paquets appartenant à un même flux de données sur la même route. Un flux correspond aux données échangées par un même programme, ou une même transaction. En pratique, le routeur examine l'en-tête des paquets `IP` pour y trouver des informations (voir Figure 4.3) lui permettant d'associer à chaque paquet un numéro de flux. Il routera ensuite les paquets ayant le même numéro de flux par la même route. Cela évite notamment de désordonner les paquets appartenant à un même flux.
- Avec la répartition de charge *par paquet*, le routeur ne se soucie pas de l'appartenance des paquets à un flux, et se concentre sur sa tâche de répartir la charge. Il peut par exemple envoyer cycliquement les paquets sur les différentes routes possibles – ce qui a alors l'avantage de garantir une parfaite répartition même en cas de flux disproportionné.
- Enfin, la répartition de charge *par destination* peut se voir comme une version extrême de la répartition de charge par flux, où les paquets sont envoyés vers telle ou telle route uniquement selon leur destination. Il n'y a pas vraiment de notion de flux, même si chaque flux est sûr d'être routé de manière unique puisqu'il correspond à un unique couple (source, destination). Il peut paraître étrange de parler de répartition de charge dans ce cas, puisque le routage `IP` en général consiste justement à envoyer les paquets vers différentes routes en fonction de leurs destinations : nous nous restreignons en fait au cas où deux paquets ayant des destinations X et Y distinctes et passant par un routeur A répartissant la charge sont envoyés sur deux routes différentes alors qu'ils repassent tous deux plus tard par un autre routeur B avant d'arriver à destination.

Le choix du mode de répartition de charge pour les routeurs dépend du fabricant, de la version de l'OS, et de la manière dont l'AS le configure. Par exemple, les routeurs Cisco peuvent être configurés pour répartir la charge par paquets ou par destination [17], alors que les routeurs Juniper donnent aux administrateurs réseaux une certaine flexibilité dans le choix des champs utilisés pour définir la répartition de charge par flux [60].

Champs utilisés pour l'identification de flux

La répartition de charge par flux se fait en calculant un *identifiant de flux* pour chaque paquet à l'aide de son en-tête. Il est couramment admis que les champs de l'en-tête utilisés pour construire cet identifiant sont au nombre de cinq, d'où l'appellation *five-tuple* (quin-tuplet), et sont alors l'adresse source, l'adresse destination, le protocole, le port source et le port destination. Les trois premiers se trouvent dans l'en-tête `IP` alors que les deux derniers se trouvent dans l'en-tête TCP ou UDP. Nous avons lancé de nombreux tests sur l'Internet pour vérifier la validité de cette assertion et déterminer quels champs étaient utilisés par les routeurs utilisant la répartition de charge par flux pour choisir la route prise par les paquets. Nous avons utilisé des sondes TCP, UDP, ICMP, et IPSec. Ces tests ont montré que les routeurs utilisent effectivement différentes combinaisons de ces cinq champs, plus trois autres : le champ `IP` *ToS* (Type of Service) et les champs ICMP *Code* et *Checksum*.

La manière de combiner ces 8 champs dépend ensuite du routeur considéré.

Nous n'avons pas réussi à obtenir de réponses des routeurs en utilisant des sondes ICMP autres que ICMP *Echo*, et ne pouvons donc conclure avec certitude sur l'utilisation du champ *ICMP Type* pour l'identification de flux, celui-ci étant toujours fixé à 1 dans nos expériences. Nous résumons en Figure 4.3 la constitution des en-têtes IP, UDP, ICMP *Echo*, et TCP où les champs utilisés pour l'identification de flux ont été grisés. Nous n'avons pas étudié la dépendance exacte de l'identification de flux en fonction de ces champs, et laissons cela à des travaux futurs.

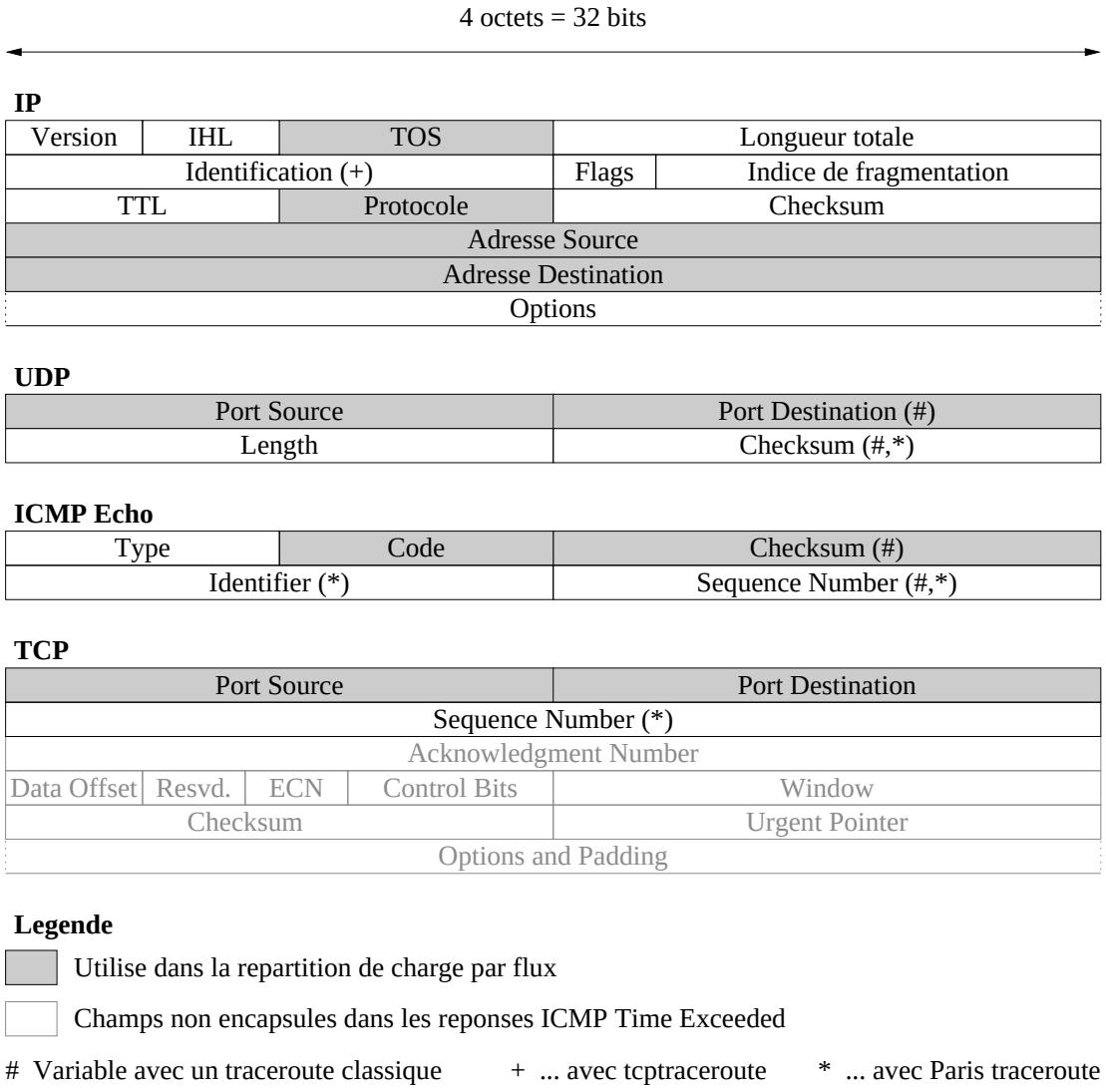

FIG. 4.3 – Constitution des en-têtes IP, UDP, ICMP *Echo* et TCP. Les champs pouvant être utilisés pour la répartition de charge par flux sont grisés.

`traceroute` et la répartition de charge par flux

À cause de la conception de `traceroute`, les sondes qu'il envoie apparaissant systématiquement comme appartenant à des flux différents. Cela vient de la manipulation du contenu des 28 premiers octets des sondes pour les identifier de manière unique, comme nous l'avons décrit en Section 4.1.1 : pour les sondes UDP , `traceroute` fait systématiquement varier le port destination, et pour les sondes ICMP il fait varier le champ *Sequence Number*. Même si ce dernier champ n'est pas directement impliqué dans l'identification de flux (voir Figure 4.3), il modifie le champ ICMP *Checksum* qui participe à cette identification. Les sondes envoyées par `traceroute`, qu'elles soient UDP ou ICMP, auront donc de grandes chances d'emprunter des routes différentes si de la répartition de charge par flux a lieu sur le chemin. La notion de route comme suite de routeurs empruntés par un paquet ou par un flux ne correspond donc pas forcément à la mesure donnée par `traceroute` qui se fait avec des paquets différents. Le `traceroute` classique souffre donc de profonds défauts en termes de mesure de topologie, comme nous l'illustrons dans la Figure 4.4. En nous appuyant sur l'exemple montré dans cette Figure, nous discutons ces défauts plus en détails.

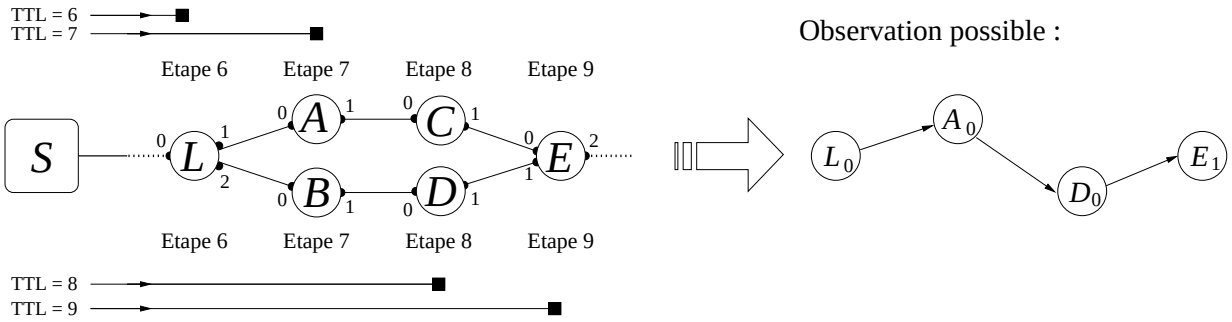

FIG. 4.4 – Liens et nœuds manquants ou fictifs dans une mesure de la topologie basée sur `traceroute` en présence de répartition de charge par flux. Dans ce cas, L est un répartiteur de chargesitué à une distance 6 de la source S. À gauche, nous voyons la topologie réelle entre les distances 6 et 9. Les nœuds ronds sont les routeurs, dont chaque interface est numérotée. Les carrés noirs représentent les paquets sondes envoyés avec des TTLs allant de 6 à 9. Il sont montrés soit en haut si L les dirige sur le routeur A soit en bas s'il les dirige sur le routeur B. À droite, nous montrons la topologie telle qu'on peut la déduire de la mesure `traceroute` décrite ici.

Nœuds et liens manquants

À cause du fait qu'on ne reçoive aucune réponse des routeurs B et C, ceux-ci ne sont pas découverts, et les liens (L_0, B_0) et (B_0, D_0) non plus. Dans le cas général, et même en envoyant plusieurs sondes par étape, la probabilité de manquer des routeurs augmente avec le nombre de routeurs situés à une distance donnée : si par exemple on a 3 routeurs à une même distance et en supposant qu'une sonde peut passer par ces 3 routeurs de manière équiprobable, la probabilité de manquer au moins un des 3 routeurs en envoyant 4 sondes

est supérieure à 0.62. Notons que la présence de routeurs invisibles implique celle de liens invisibles. Notons également que le nombre de routeurs à une certaine distance peut être potentiellement élevé : les derniers routeurs Juniper, par exemple, autorisent jusqu'à 16 possibilités pour la répartition de charge.

Faux liens
Indépendamment du fait que tous les nœuds ou liens de la topologie soient découverts, `traceroute` aura tendance à donner de fausses informations topologiques en présence de répartition de charge par flux. Dans notre exemple de la Figure 4.4, L envoie la sonde de TTL initial 7 vers A et celle de TTL initial 8 vers B, amenant au report d'un lien fictif entre A_0 et D_0. Dans notre exemple, nous avons une chance sur deux de reporter un faux lien, puisque les sondes de TTL 7 et 8 ont une chance sur deux d'être routées sur des chemins différents. Lors de mesures réelles, entre les étapes 7 et 8 chacun des liens (A_0, C_0), (A_0, D_0), (B_0, C_0), (B_0, D_0) peut être reporté avec la même probabilité, rendant impossible la différentiation entre liens réels et fictifs. Cette fois encore, la présence d'un nombre plus élevé de routeurs à une distance donnée complique le problème et augmente la probabilité de mesurer de faux liens.

Le problème des faux liens a été reconnu et étudié dans certains cas, en particulier par Huffaker et al. [55] et Spring et al. [99]. Cependant, aucune étude systématique de ce phénomène n'a été réalisée pour le moment.

4.1.3 Un nouveau `traceroute`

Nous introduisons à présent ***Paris traceroute***[2], un nouvel outil `traceroute` conçu pour la mesure en présence de répartition de charge. L'innovation clé est de contrôler l'entête des paquets sondes de manière à garder un identifiant de flux unique pour toutes les sondes lancées vers une même destination, ce qui fera qu'elles suivront le même chemin même en présence de répartition de charge par flux. Notre outil permet également de différencier, grâce aux mesures, la répartition de charge par flux et par paquets. À cause de la nature hasardeuse de cette dernière, notre outil s'avère parfois incapable d'énumérer parfaitement toutes les routes existantes, mais est considérablement plus robuste que `traceroute` et permet également d'isoler les mesures douteuses – qui sont présentes, mais en forte minorité.

Identifier les sondes sans changer le flux

La difficulté provient de la nécessité d'identifier les sondes de manière unique sans toucher aux champs utilisés par la répartition de charge par flux. Il faut encapsuler un **identifiant de processus** dans les sondes pour que chaque processus puisse reconnaître ses propres sondes, et un **identifiant d'étape** pour qu'un processus puisse trier les réponses qu'il reçoit, qui peuvent arriver dans un ordre différent de l'ordre d'émission. Comme le

[2]Disponible sur http://www.paris-traceroute.net/

montre la Figure 4.3, l'espace disponible dans les 28 octets de la sonde qui nous sont renvoyés est faible, beaucoup de champ étant également fixés et non modifiables (IP version, Checksum, ...) sous peine de suppression pure et simple par certains routeurs détectant les paquets mal formés. D'autres champs, telles l'adresse source, le TTL ou l'adresse destination ne peuvent être modifiés sans compromettre le fonctionnement de `traceroute`. De plus, il n'est pas souhaitable non plus d'encoder de telles informations dans les options `IP`, puisque les paquets ayants des options `IP` sont souvent traités par les routeurs de manière spéciale : ils ne sont alors pas transmis rapidement comme le reste des paquets mais sont interprétés par un circuit spécifique du routeur [43], avec des règles potentiellement différentes des paquets normaux dont `traceroute` est censé mesurer la route.

Intéressons-nous d'abord à l'**identifiant d'étape** du paquet. Cette valeur évoluera au cours d'un même `traceroute` et ne doit en aucun cas faire varier l'identifiant de flux. *Paris traceroute* utilise les septième et huitième octets de l'en-tête UDP, ICMP ou TCP pour cela, qui sont les deux derniers octets des 28 octets disponibles. Notons au passage que *Paris traceroute* permet d'envoyer des sondes TCP, contrairement au `traceroute` classique mais comme le tcptraceroute de Toren [106].

- Pour les sondes UDP, ces octets correspondent à la somme de contrôle (Checksum) et ne peuvent donc être modifiés arbitrairement sans risquer une suppression potentielle du paquet qui serait considéré comme corrompu. Pour donner à ce champ la valeur voulue, *Paris traceroute* manipule les données envoyées dans le reste de la sonde pour obtenir la somme de contrôle correspondante.
- Pour les sondes ICMP *Echo*, *Paris traceroute* utilise le champ *Sequence Number*, comme le `traceroute` classique. Cependant, pour éviter le même problème, *Paris traceroute* modifie simultanément le champ ICMP *Identifier* de manière antisymétrique pour maintenir un champ *Checksum* constant.
- Pour les sondes TCP, *Paris traceroute* utilise le champ *Sequence Number* (au lieu du champ `IP` *Identification* utilisé par tcptraceroute). Cette fois, aucune manipulation n'est nécessaire pour maintenir constants les champs critiques.

Pour encoder l'identifiant de processus, nous utilisons le champ `IP` *Identification* pour les trois types de sondes TCP, UDP et ICMP. Le `traceroute` classique utilisait le port source pour les sondes UDP et le champ *Identifier* pour les sondes ICMP. Notons que tcptraceroute n'utilisait pas d'identifiant de processus.

Le fait de garder constant les champs utilisés par la répartition de charge par flux n'est pas nouveau, puisque tcptraceroute appliquait déjà ce principe avec ses sondes TCP où le port destination était par défaut fixé à 80 pour émuler du trafic Web et passer à travers les pare-feu plus facilement. Les conséquences de ce choix particulier n'ont cependant pas été étudiées précisément. Notre outil généralise cette méthode aux protocoles UDP et ICMP tout en permettant l'identification des sondes par étape et par processus.

Autres fonctionnalités

En plus de fixer l'identifiant de flux des sondes envoyées, *Paris traceroute* fournit dans sa sortie davantage d'informations que le `traceroute` classique (même si un certain nombre de ces informations sont disponibles en option, selon les versions). Ces informations permettent, comme on le verra par la suite, de fournir un diagnostic plus détaillé aidant à identifier certaines anomalies.

- Le *TTL de la sonde* est le TTL qu'avait la sonde en arrivant au routeur qui nous a répondu. Nous récoltons cette information dans l'en-tête `IP` de la sonde lui-même inclus dans les 28 octets de données transportés par la réponse ICMP *Time Exceeded*. Cette valeur doit normalement valoir 1, puisque c'est la valeur à laquelle les routeurs suppriment un paquet. Une autre valeur révèle une erreur, ou un mécanisme de routage exotique.
- Le *TTL de réponse* est le TTL du paquet ICMP *Time Exceeded* de réponse du routeur à son arrivée à la source. Cette valeur, également disponible en option avec `traceroute` classique, permet d'évaluer la longueur du chemin retour, puisque les TTL fixés par les routeurs ont le plus souvent des valeurs bien définies à l'exécution comme 64, 128 ou 255. Ces valeurs usuelles sont suffisamment différentes les unes des autres pour extrapoler le TTL initial d'un paquet reçu avec un taux d'erreur faible : si le TTL à l'arrivée vaut 47 par exemple, on supposera que le TTL initial de la sonde valait 64 et que l'émetteur se situe donc à une distance de 17 routeurs.
- L'*identifiant IP* est le contenu du champ `IP` *Identification* de l'en-tête `IP` du paquet de réponse. Comme nous le verrons par la suite, ce champ est souvent mis à une valeur correspondant à un compteur spécifique au routeur, et permettant donc d'identifier le routeur ayant répondu par un autre moyen que l'adresse `IP` source de la réponse ICMP *Time Exceeded*. Cela peut notamment révéler qu'un même routeur se cache derrières plusieurs interfaces (et donc adresses `IP`) [99], ou au contraire qu'une même adresse `IP` correspond en fait à une passerelle ou pare-feu et que les réponses proviennent de routeurs différents [13]. Voir la Section 4.5 pour plus de détails.

Paris traceroute dispose également de plusieurs stratégies pour sonder la route.

- La stratégie *PacketByPacket* se comporte comme le `traceroute` classique : à chaque étape, on envoie les sondes une par une en attendant le retour d'une sonde (ou l'expiration d'un délai d'attente maximum) avant de passer à la sonde suivante.
- La stratégie *HopByHop* envoie toutes les sondes pour une étape donnée directement, avec un délai configurable (la valeur par défaut étant 50 ms), attend toutes les réponses ou une expiration de délai d'attente, puis passe à l'étape suivante. HopByHop est d'autant plus rapide par rapport à PacketByPacket qu'on envoie plus de sondes par étape.
- La stratégie *Concurrent* envoie toutes les sondes, pour toutes les étapes, d'un seul coup et avec un délai entre sondes configurable et fixé à 50ms par défaut. Cette méthode est beaucoup plus rapide que les deux précédentes mais nécessite la connaissance *par avance* du nombre d'étapes à effectuer.
- La stratégie *Scout* commence par envoyer une sonde avec un TTL élevé pour être sûr

d'atteindre la destination. En cas de réponse de celle-ci on estime le nombre d'étapes nécessaire (grâce au TTL de la sonde récupéré dans la réponse ICMP) et on utilise la stratégie Concurrent. Sinon, on peut au choix échouer et ne rien sonder ou utiliser la méthode HopByHop pour sonder la route. Scout est similaire au *raceroute* proposé par Moors [78].

Comme l'outil `traceroute` classique, *Paris traceroute* permet à l'utilisateur de spécifier un TTL minimum et maximum afin de restreindre le sondage à une partie de la route, et ainsi d'accélérer les mesures.

4.2 Mesures expérimentales

Avant d'étudier dans le reste du chapitre les améliorations apportées par notre outil et d'approfondir le sujet des erreurs de mesures, nous décrivons dans cette section le protocole expérimental utilisé pour effectuer des mesures `traceroute` ou *Paris traceroute*. Nous décrivons également sommairement quelques statistiques de base.

4.2.1 Protocole expérimental

Notre source est située au LIP6, laboratoire de l'Université Pierre et Marie Curie à Paris. Cette université n'a qu'une seule liaison à l'Internet grand public qui se fait par le réseau académique national RENATER. Même si notre propos dans ce chapitre reste essentiellement centré sur les mesures collectés par un ou plusieurs `traceroute`(s) vers *une seule* destination, nous ne pouvons pas nous restreindre à sonder une seule destination dans nos mesures par souci de représentativité. L'utilisation d'une seule source biaise déjà nos mesures, mais nous pensons que ce biais est beaucoup moins conséquent que celui qu'engendrerait le choix d'une seule destination. Nous avons donc engendré une liste de 5000 adresses IPv4 aléatoire en tirant aléatoirement des adresses dans l'espace d'adressage `IP` et en gardant celles qui répondaient au *ping* à la création de notre liste. Cette dernière clause, en accord avec les travaux de Xia et al. [132], nous permet de nous situer dans un cadre plus réaliste par rapport à l'utilisateur moyen de `traceroute`, qui lance ce dernier vers des destinations connues plus souvent que vers des adresses non attribuées.

Pendant une période de 74 jours entre Juin et Août 2006, nous avons effectué 1465 passes, chaque passe consistant en l'envoi d'un *Paris traceroute* suivi immédiatement d'un `traceroute` classique vers chacune des destinations de notre liste. Ces données sont celles que nous utiliserons dans ce chapitre. Elles nous permettront d'identifier, d'expliquer et de corriger des artefacts de mesure. Elles n'ont pas vocation à être statistiquement *représentatives* des `traceroute` effectués sur l'Internet en général, mais devraient suffire de par leur taille à donner un aperçu de la diversité des mesures et des anomalies observées. Nous avons également procédé à d'autres mesures et obtenu des résultats similaires. Par souci de cohérence et de simplicité, nous avons donc restreint les chiffres et observations donnés dans ce chapitre à cette mesure, qui est la plus complète.

Chaque passe est effectuée en lançant 32 processus en parallèle sur notre machine

source, chacun sondant 1/32 de la liste de destinations. Chaque processus lance, pour chaque destination d dont il a la charge, deux mesures successives vers d : d'abord un *Paris traceroute*, puis un `traceroute` classique (celui inclus dans la distribution 1.4a5 de NetBSD). Il passe ensuite à la destination suivante. Pour *Paris traceroute* comme pour `traceroute` classique, nous avons envoyé une seule sonde UDP par étape. Les multiples réponses possibles à chaque étape ont donc été collectées au cours du temps en répétant les passes plutôt qu'en envoyant directement plusieurs sondes par étape. La stratégie retenue pour *Paris traceroute* est PacketByPacket, pour se conformer au `traceroute` classique. Le délai d'attente maximum d'une réponse a été fixé à 2 secondes. La seule différence technique entre les mesures du `traceroute` classique et de *Paris traceroute* est que ce dernier fait varier les champs de ses sondes différemment, comme expliqué dans le chapitre précédent. Les valeurs de port source et destination – non utilisées par *Paris traceroute* – sont fixées arbitrairement dans l'intervalle [10,000, 60,000]. En ce qui concerne le `traceroute` classique, le comportement par défaut a été conservé. Dans les deux cas, le sondage vers une destination donnée s'arrêtait quand la destination répondait ou quand un message ICMP différent de *Time Exceeded* était reçu (indiquant une probable erreur de routage). Nous avons également spécifié des conditions d'arrêts supplémentaires : *Paris traceroute* s'arrête au plus tard après l'étape 36, ou après la huitième étoile (non-réponse) rencontrée d'affilée. Le `traceroute` classique lancé juste après doit alors s'arrêter au plus tard trois étapes après la dernière étape ayant engendré une réponse dans le *Paris traceroute* précédent. Cette règle – qui peut sembler étrange – vient à la fois d'une volonté d'accélérer la mesure en évitant les attentes inutiles, et du fait que le `traceroute` classique ne permet pas d'incorporer des conditions d'arrêts sophistiquées et se borne à admettre un nombre d'étapes limite prédéfini. Enfin, nous avons sondé à partir de la deuxième étape (TTL=2) pour éviter le sondage répétitif de la passerelle de notre Université, qui est le premier routeur traversé par les sondes.

Une passe prenait en moyenne 75 minutes, à la vitesse moyenne d'une destination toutes les 27.3 secondes (pour la double exécution d'un *Paris traceroute* **et** d'un `traceroute` classique). Par souci de clarté nous avons listé ci-dessous les points principaux à retenir.

- Nous avons sondé 5000 destinations aléatoires et répondant au *ping*
- Chaque destination est sondée avec *Paris traceroute* et juste après avec le `traceroute` classique. Une fois que les 5000 destinations sont sondées, on recommence.
- La cadence que nous avons obtenue est d'une passe toutes les 75 minutes
- La durée totale de la mesure est de 74 jours, soit 1465 passes.

4.2.2 Quelques statistiques

Pour les 241 millions réponses reçues contenant une adresse `IP` source valide, nous avons recherché le numéro d'AS correspondant en utilisant les techniques de Mao et al. [69]. Nous avons constaté la couverture de 1498 AS, ce qui correspond à 6% des AS recensés au jour de la mesure. De plus, nos mesures couvrent les neuf AS principaux constituant le cœur de l'Internet (appelés *tier-1*) et 75 des 100 AS constituant le top-20 de chacune des 5 régions

du monde selon le rapport hebdomadaire de l'APNIC du moment [3]

Les étoiles (non-réponses) apparaissent essentiellement à la fin des traces : quand une destination ne répond pas, notre mesure continue longtemps (jusqu'à 8 essais successifs pour *Paris traceroute*). Seules 7.7 millions d'étoiles apparaissent au milieu des routes (à comparer avec la centaine de millions d'étoiles détectées en fin de route et les 241 millions de sondes envoyées). Nous avons également recensé les adresses `IP` sources invalides, c'est-à-dire non attribuables selon les règles édictées par l'IANA [56]. Nous avons trouvé 11 adresses `IP` invalides engendrant un total de 57000 réponses invalides, soit 0.024% du total. Aucune d'entre elles n'apparaît dans les structures étudiées plus loin, et ne vient brouiller nos statistiques.

4.2.3 Définitions et Notations

Nous définissons une *trace* comme étant la sortie d'un unique `traceroute` classique ou d'un *Paris traceroute* dans le cadre de nos mesures, et donc avec une sonde par étape. Formellement, une trace est un ℓ-uplet $\mathbf{r} = (r_0, \cdots, r_\ell)$ où r_0 est l'adresse `IP` de la source [4], et pour tout i, $1 \leqslant i \leqslant \ell, r_i$ désigne l'adresse `IP` collectée lors de la réception de la sonde à l'étape i ou une étoile si aucune réponse n'a été reçue. L'entier ℓ est appelé la *longueur* de la trace. Nous appelons tout $(k+1)$-uplet de la forme $(r_i, r_{i+1}, \ldots, r_{i+k})$ une *sous-route* de $\mathbf{r}$ de longueur k.

4.3 Structures particulières de la topologie

Pour analyser finement les conséquences de la répartition de charge en termes d'erreurs sur la topologie `IP`, nous introduisons dans ce chapitre trois structures particulières que nous définissons en termes de routes. La Section 4.3.1 introduit les boucles, la Section 4.3.2 les cycles, et la Section 4.3.3 les diamants. Pour ces trois structures, nous expliquons comment procéder à leur détection et donnons des mécanismes potentiellement responsables de leur formation, ainsi que des statistiques de base sur leur présence dans les traces `IP` classiques – en d'autres termes, la probabilité d'observer de telles structures en effectuant un ou plusieurs `traceroute` classiques.

4.3.1 Boucles

Définition

Dans certaines traces, la même adresse `IP` apparaît deux fois (ou plus) à la suite : nous appelons ce phénomène une *boucle* car l'interprétation naïve de cet évènement – en

[3] L'APNIC génère automatiquement des rapports décrivant l'état des tables de routage sur l'Internet. Il classe les AS pour chaque région du monde selon le nombre de réseaux pairs annoncés.

[4] Nous avons dit plus haut que les routes étaient tracées à partir de l'étape 2. Cela ne gêne pas notre définition des traces, nous fixons juste r_0 et r_1 en tant qu'adresse de la source et de la passerelle de l'Université, respectivement. La sortie correspond donc en réalité à la suite $(r_2, \cdots, r_\ell)$

supposant que `traceroute` montre réellement la route suivie par les paquets – est que les paquets passent deux fois de suite (ou plus) par le même routeur. Cet interprétation est cependant très peu plausible, un routeur ne pouvant pas faire *transiter* un paquet d'une interface à elle-même (du moins pas selon les standards [8]). Les boucles sont donc très probablement un artefact de la mesure.

Formellement, on dira qu'une boucle est observée sur l'adresse `IP` r_i vers la destination d si dans nos données il existe au moins une trace vers d contenant la sous-route $\ldots, r_i, r_i, \ldots$. Le terme "adresse `IP`" utilisé ici implique que r_i n'est pas une étoile : on ne considère pas une suite de deux étoiles comme une boucle. Une telle observation dans nos données sera désignée comme une *instance* de boucle. La *signature* de la boucle sera définie par le couple (r_i, d).

Par exemple, supposons que nous disposons des 4 traces suivantes : $(\ldots, a, a, b \ldots, d_1)$, $(\ldots, a, a, b \ldots, d_2)$, $(\ldots, a, a, b, \ldots, d_2)$ et $(\ldots, a, a, c, \ldots, d_2)$. Nous avons alors quatre instances de boucles (une pour chaque trace), et deux signatures de boucles (a, d_1) et (a, d_2). Selon le contexte, les statistiques feront référence aux instances ou aux signatures de boucles. Cependant, quand le contexte est suffisamment clair ou quand une telle distinction n'a pas lieu d'être, nous utiliserons simplement le terme "boucle". Ajoutons enfin qu'une trace peut contenir plusieurs instances de boucles, mais que dans ce cas on ne compte qu'une seule instance par signature et par trace. Ainsi, une trace vers d comme $\ldots, a, a, \ldots, b, b, \ldots, a, a, \ldots, d$ contient une instance de la boucle (b, d) et une instance de la boucle (a, d).

Les boucles longues font partie des boucles, nous les appelons n-boucles. Nous disons qu'une trace $\mathbf{r}$ contient une instance de n-boucle, avec $n \geq 1$, sur l'adresse `IP` r quand r apparaît exactement $n+1$ fois consécutives sur $\mathbf{r}$ (n est le nombre de fois où r est observé à deux étapes consécutives). Nous appelons n la *longueur* d'instance de la boucle. Les boucles les plus courtes, qui sont les plus fréquentes, sont de longueur 1 et correspondent à l'apparition consécutive de 2 adresses identiques. Nous étendons également cette définition aux signatures de boucles : la longueur d'une signature de boucle sera la longueur maximale sur l'ensemble de ses instances.

Nous utilisons le terme "boucle" en référence à la théorie des graphes, où une boucle désigne un lien liant un nœud à lui-même. C'est bien sûr différent d'une boucle de routage consistant en un renvoi répétitif d'un paquet entre plusieurs interfaces, et impliquant donc plusieurs adresses `IP`. Les boucles de routage sont en fait désignés par des *cycles* dans ce chapitre, et sont discutées en Section 4.3.2.

En utilisant les définitions ci-dessus, l'énumération des boucles dans nos données est immédiate : nous scannons simplement toutes les traces et cherchons des instances d'adresses `IP` identiques consécutives. Afin de calculer nos statistiques, nous maintenons une liste de toutes les signatures de boucles rencontrées, et pour chaque signature (r, d) nous gardons un certain nombre d'informations, telles que la longueur maximale, le nombre d'instances, le nombre de fois que l'adresse `IP` r a été observée sur une trace vers d (que cette observation soit liée à une boucle ou non), etc.

Statistiques

Les boucles sont étonnamment fréquentes : sur nos données, et malgré leur durée relativement courte, 7.51% des adresses `IP` relevées par nos mesures ont été au moins une fois impliquées dans une instance de boucle, et plus de 4.35% des traces contiennent une boucle – ce dernier chiffre correspond en fait à la probabilité d'observer une boucle lors de l'exécution d'un `traceroute`, du moins depuis notre source. De plus, en sondant répétitivement, des boucles peuvent apparaître sur des routes déjà sondées et n'ayant pas donnée lieu à des boucles auparavant. En prenant en compte l'ensemble des 1465 passes de nos mesures, des boucles ont été observées au moins une fois vers 24.6% des destinations. La différence entre ce chiffre et les 4.35% de traces concernées par des boucles est due à la forte hétérogénéité des boucles : alors que certaines semblent *persistantes* et apparaissent quasiment à chaque trace vers la destination concernée, de nombreuses boucles sont *occasionnelles* et n'apparaissent qu'une seule fois (ou un petit nombre de fois) sur les 1465 passes. Nous observons également des boucles au comportement intermédiaire : les boucles *systématiques*. Nous disons qu'une boucle (r, d) est systématique quand l'apparition de r sur une trace vers d implique presque sûrement l'apparition de la boucle $\ldots, r, r, \ldots$ dans la trace. La différence avec les boucles persistantes est que l'apparition de r peut être en fait très sporadique.

Pour enquêter plus précisément sur ces distinctions, nous définissons deux grandeurs statistiques pour chaque signature de boucle (r, d) :

1. sa *probabilité d'apparition* est la probabilité qu'une trace vers d contienne une instance de (r, d), et
2. sa *probabilité d'apparition conditionnelle* est la probabilité qu'une trace vers d *contenant* r contienne une instance de (r, d).

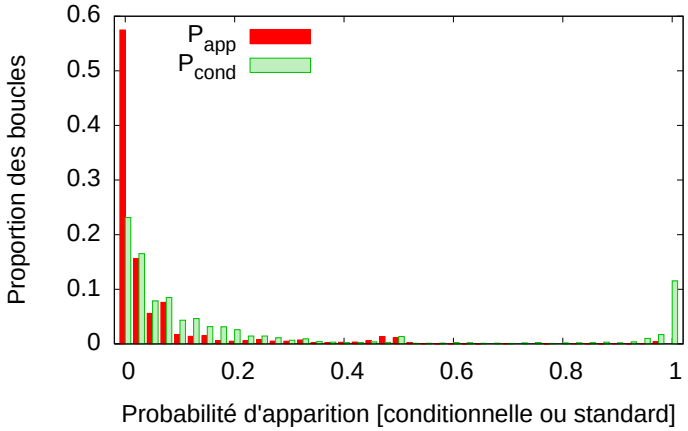

FIG. 4.5 – En rouge foncé : répartition de la probabilité d'apparition des boucles. En vert clair, répartition des probabilités d'apparition conditionnelle.

La Figure 4.5 montre la distribution de ces deux caractéristiques sur l'ensemble des

signatures de boucles observées. Notons que la proportion des boucles persistantes (probabilité d'apparition proche de 1) semble très faible, bien inférieure à la proportion de boucles systématiques (ayant une probabilité d'apparition conditionnelle proche de 1). Cela n'est vrai qu'en termes de nombre de signatures. Ce petit nombre de boucles persistantes engendre un volume considérable d'instances, puisqu'elles sont justement observées sur presque toutes les traces : on leur doit 7.45% des instances de boucles dans nos données.

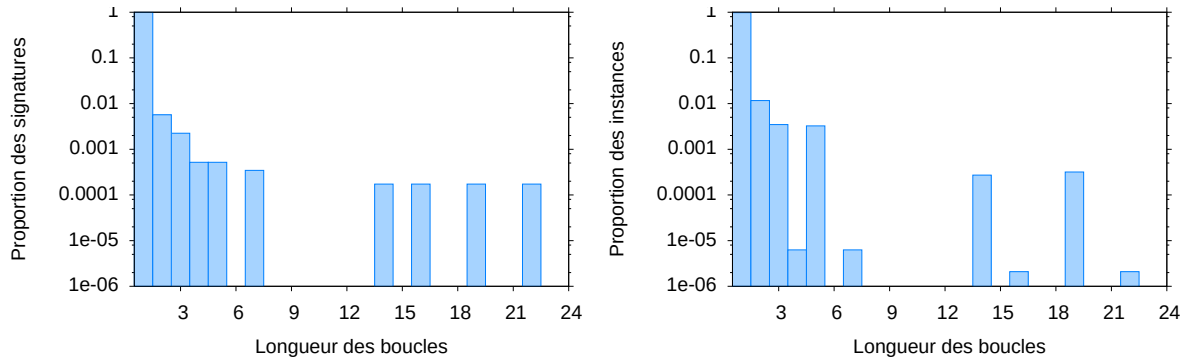

FIG. 4.6 – Distribution des longueurs des boucles en termes de signatures (à gauche) et d'instance (à droite).

La Figure 4.6 montre la répartition des longueurs observées parmi les n-boucles, en termes de signatures (à gauche) et d'instances (à droite). L'échelle est logarithmique à cause de l'hégémonie des boucles de taille 1 et de la grande rareté des longues boucles. Toutefois, notons que nous avons pu observer quelques n-boucles persistantes de grande taille, qui ne sauraient donc être considérées comme des évènements exceptionnels. Notez également que la première barre n'atteint pas vraiment 1 : c'est une fausse impression due à l'échelle logarithmique.

Les statistiques d'apparition et de longueur semblent toutes deux confirmer que plusieurs espèces de boucles existent, et que les phénomènes à l'origine des boucles ont probablement des natures différentes. Cela nous incite à enquêter sur chaque type de boucle séparément pour isoler leurs causes en prenant en compte le comportement observé. En pratique, cette approche – l'examen minutieux des statistiques pour élaborer des diagnostics appropriés – nous a beaucoup aidé, comme le montrent les résultats de la Section 4.5.

4.3.2 Cycles

Définition

Formellement, une trace est dite *cyclique* sur l'adresse `IP` r, ou r-cyclique, si elle contient r au moins deux fois à des étapes séparées par au moins une adresse `IP` r' valide et distincte de r. Nous sous-entendons ici que ni r ni r' ne peuvent être des étoiles. Cette définition assure que nous ne mésinterprétons pas une n-boucle potentielle en cycle. Par exemple, la

trace $(\ldots b, c, d, e, c, f, \ldots)$ est cyclique en c, mais la trace $(\ldots b, c, *, *, c, f, \ldots)$ ne l'est pas car la sous-route $(c, *, *, c)$ pourrait être une 3-boucle.

Comme pour les boucles, nous utilisons le terme *instance de cycle* pour toute occurrence d'un cycle sur une trace, et définissons la *signature d'un cycle* comme étant le couple (r, d) de l'adresse `IP` répétée r et de la destination concernée d. Toujours comme pour les boucles, on ne compte qu'une seule instance de cycle par signature et par trace. La plupart des cycles que nous ayons observés apparaissent à plusieurs endroits d'une même trace et se chevauchent même les uns les autres sur les traces, rendant difficile d'établir une énumération rigoureuse et de définir proprement certaines grandeurs, comme une éventuelle longueur de cycle. Afin de fournir des statistiques, nous avons donc considéré deux grandeurs permettant d'évaluer la "longueur" d'un cycle :

- la *période* d'un cycle, associée à une signature (r, d), est la plus petite distance séparant deux apparitions non consécutives de r dans une trace r-cyclique vers d, et
- l'*étendue* d'un cycle, associée à une signature (r, d), est la plus grande distance séparant deux apparitions de r dans une trace r-cyclique vers d.

Ces deux grandeurs capturent les plus petite et plus grande longueurs d'un parcours circulaire des sondes de r à r sur l'ensemble des traces vers la destination d.

Une catégorie intéressante de cycle est les *cycles périodiques*, qui sont observés lors d'une trace telle que $(\ldots, b, c, b, c, \ldots)$. Une trace est dite *$k$-périodiquement cyclique* sur les adresses `IP` $r_1, r_2, \cdots, r_k$ quand $k \geq 2$ et quand cette séquence d'adresses est rencontrée au moins deux fois de suite et dans le même ordre sur la trace. Plus formellement, nous définissons la signature d'un cycle périodique (qui nous permettra de les énumérer) comme un couple $((r_1, \cdots, r_k), d)$ tel qu'au moins une des traces vers d soit k-périodiquement cyclique sur $r_1, \cdots, r_k$.

Comme pour les boucles, l'énumération des cycles se fait simplement en scannant chaque trace, en détectant les instances de cycles, et en agrégeant les informations sur chaque signature de cycle rencontrée.

Statistiques

Les cycles sont moins fréquents que les boucles dans nos données : il n'apparaissent que sur 0.60% des traces. Par contre, on peut les observer dans le long terme vers plus de destinations : 17.5% d'entre elles ont fini par engendrer au moins un cycle au cours des 1465 passes, et 4.73% des adresses `IP` relevées dans nos mesures sont impliquées dans au moins une signature de cycle.

La Figure 4.7 présente les statistiques d'apparition similaires à celles introduite pour les boucles, en utilisant des termes identiques. Pour éviter les répétitions, nous invitons le lecteur à consulter, si ce n'est pas déjà fait, la Section 4.3.1 avant de continuer dans celle-ci.

Les cycles persistants semblent très rares, et même invisibles sur la Fig. 4.7. Nous avons observé 2 cycles persistants sur les 5,674 signatures de cycle observées, qui engendrent tout de même 3.93% des instances de cycles. Les cycles systématiques sont plus communs (voir le pic dans la Figure 4.7 pour une probabilité d'apparition conditionnelle de 1), et représentent 8.95% des instances des cycles. Ils gardent une probabilité d'apparition très faible malgré

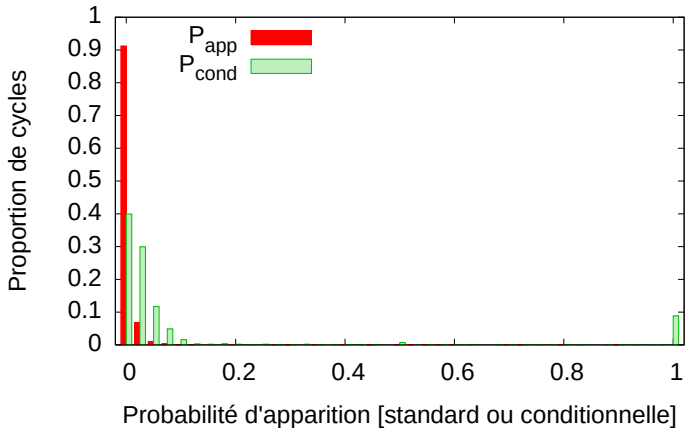

FIG. 4.7 – En rouge foncé : répartition de la probabilité d'apparition des cycles. En vert clair, répartition des probabilités d'apparition conditionnelle.

ce comportement systématique. Dans l'ensemble, on peut considérer les cycles comme des évènements essentiellement occasionnels, ce qui les rend plus difficile à traquer.

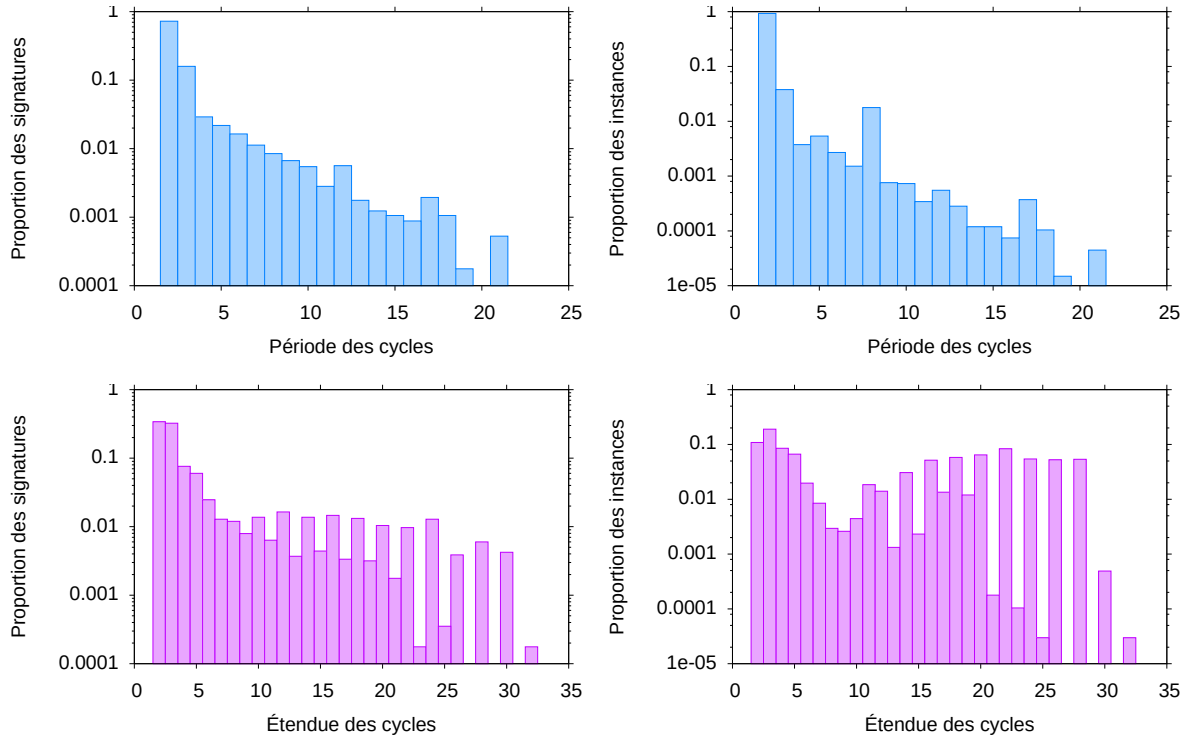

FIG. 4.8 – Distribution de la période (en haut) et de l'étendue (en bas) des cycles en termes de signatures (à gauche) et d'instances (à droite).

La Figure 4.8 montre la distribution des périodes et étendues des cycles observés. Comme pour les boucles, nous utilisons des statistiques basées sur le nombre de signatures ou le nombre d'instances. Ces statistiques illustrent la variété dans la nature des cycles observés. En observant la distribution des périodes des cycles, il paraît clair que les cycles de période 2 sont largement prédominants. Cependant, les cycles ayant une *étendue* égale à 2 ne sont pas aussi courants, et sont même dépassés en nombre d'instances par les cycles d'étendue 3. Cela indique bien sûr que tous les cycles n'ont pas une étendue égale à leur période. Cela est également lié à l'observation (Figure 4.8, en bas à droite) que les grandes étendues *paires* sont beaucoup plus représentées en termes d'instances que les grandes étendues impaires : ces phénomènes peuvent être expliqués par les cycles périodiques introduits plus hauts. Ces cycles périodiques peuvent avoir par exemple une période de 2 mais une étendue supérieure à 2 pouvant aller jusqu'au nombre maximal d'étapes autorisé dans nos mesures, et restant forcément paire. Notons également la présence de cycles ayant une très grande période, dont la proportion semble décroître continûment en fonction de la période, surtout en termes de signatures.

Tout cela suggère encore une fois l'existence de phénomènes variés provoquant la mesure de cycles de natures très différentes.

4.3.3 Diamants

Les boucles et cycles sont observés sur des traces uniques. D'autres types de structures apparaissent seulement quand on agrège les mesures obtenues par plusieurs traces Une structure typique est ce que nous appelons un *diamant*. La Figure 4.9, qui représente une partie d'un graphe collecté par agrégation de traces, montre un premier exemple graphique de ce que nous entendons par diamants (et éclairera le lecteur quant aux choix de ce terme).

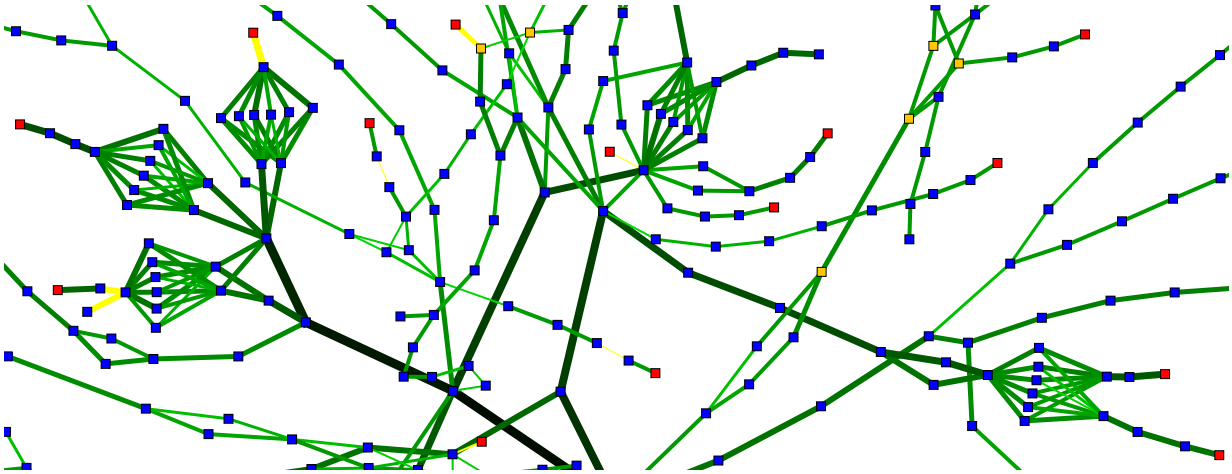

FIG. 4.9 – Exemples de diamants dans nos données.

Définition

Étant donné un ensemble S de traces, un *diamant* est défini par un couple (r, t) d'adresses `IP` tel que l'ensemble $\{r_i\}$ des adresses `IP` présentes dans des traces de la forme $\ldots, h, r_i, t, \ldots$ soit de cardinal supérieur à 2. En d'autres termes, il s'agit d'un couple d'adresses `IP` présentes dans des traces à 2 étapes de distance tels que l'adresse `IP` située entre elles ait plusieurs valeurs différentes selon les traces considérées. Comme d'habitude, le terme "adresse `IP`" implique ici qu'aucun des termes h, t et r_i ne sont des étoiles. Nous appelons alors h la *tête* du diamant, et t sa *queue*. Le *corps* du diamant est l'ensemble $\{r_1, \ldots, r_k\}$ des adresses `IP` vues entre h et t, et la *taille* du diamant est l'entier k.

Cette définition est très générique, et contrairement aux signatures de boucles et de cycles elle n'implique pas de destination particulière. Nous allons différencier les diamants selon l'ensemble de traces utilisé. Nous verrons en Section 4.4.3 qu'une telle distinction permet des observations intéressantes liées à la répartition de charge. Les définitions ci-dessous sont complexes et requièrent un certain temps avant de se familiariser avec les termes utilisés. Nous encourageons donc fortement le lecteur à se reporter autant que possible à la Figure 4.10.

Diamant global Un diamant (h, t) apparaissant en considérant l'ensemble des traces de notre mesure, toutes destinations confondues, est appelé un *diamant global*. Le corps d'un diamant global (h, t) correspond à la définition classique : c'est l'ensemble des adresses apparaissant entre h et t sur au moins une trace. La taille d'un diamant global est simplement le cardinal de cet ensemble.

d-diamant Un diamant (h, t) apparaissant dans l'ensemble restreint des traces vers une destination spécifiques d est particulier, en ce qu'il peut être observé *a priori* sur un seul `traceroute` vers d si celui-ci utilise suffisamment de sondes par étapes. Nous appelons un tel diamant un *d-diamant*, car il apparaît en sondant uniquement la route vers d. Les caractéristiques d'un d-diamant (corps, taille) sont naturellement définies comme avant, en gardant à l'esprit que l'ensemble des traces considéré est l'ensemble des traces vers d.

Diamant local Les d-diamants sont faciles à concevoir, mais difficiles à utiliser pour fournir des statistiques globales puisqu'on dispose de beaucoup de possibilités pour agréger les données provenant de d-diamants ayant le même couple (h, t). Afin de remédier à cela, nous définissons les diamants locaux, sortes d'agrégats statistiques des d-diamants pour un couple (h, t) donné. Formellement, dès lors qu'il existe une destination d pour laquelle (h, t) est un d-diamant, nous appelons (h, t) un *diamant local*. Le *corps* d'un diamant local (h, t) est l'union de tous les corps des d-diamants (h, t), et sa *taille* est définie comme étant le maximum des tailles de tous les d-diamants (h, t) : notons qu'elle ne correspond cette fois pas au cardinal du corps. Naturellement, tous les diamants locaux sont forcément des diamants globaux.

En pratique, les diamants sont recensés dans nos données de la manière suivante. Nous scannons toutes les traces et maintenons, pour chaque triplet (h, t, d) de deux adresses `IP`

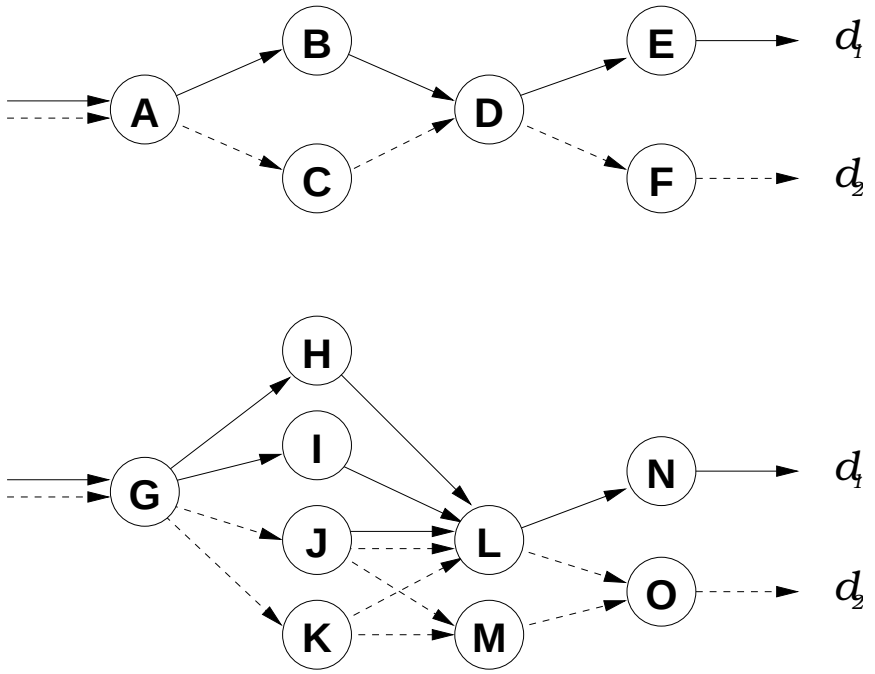

FIG. 4.10 – Exemples de diamants globaux, d-diamants, et diamants locaux. Dans cet exemple, deux traces sont présentes, l'une vers d_1 et l'autre vers d_2. Nous avons omis de numéroter les différentes interfaces de chaque routeur pour plus de simplicité, considérant que l'adresse `IP` d'un routeur est celle de l'interface par laquelle les sondes arrivent. $A, \ldots, O$ sont donc les adresses observées sur ces traces, et une flèche solide (resp. pointillée) entre deux adresses indique qu'elles apparaissent consécutivement dans la trace vers d_1 (resp. d_2). (G, L) est un d_1-diamant de taille 3, et un d_2-diamant de taille 2. Par conséquent, (G, L) est un diamant local de taille 3, qui est la plus grande taille des d-diamants (G, L). De plus, (G, M), (J, O) et (K, O) sont tous des d_2-diamants de taille 2 et donc également des diamants locaux de taille 2. Par contre, (A, D) n'est un d-diamant pour aucune destination d, et n'est donc pas un diamant local. Enfin, (A, D), (G, L), (G, M), (J, O) et (K, O) sont des diamants globaux.

et d'une destination, l'ensemble $A_d(h, t)$ de toutes les adresses `IP` vues entre h et t sur les traces vers d. Nous calculons ensuite pour chaque couple (h, t) l'ensemble $A_{all}(h, t)$ des adresses `IP` vues entre h et t sur *l'ensemble* des traces en fusionnant les $A_d(h, t)$ pour tous les d. Ces ensembles constituent notre base de travail.

Les d-diamants sont simplement les triplets (h, t, d) tels que $|A_d(h, t)| \geqslant 2$. Les diamants locaux sont les couples (h, t) tels qu'il existe un d vérifiant $|A_d(h, t)| \geqslant 2$, et les diamants globaux sont les couples (h, t) tels que $|A_{all}(h, t)| \geqslant 2$. Les corps et tailles de ces diamants se déduisent des définitions ci-dessus et se calculent facilement à partir des $A_d(h, t)$ et des $A_{all}(h, t)$.

Notons enfin que contrairement aux boucles et aux cycles, les termes "instance" et

"signature" n'ont pas de sens pour les diamants, ceux-ci étant définis sur l'*agrégation* des traces.

Statistiques

Nous observons 28231 diamants globaux dans l'ensemble de nos mesures. La Figure 4.11 présente la distribution de leur taille.

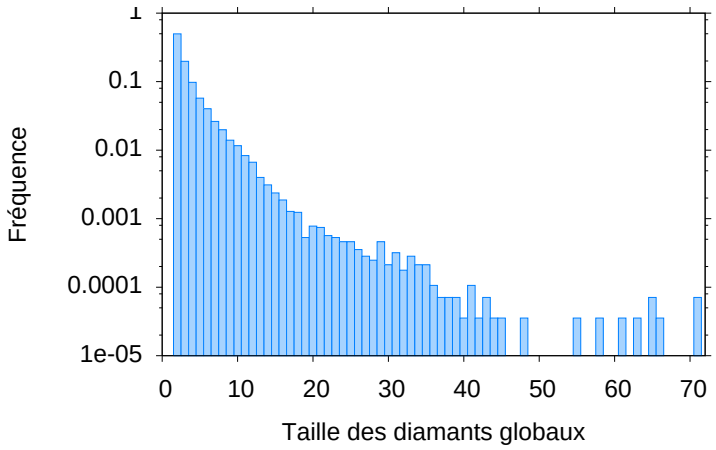

FIG. 4.11 – Distribution de la taille des diamants globaux.

Les diamants concernent une très grande partie de l'ensemble des adresses `IP` collectées lors de nos mesures : 44.6% d'entre elles sont impliquées dans au moins un diamant global. Ajoutons que les diamants globaux ne sont pas – dans nos mesures – des entités disjointes et sont au contraire hautement imbriqués les uns dans les autres, comme on pouvait déjà le voir sur la Figure 4.9. En effet, considérant l'ensemble des adresses `IP` relevées par la mesure, 16.4% d'entre elles se trouvent dans la tête d'au moins un diamant global, 30.7% se trouvent dans le corps et 27.1% dans la queue. La somme de ces trois quantités excédant de loin 44.6%, il est clair que certaines adresses `IP` se retrouvent à jouer plusieurs rôles dans plusieurs diamants globaux différents mais non disjoints.

Les diamants locaux et d-diamants sont aussi relativement fréquents dans nos mesures : nous relevons 95936 d-diamants formant 24326 diamants locaux. Parmi l'ensemble des 5000 destinations utilisées, 90.2% ont conduit à l'observation d'au moins un d-diamant. Parmi ces destinations, le nombre moyen de d-diamants différents observés sur la même destination est de 21.3. Ces derniers chiffres montrent clairement la présence des diamants même au niveau de la simple trace, à condition que celle-ci se répète dans le temps ou soit effectuée avec plusieurs sondes par étape.

La Figure 4.12 présente la distribution des tailles des d-diamants et des diamants locaux. En la comparant avec la Figure 4.11, nous observons que les *très grands* d-diamants sont beaucoup moins nombreux que les très grands diamants globaux. Cette observation, combinée au fait qu'il y a plus de diamants globaux que de diamants locaux, indique que des traces vers différentes destinations peuvent se combiner et créer des diamants globaux

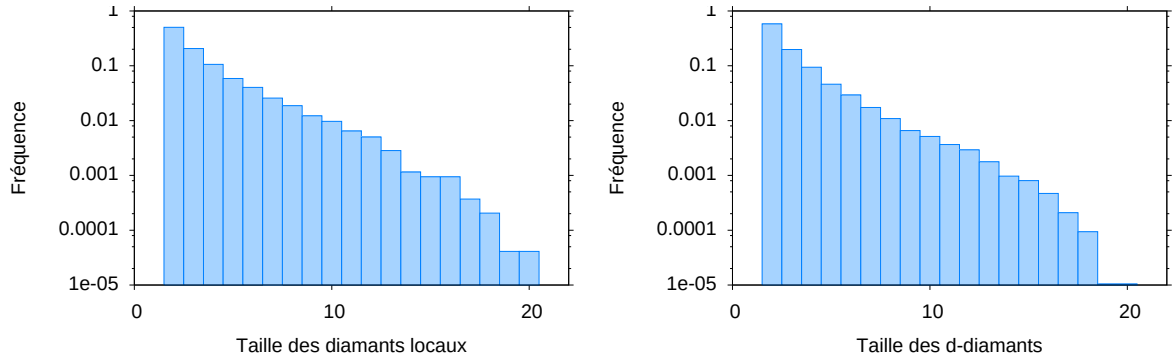

FIG. 4.12 – Distribution de la taille des diamants locaux (à gauche) et des d-diamants (à droite).

sans présence de diamants locaux, comme nous le schématisons en Figure 4.10 pour le diamant (A, D).

4.3.4 Récapitulatif

Nous présentons dans la Table 4.1 un récapitulatif des principales statistiques de présence des structures étudiées (boucles, cycles et diamants) dans les mesures classiques. Le premier tableau résume la fréquence de ces structures dans la carte topologique obtenue avec nos mesures, après fusion des 1465 passes. Le deuxième tableau se focalise sur les boucles et les cycles dont on peut quantifier la présence en termes de *fréquence dans les traces* plutôt que dans le *graphe* obtenu par la fusion des traces (ce qui n'est pas le cas des diamants, qui sont mesurés *après* l'agrégation des traces).

	Nombre et pourcentage d'adresses `IP` impliquées		Nombre et pourcentage de destinations impliquées	
Boucles	2498	7.51%	1232	24.64%
Cycles	1572	4.73%	877	17.54%
Diamants locaux	13465	40.49%	4508	90.16%

	Instances	Signatures	Pourcentage de traces impliquées
Boucles	478707	5795	4.35%
Cycles	67229	5674	0.60%

TAB. 4.1 – Résumé des statistiques présentées dans cette section.

4.4 Artefacts de mesures par répartition de charge

La Section 4.1.2 a expliqué comment la répartition de charge pouvait entraîner des erreurs de mesure de topologie. Nous avons justement constaté que *toutes* les structures décrites dans la section précédente peuvent être provoquées par la répartition de charge.

Par exemple, les boucles non persistantes, qui sont celles qui apparaissent sporadiquement dans les traces vers une même destination, peuvent être causées par la répartition de charge comme le montre la Figure 4.13. Ces boucles peuvent être dues à une route où la répartition de charge agit sur deux chemins possibles de longueurs différentes de 1.

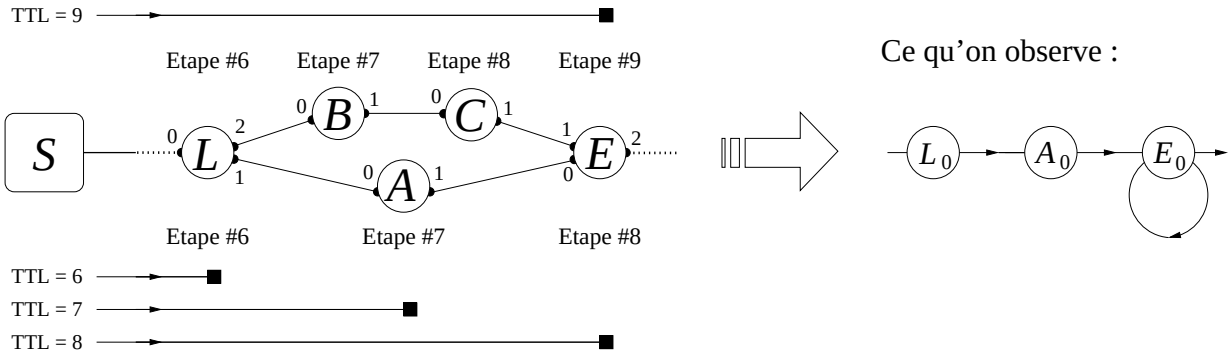

FIG. 4.13 – Boucle provoquée par la répartition de charge.

De manière similaire, de la répartition de charge entre deux routes ayant des longueurs différant de plus de 1 peut provoquer l'apparition de cycles dans nos mesures. Notons que la répartition de charge a généralement lieu entre des routes de longueurs identiques. Cela explique la relative rareté des boucles par rapport à la fréquence des routeurs utilisant la répartition de charge, et explique également pourquoi les cycles sont encore plus rares que les boucles (en termes d'instances).

La répartition de charge peut également induire des diamants, comme le montre la Figure 4.14. Notons que la présence de diamants ne relève pas forcément une erreur de mesure : de véritables diamants existent dans la topologie. Le problème est de différencier les vrais diamants des faux diamants.

Dans cette section, nous allons essentiellement comparer les données du `traceroute` classique (qui nous ont servi de base pour les statistiques données dans la section précédente) et celles de *Paris traceroute*. Les premières serviront de référence. *Paris traceroute* supprimant l'effet de la répartition de charge par flux, nous pourrons quantifier l'impact de cette dernière sur la présence des boucles, cycles et diamants dans les observations, ce qui nous semble un bon point de départ pour évaluer l'impact de la répartition de charge sur la validité des cartes `IP` en général.

Les statistiques présentées dans cette section sont accompagnées de commentaires que nous jugeons importants, et ne permettent pas d'obtenir facilement une vision d'ensemble. Pour cela, nous encourageons le lecteur à consulter la Table 4.2 située à la fin du chapitre,

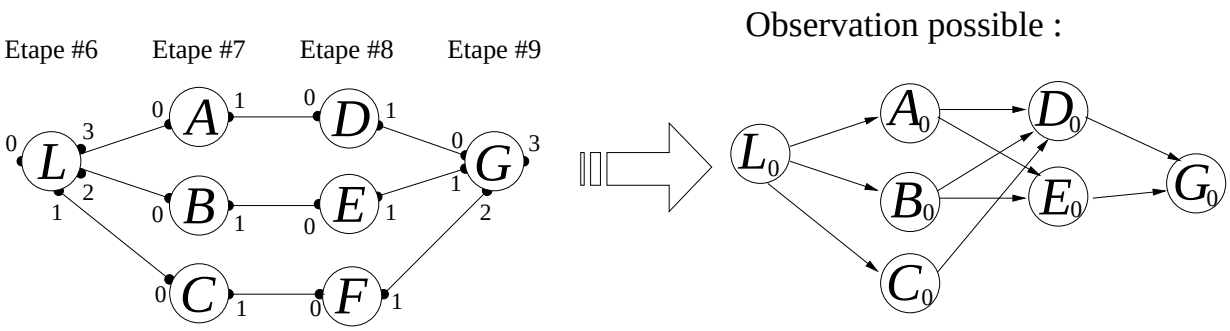

FIG. 4.14 – Un exemple de plusieurs diamants provoqués par la répartition de charge alors qu'aucun diamant n'existe dans la vraie topologie. Pour plus de clarté, nous avons omis de représenter les paquets sondes, trop nombreux.

dont la première ligne de chiffres quantifie la disparition des boucles, cycles et diamants entre les mesures classiques et celles effectuées avec *Paris traceroute*.

4.4.1 Boucles

Nous comparons la présence des boucles dans les mesures obtenues par le `traceroute` classique et celles obtenues par *Paris traceroute*, ces dernières évitant la répartition de charge par flux. De nombreuses instances de boucles disparaissent avec *Paris traceroute*, à savoir 79.9% d'entre elles. Notons qu'un faible nombre d'instances – représentant 0.21% du nombre d'instances vus avec `traceroute` classique – apparaît avec *Paris traceroute* alors qu'elles n'existaient pas dans les mesures classiques. Cela est dû au caractère sporadique de certaines boucles qui n'apparaissent qu'occasionnellement et n'ont pas été observées par le `traceroute` classique par simple "malchance". En comptant les *signatures* de boucles, ces chiffres deviennent respectivement 89.8% et 2.73%. Les premiers chiffres montrent déjà que la répartition de charge est très probablement la première cause d'apparition des boucles dans nos observations, et les deuxièmes illustrent la représentativité de nos mesures en indiquant combien de signatures et d'instances apparaîtraient si nous avions des mesures deux fois plus longues. Nous approfondissons notre étude en nous intéressant aux caractéristiques des boucles disparaissant avec *Paris traceroute*. La Figure 4.15 montre les probabilités d'apparition conditionnelles (voir Section 4.3.1) des boucles dans les mesures classiques et dans les mesures de *Paris traceroute*.

On peut constater que quasiment toutes les boucles n'étant ni systématiques (probabilité conditionnelle proche de 1) ni extrêmement sporadique (probabilité conditionnelle très proche de 0), donc celles apparaissant de manière non systématique mais à une fréquence non négligeable, ont disparu. Ce type de comportement est précisément celui qu'on attendrait de boucles causées par la répartition de charge, qui apparaissent quand le hasard de la répartition envoie des sondes consécutives sur deux routes ayant une différence de longueur de 1. Notons que les boucles persistantes qui apparaissaient dans les mesures classiques sont *toutes* restées présentes dans les mesures avec *Paris traceroute*, ce qui était

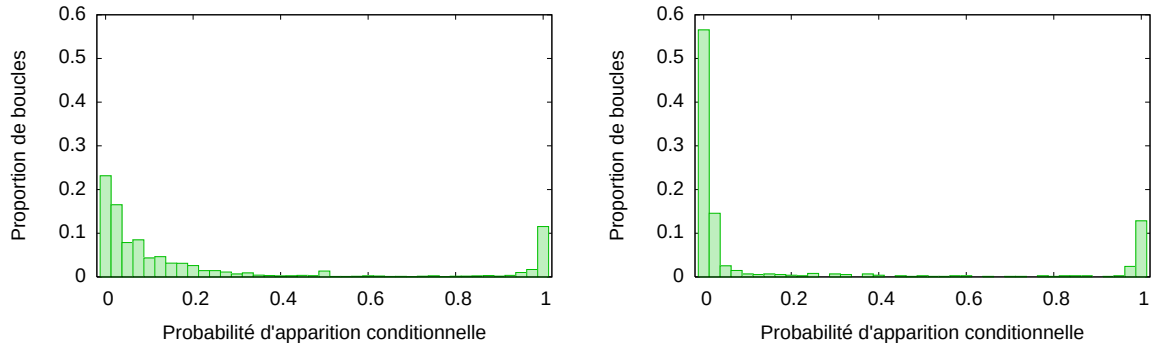

FIG. 4.15 – Répartition des probabilités d'apparition conditionnelles des boucles dans les mesures classiques (à gauche) et dans les mesures *Paris traceroute* (à droite).

attendu puisque les boucles persistantes ne peuvent raisonnablement pas être expliquées par la répartition de charge. Cela sera discuté plus précisément en Section 4.5.

Nous observons également que 89.3% des boucles systématiques mais non persistantes disparaissent. Nous avons vu comment un routeur répartissant la charge sur deux routes dont les longueurs diffèrent de 1 peut provoquer l'apparition de boucles. Nous allons à présent décrire un mécanisme de répartition de charge par flux capable de créer des boucles systématiques mais non persistantes, ce qui expliquerait la disparition d'une grande partie de ces boucles avec *Paris traceroute*. Nous utilisons la Figure 4.13 pour illustrer notre propos. Si le routeur L transmettait les paquets ayant un identifiant de flux pair vers B et les autres vers A, sachant que les sondes successives de `traceroute` incrémentent le port destination, le calcul de l'identifiant de flux peut être (selon les routeurs) tel que L transmettra systématiquement les sondes successives de manière alternée : un paquet sur deux ira vers A et un paquet sur deux vers B. Dès lors, seules deux traces sont possibles : une avec la sous-route (L_0, B_0, E_0, E_0) et une avec la sous-route (L_0, A_0, C_0, F_0), où F_0 est le routeur situé après E_0. La première sous-route contient donc une boucle systématique dont la probabilité d'apparition est 50% et la probabilité d'apparition conditionnelle est 100%.

Ces observations confirment le fait que la répartition de charge, qu'elle soit par flux ou par paquets, est une source importante de boucles. Nous ne disposons pas de mesures pour la répartition de charge par paquets, mais pouvons raisonnablement affirmer qu'elle peut provoquer des boucles de manière similaire à la répartition de charge par flux, la seule différence étant qu'elle ne sera pas supprimée par *Paris traceroute*. La présence de boucles persistantes, qui provoquent tout de même 7.45% des instances de boucles, semble cependant indiquer que d'autres phénomènes entrent en jeu. Nous reviendrons sur ce point en Section 4.5.

4.4.2 Cycles

De manière similaire, nous comparons les cycles observés dans les mesures classiques et ceux observés avec *Paris traceroute*. Nous observons la disparition de 39.7% des instances de cycles et également l'apparition – en bien plus faible quantité – d'instances de cycles (représentant 1.32% du nombre d'instances présent dans les mesures classiques). En termes de signatures, les chiffres sont plus importants et deviennent respectivement 79.5% et 11.0%. Le diagnostic immédiat est que la répartition de charge est une cause importante d'apparition des cycles, apparemment la première en nombre de signatures de cycles. Sa relativement faible implication dans l'apparition des *instances* de cycles peut s'expliquer par l'aversion qu'auront les routeurs à répartir leur charge sur des routes dont les longueurs diffèrent d'au moins 2.

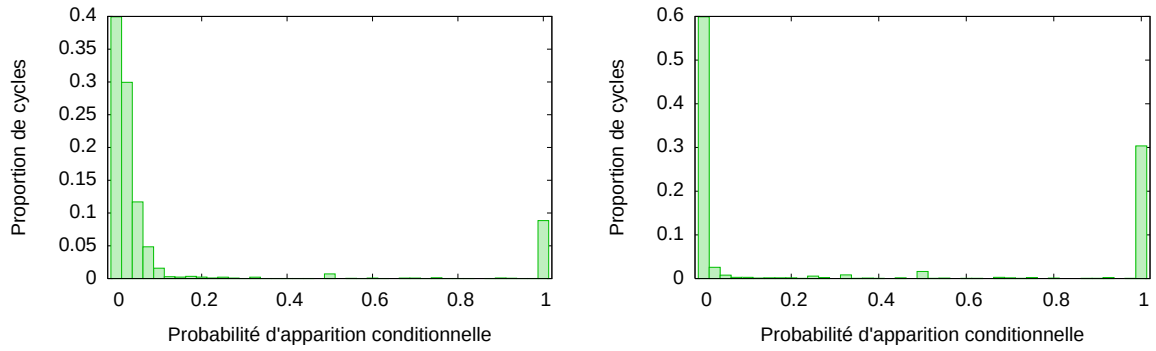

FIG. 4.16 – Répartition des probabilités d'apparition conditionnelles des cycles dans les mesures classiques (à gauche) et dans les mesures *Paris traceroute* (à droite).

Comme pour les boucles, nous analysons l'effet de *Paris traceroute* sur les différentes catégories de cycle (en termes de probabilité d'apparition). La Figure 4.16 montre que les cycles n'étant ni systématiques ni exceptionnels disparaissent presque complètement. Cela nous amène aux mêmes conclusions que pour les boucles, et confirme l'importance la répartition de charge en tant que cause d'apparition des cycles.

4.4.3 Diamants

Nous comparons la présence des diamants dans les mesures classiques et dans celles réalisées avec *Paris traceroute*.

Nous observons beaucoup moins de diamants globaux avec *Paris traceroute* : nous en relevons seulement 14318, ce qui revient à dire que 49.3% des diamants globaux disparaissent. La Figure 4.17 montre une comparaison des distributions des tailles des diamants globaux dans les deux mesures, à la même échelle.

Nous observons que la disparition de la moitié des diamants globaux s'accompagne d'une nette diminution de leur taille. De plus, ils sont **moins entrelacés** : les proportions

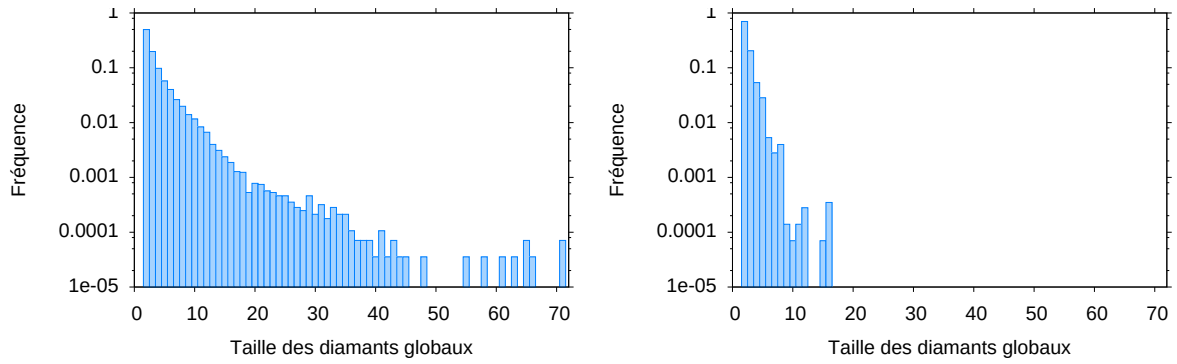

FIG. 4.17 – Distribution des tailles des diamants globaux dans les mesures classiques (à gauche) et dans les mesures *Paris traceroute* (à droite).

d'adresses `IP` relevées dans les mesures *Paris traceroute* appartenant à une tête de diamant, un corps et une queue s'élèvent respectivement à 11.3%, 24.8% et 21%, dont la somme vaut 57.1 alors que seules 39.5% des adresses `IP` appartiennent à un diamant. Rappelons que dans les mesures classiques, les chiffres étaient respectivement de 16.4%, 30.7% et 27.1% ce qui donnait une somme de 74.2 à comparer avec les 44.6% d'adresses appartenant à des diamants.

Le nombre, la taille et la complexité des d-diamants et des diamants locaux sont aussi considérablement réduits : nous n'observons que 43252 d-diamants engendrant 11280 diamants locaux, ce qui veut dire que 54.9% des d-diamants et 53.6% des diamants locaux disparaissent.

Notons cependant que la proportion de destinations vers lesquelles des d-diamants sont observés ne change quasiment pas, passant de 90.2% pour les mesures classiques à 89.6% pour les mesures *Paris traceroute*. Cela s'explique par les *vrais* diamants : comme nous l'avons décrit dans la Section 4.3.3, les faux diamants (éliminés par *Paris traceroute*) apparaissent surtout dans les routes subissant beaucoup de répartition de charge, où de vrais diamants ont de grandes chances d'apparaître également. Un chiffre est pour cela révélateur : étant donné une destination d donnant lieu à l'observation de d-diamants, le nombre attendu de d-diamants dans les mesures classiques était de 21.3, et tombe à 9.7 pour les mesures *Paris traceroute*. La Figure 4.18 présentes une comparaison des distributions des tailles des d-diamants observés dans les deux mesures.

Cette comparaison donne une indication sur l'impact de la répartition de charge par flux sur nos observations. Les diamants qui disparaissent étaient probablement dus à des faux liens reportés par `traceroute` classiques, et de même pour les diamants dont la taille de retrouve réduite. 50.7% des diamants globaux (resp. 46.4% des diamants locaux) observés avec `traceroute` classique disparaissent avec *Paris traceroute*, ce qui montre que les diamants sont souvent (environ une fois sur deux dans nos données) des artefacts de la mesure induits par la répartition de charge.

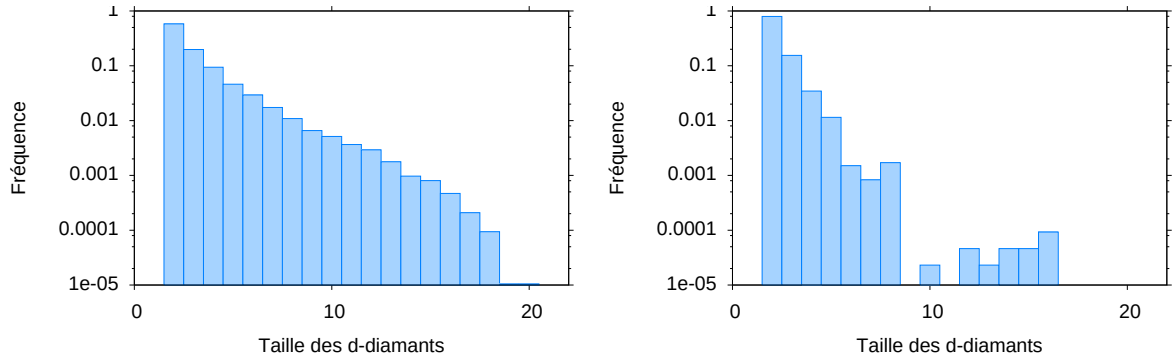

FIG. 4.18 – Distributions des tailles des d-diamants observés dans les mesures classiques (à gauche) et dans les mesures *Paris traceroute* (à droite).

4.5 Phénomènes exotiques

Nous avons pu voir dans la section précédente que la plupart des observations de boucles, cycles et diamants étaient dus à la répartition de charge par flux puisque qu'elles disparaissent quand on utilise *Paris traceroute*. Nous nous attaquons à présent à celles de ces observations qui restent présentes dans les mesures *Paris traceroute*, et cherchons leurs causes. Nous verrons que *Paris traceroute* fournit des informations utiles pour trouver et comprendre les phénomènes mis en jeu, et montrerons que l'essentiel des boucles et des cycles restants consiste bien en des artefacts de la mesure.

Nous présentons un certain nombre de statistiques dans cette section, dont beaucoup s'expriment en *proportions*. Par souci de précision, nous travaillons cette fois sur l'ensemble des boucles et cycles *qui restent présents dans les mesures* Paris traceroute et qui ne sont donc pas causés par la répartition de charge par flux. Une proportion de "100% des boucles" signifiera donc implicitement "100% *des boucles non causées par la répartition de charge par flux*".

4.5.1 Transit de paquets de TTL zéro

Nous avons trouvé une explication possible pour les boucles dans le manuel de `traceroute`, qui mentionne un bug dans le module de routage de certaines versions de BSD : ces routeurs décrémentent le TTL de paquets ayant un TTL de 1 – et le font donc descendre à zéro – et le transmettent au routeur suivant au lieu de le supprimer. Le routeur suivant, recevant un paquet de TTL 0, le supprimera immédiatement, mais le fait est que le routeur bogué n'aura pas répondu quand il était censé le faire. La Figure 4.19 montre les conséquences de la présence d'un tel routeur sur une trace.

Cette hypothèse intéressante peut se valider simplement dans nos mesures, puisque *Paris traceroute* fournit le TTL de la sonde au moment où elle a été supprimée. Si ce TTL vaut 1, tout est normal, mais s'il vaut 0 on peut penser que notre scénario se vérifie. Pour chaque instance de boucle de longueur 1, il suffit donc de regarder le TTL de la première des

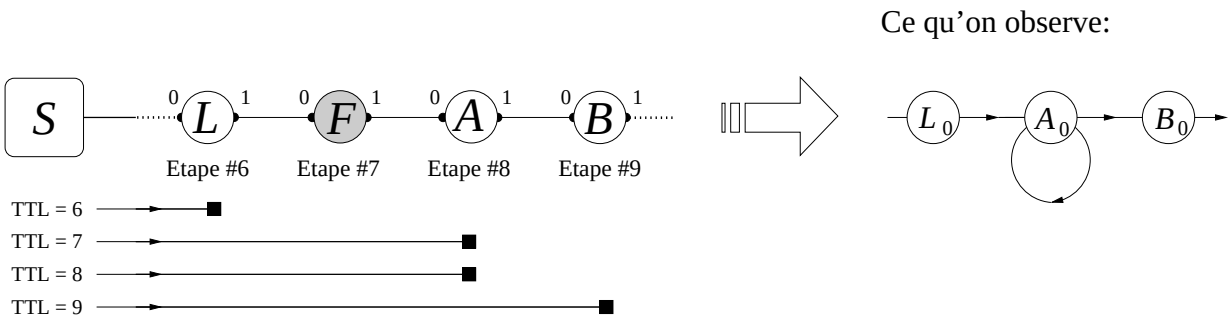

FIG. 4.19 – Boucle provoquée par un routeur mal configuré (F) transmettant des paquets de TTL 0.

deux sondes ayant engendré l'observation de cette boucle. La Figure 4.20 montre comment *Paris traceroute* met en valeur ce type d'anomalie.

```
6  10.10.146.134  452.135 ms
7  10.10.127.197  452.246 ms !T0
8  10.10.127.197  450.898 ms
9  10.10.127.37   451.499 ms
```

FIG. 4.20 – Morceau de sortie qu'aurait le programme *Paris traceroute* dans l'exemple de la Figure 4.19.

Notons que l'observation d'une boucle est vraiment là un artefact de la mesure, puisque les paquets ne traversent pas le même routeur deux fois de suite. Les paquets normaux (de TTL élevés) sont en fait transmis de manière complètement normale le long de cette route.

Nous avons lancé la détection de boucles de ce type dans nos données pour mesurer la proportion de boucles causées par ce problème. Nous avons pu confirmer que les boucles ainsi détectées étaient toujours persistantes ou au moins systématiques, ce qui semble forcé au vu du mécanisme mis en jeu. Parmi les boucles persistantes, 68.6% apparaissent causées par le transit de paquets de TTL zéro. Cela représente 17.1% des signatures de boucles, et 51.1% des instances. La différence entre ces deux derniers chiffres vient de la grande importance des boucles persistantes en termes de nombre d'instances : 63.4% des instances de boucles sont imputables aux boucles persistantes.

4.5.2 Boucles de routages

Pour les cycles, la première explication alternative à la répartition de charge qui vient à l'esprit est que les paquets suivent *réellement* une route cyclique (ou boucle de routage), et que les cycles ne sont donc pas des artefacts de mesure mais correspondent bel et bien à la réalité subie par les paquets. La grande étendue de nombreux cycles *périodiques* et

le fait que l'observation de tels cycles se fait souvent sur des traces n'atteignant pas leur destination confirme cette intuition.

Pour expliquer un tel comportement, nous avançons l'hypothèse de paquets piégés momentanément pendant la convergence des tables de routage[5]. Une observation qui tend à valider note hypothèse est que presque tous les cycles périodiques observés sont transitoires : ils apparaissent sur une route jusque-là normale, restent observables pendant un certain nombre de passes, puis disparaissent définitivement. Dans nos observations, cette durée peut prendre des valeurs bien dispersées. De nombreux cycles périodiques apparaissent seulement sur une passe, mais le plus tenace d'entre eux a tout de même été observé continûment pendant une semaine avant de disparaître.

Pour valider notre hypothèse définitivement, nous avons utilisé la technique de Spring et al. [98] pour vérifier que les réponses venant d'adresses `IP` identiques étaient réellement émises par le même routeur. Cette méthode est basée sur le fait qu'une proportion significative des routeurs (52% dans nos données) utilisent un compteur interne lors de l'affectation du champ `IP` *Identification* des paquets qu'ils créent, ce compteur étant incrémenté de 1 à chaque envoi de paquets. (Parmi les 48% routeurs restants, plus de 99% d'entre eux mettent systématiquement ce champ à zéro). Les paquets sondes d'une même trace étant envoyés dans un laps de temps relativement court, et les routeurs créant relativement peu de paquets eux-mêmes, les $2^{16} = 65{,}536$ valeurs possibles du champ `IP` *Identification* font que l'observation de paquets réponses ayant des champs *Identification* de valeurs quasi consécutives est un indice fort que ces réponses proviennent effectivement d'un même routeur. Une seule observation ne prouve rien, mais la répétition de telles observations au cours du temps peut tout de même nous donner une quasi-certitude quant à nos conclusions.

Nous avons donc appliqué cette méthode là où c'était possible (rappelons que certains routeurs n'utilisent pas de compteur pour le champ *Identification*, rendant leur "authentification" impossible) et avons observé un résultat impressionnant de clarté puisque 100% des adresses `IP` impliquées dans des cycles périodiques et associées à des paquets ayant des champs *Identification* non nuls ont pu être authentifiés avec une très forte probabilité comme provenant d'un même routeur. Cela montre donc que les cycles périodiques correspondent réellement à la route suivie par les paquets.

Ces cycles représentent 60.2% des signatures de cycles et 91.6% des instances.

Notons que 43% des traces contenant un cycle périodique ont tout de même atteint leur destination, ce qui reste tout à fait plausible puisqu'une trace complète prend environ 15 secondes alors que la convergence des tables de routage, d'après une étude récente [52], prend souvent moins de 10 secondes, n'affectant alors qu'une partie des traces concernées.

4.5.3 Traces interrompues

Une autre explication pour les boucles, qui est également valable pour les cycles, vient des réponses ICMP *Unreachable*. Certains routeurs, réalisant qu'un paquet ne peut être

[5] Après un changement dans la topologie, certains routeurs peuvent posséder des visions érronées de la topologie et doivent se réajuster, ce qu'on appelle convergence des tables de routage

transmis à sa destination pour telle ou telle raison, peuvent supprimer le paquet et envoyer à son émetteur un message ICMP spécial tel que les ICMP *Host Unreachable*, *Network Unreachable*, et *Source Quench* (entre autres). Si cela arrive avec une sonde, `traceroute` classique et *Paris traceroute*, en recevant un paquet ICMP autre que *Time Exceeded*, considèrent alors tous deux que la trace est terminée. Si pendant le sondage un routeur répond d'abord normalement quand la sonde l'atteint, puis un peu plus tard envoie un message de ce type en réponse à une sonde qu'il aurait dû transmettre, il apparaîtra deux fois sur la trace : la première fois quand il envoie le message ICMP *Time Exceeded*, et la deuxième fois quand il envoie l'autre message ICMP.

Nous avons observé empiriquement que dans ces cas, la plupart du temps le routeur envoie le message spécial en réponse à la sonde immédiatement consécutive à l'émission de sa réponse *Time Exceeded*, ce qui fait donc apparaître une boucle en fin de trace.

Il est facile de détecter ce type d'évènement puisque *Paris traceroute* affiche le type de message ICMP reçu. De plus, nous recherchons des boucles ou cycles constituées de la double apparition d'une adresse `IP` dont l'une est forcément située à la fin de la trace, ce qui restreint encore le champ de recherche et permet d'écarter toute erreur. Nous montrons en Figure 4.21 deux exemples de sorties de *Paris traceroute* présentant respectivement une boucle et un cycle dus à ce phénomène.

```
(...)
17 11.14.43.212 91.820 ms {30705}
18 11.14.41.224 102.166 ms {0}
19 11.14.42.159 103.704 ms {0}
20 68.77.67.222 103.989 ms {0}
21 29.43.20.53 104.355 ms {0}
22 29.43.75.68 105.720 ms {56753}
23 64.184.0.182 120.246 ms {23140}
24 26.49.101.79 121.193 ms {7992}
25 26.49.101.79 119.644 ms !H {7993}
```

```
(...)
12 62.4.64.54 12.224 ms {0}
13 23.83.46.21 12.734 ms {0}
14 23.83.57.12 12.591 ms {17266}
15 23.83.46.206 12.783 ms {36598}
16 23.83.46.21 12.535 ms {3431}
17 *
18 *
19 *
20 23.83.46.206 64.585 ms !H !T7 {36941}
```

FIG. 4.21 – Exemples partiels de sortie de *Paris traceroute* lors de traces interrompues, mettant en évidence une boucle (à gauche) et un cycle (à droite). La sortie est directement tirée de nos données, seules les adresses `IP` ont été modifiées par confidentialité. Les `!H` indiquent la réception d'un paquet ICMP de type *Host Unreachable*, qui provoque la terminaison d'une trace. Le chiffre entre accolades représente la valeur du champ `IP` *Identification* des réponses : on remarque que les valeurs affichées aux cotés des réponses d'adresse `IP` identiques sont très proches, ce qui tend à authentifier notre hypothèse. Le `!T7` visible à coté de la dernière réponse, à droite, signifie que notre sonde a été supprimée alors qu'elle avait un TTL égal à 7.

Les traces interrompues représentent 62.0% des signatures de boucles et 27.0% des signatures de cycles. En termes d'instances, ces nombres deviennent respectivement 19.7% et 3.5%. La proportion d'instances est plus faible à cause du caractère occasionnel du

phénomène en jeu. Notons tout de même la présence de quelques rares boucles et cycles observés de manière persistante.

Notons enfin que la forme continue de la distribution des périodes des signatures de cycles (en haut dans la Figure 4.8) est essentiellement due à ce phénomène, qui engendre 88.8% des signatures de cycles de période supérieures à 12.

4.5.4 Adresses IP fictives

Notre dernière hypothèse pour la présence de boucles dans les observations vient de l'observation de *n-boucles persistantes*, toujours inexpliquée. Certains sous-réseaux sont connus pour être réfractaire aux mesures `traceroute` car étant derrière des boîtes NAT ou des pare-feu. Dans ces sous-réseaux, les paquets sortants (donc les réponses à `traceroute`) peuvent voir leur adresse `IP` remplacée, par exemple en faveur de celle de la passerelle de bordure. Nous illustrons cela en Figure 4.22

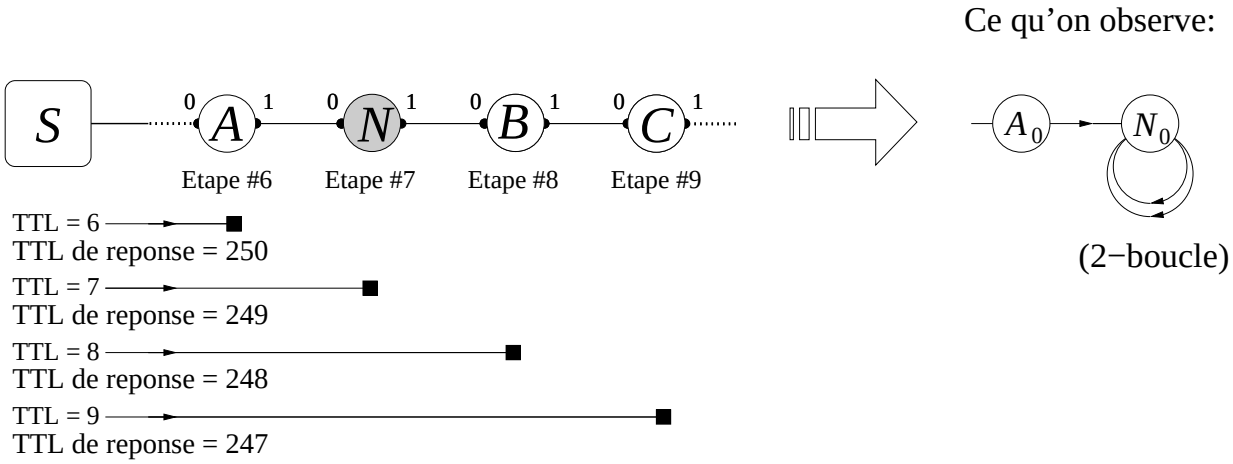

FIG. 4.22 – Exemple de n-boucle provoquée par un sous-réseau NAT dans lequel la passerelle N remplace l'adresse `IP` source des paquets sortants (ceux émis par B et C) par sa propre adresse, faisant croire à la source que les paquets ont été émis par elle.

Pour valider cette hypothèse, nous avons cherché à trouver des indices montrant que les réponses consécutives reçues depuis la même adresse `IP` venaient en fait de routeurs différents. Le meilleur moyen que nous ayons trouvé consiste en la comparaison des TTL des réponses : si ces dernières viennent d'un même routeur, alors les TTL seront *a priori* identiques, ou du moins *de temps en temps* identiques : si la route retour subit de la répartition de charge sur des routes de longueurs différentes, le TTL de la réponse peut varier, mais le fera de manière aléatoire. Dès lors, l'observation de réponses provenant d'une même adresse `IP` mais ayant *systématiquement* des TTL différents montre que les réponses ne peuvent pas être émises par le même routeur[6]. L'adresse `IP` est donc truquée,

[6] Nous avons préféré ne pas utiliser directement la méthode de comparaison des champs `IP` *Identification*

que ce soit par le phénomène décrit ci-dessus ou par un autre. Lorsque, comme illustré sur la Figure 4.19, les TTL des réponses consécutives diminuent de 1 à chaque étape pendant toute la longueur de la boucle – comme on l'attendrait en présence d'une route normale – il est encore plus évident qu'un mécanisme de trucage de l'adresse `IP` source des réponses est à l'œuvre.

Nous appelons *adresses* `IP` *truquées* le phénomène décrit ici, et *boucles truquées* les boucles causées par ce type de phénomène. Nous observons que 6.8% des signatures de boucles et 18.6% des instances appartiennent à des boucles truquées. Les boucles truquées observées sont essentiellement persistantes ou systématiques. Un résultat intéressant est que la quasi totalité des boucles persistantes détectées dans nos observations et n'ayant pas été expliquées par l'une des causes mentionnées précédemment ont été identifiées en tant que boucles truquées. De plus, toutes les n-boucles de longueur > 1 se sont également avérées truquées.

Ce phénomène peut également provoquer l'apparition de cycles si (par exemple) de la répartition de charge a lieu sur la route retour et que les réponses peuvent passer par deux passerelles N_1 et N_2 qui remplaceront donc l'adresse des réponses alternativement par N_1 et N_2. Nos observations relèvent peu de cycles de ce type : seuls 1.6% des signatures de cycles et 3.0% des instances.

4.5.5 Bilan

En considérant la répartition de charge par flux, le transit de paquets de TTL zéro, les boucles de routage, les routes interrompues et les adresses `IP` truquées, nous avons pu expliquer la provenance de la moitié des diamants et de plus de 95% des boucles et cycles observés, que ce soit en termes d'instances ou de signatures.

La Table 4.2 présente le détail des proportions de boucles, cycles et diamants imputables à chaque phénomène cité. Les proportions sont exprimées en termes de signatures et d'instances pour les boucles et les cycles, et en termes de diamants globaux et de diamants locaux pour les diamants. Contrairement à la convention utilisée dans le reste de cette section, nous écrivons ici les proportions par rapport aux mesures `traceroute` classiques. Certains phénomènes ne peuvent provoquer l'apparition de certaines structures, nous avons utilisé des tirets '-' pour le signaler.

Même si nos mesures ne se veulent pas représentatives du `traceroute` moyen effectué sur l'Internet, ces chiffres montrent à quel point *Paris traceroute* est capable de nettoyer la plupart des artefacts de mesure. Le cas des structures encore inexpliqués reste intéressant. Nous savons que certains diamants sont réels (mais combien ?) et pensons que la répartition de charge par paquets est responsable de l'essentiel des boucles et cycles restants, mais ne pouvons pas fournir de chiffres fiables démontrant cette supposition.

utilisée dans la sous-section précédente, celle-ci étant adéquate pour la *confirmation* que des paquets viennent d'un même routeur mais n'étant pas désignée pour faire l'inverse. Cependant, nous avons tout de même vérifié que les deux méthodes ne donnaient pas de résultats conflictuels : aucune des adresses `IP` identiques déclarées comme venant de routeurs différents par la méthode du TTL n'ont été signalées comme provenant d'un même routeur par la méthode basée sur le champ `IP` *Identification*.

	Diamants		Boucles		Cycles	
	globaux	locaux	signatures	instances	signatures	instances
Rep. de charge par flux	49.20	53.60	87.07	79.69	68.50	38.38
Transit de TTL nul	-	-	2.21	10.38	-	-
Boucles de routage	-	-	-	-	18.96	56.44
Traces interrompues	-	-	8.02	4.00	8.51	2.16
Adresses `IP` truquées	-	-	0.88	3.78	0.50	1.85
Cause inconnue	50.80	46.40	1.81	2.15	3.50	1.17

TAB. 4.2 – Répartition des causes identifiées des structures étudiées dans ce chapitre. Les chiffres représentent tous des pourcentages.

4.6 Conclusion et perspectives

Les travaux présentés dans ce chapitre ont mis en évidence des différences entre la topologie `IP` réelle et son image telle qu'on peut la déduire naïvement des sorties `traceroute`. Nous avons tenté de quantifier ces différences en nous focalisant sur trois entités bien définies en termes de traces et correspondant à des structures de graphes particulières : les boucles, cycles et diamants. À l'aide d'une étude statistique, nous avons montré l'importante présence de ces entités dans nos traces et le caractère artificiel de la quasi-totalité des cycles et des boucles et de la moitié des diamants. Nous avons ensuite expliqué comment la répartition de charge par flux peut provoquer l'observation de tels artefacts et montré qu'elle était la cause principale de ces observations en pratique. Nous avons ensuite mis en évidence d'autres phénomènes capables d'engendrer des boucles et des cycles et avons quantifié leur impacts respectifs.

L'outil *Paris traceroute* que nous avons développé et rendu public [135] permet d'éliminer les effets de la répartition de charge par flux, et fournit donc une information plus fiable, comme le montre nos études sur les artefacts de mesure. Ses nombreuses options lui donnent une plus grande flexibilité d'utilisation. Sa sortie plus complète enrichit l'information fournie, ce qui permet de détecter les phénomènes décrits dans ce chapitre et de corriger leurs effets.

Dans le cadre de simples mesures effectuées avec discrétion, le gain de fiabilité et de précision dans la mesure est non négligeable. Dans le cadre de grands projets de cartographie de la topologie `IP`, l'utilisation de *Paris traceroute* au lieu du `traceroute` classique est également d'un grand bénéfice : sur la carte `IP` collectée par nos expériences, 21.8% des liens du graphe obtenu avec les mesures `traceroute` classiques s'avèrent être des faux liens grâce aux comparaisons avec les mesures *Paris traceroute* et aux diagnostics développés dans ce chapitre. Ce chiffre augmente encore si on considère les *arcs* du graphe dirigé (chaque lien mesuré possède en effet une direction correspondant à l'éloignement par rapport à la source) : 26% des arcs sont fictifs dans les mesures classiques.

À l'avenir, nous espérons enquêter sur la répartition de charge par paquets, identifier d'autres structures simples (comme les boucles, cycles ou diamants) permettant de signaler

des anomalies dans la mesure et déterminer les phénomènes mis en jeu. Nous espérons également déployer nos mesures sur une plus large étendue, en considérant par exemple de multiples sources bien réparties dans l'Internet et en comparant les résultats obtenus.

Conclusion

Nous nous sommes intéressé dans cette thèse aux deux grandes problématiques que sont la modélisation et la mesure de l'Internet. Nous résumons dans cette conclusion nos résultats tout en décrivant notre démarche globale. Nous proposons également, à la fin de cette conclusion, une courte synthèse plus personnelle de ce que cette thèse a pu m'apporter et de ce qu'il faut en retenir.

Modélisation par graphe aléatoire

En toute généralité, la modélisation est un constituant fondamental de la recherche : on a souvent besoin de simplifier, classer, nommer des problèmes avant de pouvoir y réfléchir.

Dans le cadre de notre thèse, nos travaux préliminaires sur l'Internet exigeaient le lancement de simulations locales sur des réseaux virtuels modélisant l'Internet. Le choix d'un modèle adéquat fut une étape importante : les modèles existants sont très différents dans leur nature comme dans les propriétés des réseaux synthétiques obtenus, et sont plus ou moins difficiles à mettre en œuvre. Dans notre cas, la nécessité forte de travailler sur des très grands réseaux, dont la taille se chiffre en millions de sommets, exige des modèles simples et performants.

Devant l'impossibilité d'imiter la réalité à la perfection et donc de travailler sur une réplique *parfaite* de l'Internet, nous avons préféré utiliser un modèle épuré, générant des graphes tirés aléatoirement parmi les graphes respectant à la lettre des propriétés ciblées, à savoir le nombre de sommets, la distribution des degrés et la connexité, choisis pour leur rôle particulièrement important.

Voulant générer des graphes selon ce modèle, et comme les implantations déjà réalisées étaient beaucoup trop lentes pour nos besoins car de complexité au mieux quadratique, nous avons entrepris de créer notre propre outil de génération aléatoire de graphes connexes à séquence de degrés prescrite. Au passage, nous avons tenté d'utiliser une technique très simple donnant des résultats approchés, mais avons constaté une erreur trop grande entre les paramètres souhaités et ceux obtenus, ce qui témoigne d'un biais trop important pour que les graphes ainsi générés soient utilisables en toute rigueur.

Pour obtenir des performances acceptables, nous avons dû travailler sur l'algorithme lui-même, et avons optimisé une heuristique utilisée dans la meilleure implantation alors existante. Forts de l'expertise acquise, nous avons pu mettre au point une technique simple permettant d'obtenir une réduction considérable du temps de calcul pour des résultats

identiques, rendant possible la génération de graphes de plusieurs dizaines de millions de sommets en des temps de l'ordre de la minute. Nous avons ensuite publié les sources de notre programme sur le Web, et avons connu un certain succès puisque de nombreux articles utilisent déjà notre outil [134].

Une fois notre modèle implanté, son application à la modélisation de l'Internet passe par la connaissance précise des paramètres clés à imiter. Dans notre cas, il nous était nécessaire de connaître à la fois la taille du réseau et sa distribution de degrés pour pouvoir être en mesure de générer des modèles réduits réalistes. Le problème est qu'aucune technique connue ne permettait de mesurer avec précision ce type d'informations sur l'Internet. Il existe pourtant des mesures de la topologie de l'Internet, qui restent très incomplètes mais fournissent tout de même une quantité d'information énorme et dont nous pensions pouvoir tirer une mesure relativement précise de la distribution de degrés et/ou de la taille du réseau. Nous nous sommes donc lancés dans cette problématique : comment, à partir de mesures partielles et biaisées d'un objet tel que la topologie `IP`, peut-on déduire une valeur précise de certaines quantités dans le réseau originel ?

Mesure de paramètres sur la topologie `IP`

Nous avons commencé par nous intéresser à la mesure de la plus élémentaire des propriétés : la taille, i.e. le nombre de nœuds du réseau. Ce travail étant très prospectif, nous l'avons développé et évalué dans un cadre très formel, schématique et assez éloigné de la réalité. La problématique n'était pas ici de simplement déterminer le nombre de routeurs de l'Internet, mais plutôt de créer des méthodes permettant d'inférer cette quantité de façon transposable à d'autres métriques (telles que le nombre de liens). Nous nous sommes donc placés à un niveau plus théorique nous permettant de généraliser l'approche : étant donné une mesure partielle d'un réseau, basée sur la collecte de chemins existants entre un ensemble S de sources et T de destinations, connaissant S et T, déterminer la taille du réseau original. Nous nous affranchissons ainsi de toute considération pratique et la détermination de la taille de l'Internet n'est plus qu'une application possible de notre travail. Notons cependant que de nombreux choix, en termes de modélisation, ont été faits en tentant d'imiter Skitter, un projet de mesure de la topologie `IP` basé sur `traceroute`.

Notre première approche, et notre premier échec, fut de s'intéresser à une grandeur clé pour toute mesure basée sur une collecte de chemins : la centralité de plus court chemin. L'idée directrice est que la probabilité d'observer un sommet est directement liée au nombre de chemins sur lequel il est. Cela nous a permis de mieux cerner le problème et d'en comprendre les tenants. Notamment, le parallèle entre la détermination de la taille du réseau et un problème statistique bien connu, le "species problem", nous permit de l'aborder sous un angle bien plus circonspect. Le non-résultat peut se résumer à cela : la difficulté de l'évaluation de la taille de l'Internet à partir d'une mesure basée sur des chemins vient de l'extrême hétérogénéité de la centralité des sommets. En d'autres termes, on dispose d'énormément d'information sur un petit nombre de sommets (les sommets centraux) alors que les sommets peu, ou pas observés, constituent l'immense majorité du graphe.

Les deux pistes explorées par la suite furent inspirées des méthodes usuelles en statistiques pour résoudre ce type de problème. La première méthode, dite du leave-one-out, s'apparente sur le principe à l'échantillonnage aléatoire, et présente d'excellents résultats dès lors que la taille du graphe mesuré dépasse la racine de la taille du graphe originel, ce qui est effectivement le cas pour la plupart des mesures de l'Internet. Cette méthode souffre cependant de graves inconvénients, le principal étant que dans le cas de l'Internet elle ne peut être immédiatement transposable au nombre d'arêtes, et encore moins aux degrés des sommets : son applicabilité dépend de la capacité à sélectionner un sommet au hasard, ce qui est possible grâce à la relative haute couverture de l'espace d'adressage IPv4, mais impensable dans l'espace des liens.

La deuxième méthode est un paradigme classique en statistiques : le re-échantillonnage. Cette méthode à l'avantage d'être aisément généralisable aux arêtes, voire aux degrés. Son inconvénient principal est son manque de précision : même si on se rapproche effectivement des bonnes valeurs par rapport à la carte partielle, on en reste assez loin, même avec des cartes partielles relativement complètes.

L'étude du bon comportement de ces méthodes, en particulier de la deuxième, nécessite d'énormes ressources de calculs, et fut un frein empêchant de trop complexifier les modèles utilisés, nous forçant à rester sur des topologies purement synthétiques en tant que réseau témoin, et sur des modèles simplistes pour l'obtention d'une mesure partielle du réseau. Il n'est dès lors pas surprenant que leur performance se voie réduite, notamment dans le cas du re-échantillonnage, quand on les applique sur l'Internet : la qualité de nos estimateurs était loin d'être garantie sur une topologie aussi différente. Nous ne sommes donc pas en mesure de déterminer avec précision la taille de l'Internet. Néanmoins, nous avons réalisé un premier pas vers ce but, et nos méthodes gardent le mérite de donner des valeurs corrigées par rapport à la simple observation de la mesure partielle.

Un outil `traceroute` plus précis

La mesure quantitative des propriétés fondamentales du réseau Internet est non aisée, pour les raisons décrites ci-dessus. L'une d'entre elles est le fossé existant entre le comportement du réseau en pratique et les modèles formels utilisés, trop "parfaits". Nous nous sommes notamment intéressés à évaluer à quel point les cartes de l'Internet obtenues par `traceroute` recèlent de fausses informations.

En l'absence d'une carte de référence de l'Internet, et ne pouvant donc pas déterminer si un nœud ou lien mesuré est fictif, nous nous sommes intéressés aux structures insolites présentes dans les cartes de l'Internet, qui nous semblait être probablement dues à des mesures localement erronées.

Nous nous sommes concentrés sur trois types de structures jugées anormales : les boucles, les cycles, et les diamants, vus en tant que structures de graphe. La nécessité d'enquêter plus profondément sur les phénomènes mis en jeu par de telles structures nous a poussé à nous rapprocher du niveau réseau, et à quitter momentanément le paradigme 'graphe'. Nous avons donc défini ces structures en termes d'observation de routes, ceci afin d'effectuer un recensement de leur présence dans les cartes de l'Internet.

Une telle étude nécessite de savoir exactement comment l'outil `traceroute` utilisé pour les mesures a fonctionné. Ce comportement pouvant varier d'une machine à l'autre, nous avons réalisé notre propre jeu de mesure, bien moins important que les données du projet Skitter par exemple, mais suffisant pour le but recherché. Cela nous permet notamment d'approfondir notre recherche en fonction des résultats obtenus, en d'autres termes de modifier notre façon de mesurer en fonction de ce que l'on a vu et de ce que l'on veut voir.

L'approche consistant à analyser les motifs d'apparition des anomalies s'est révélée payante car elle nous a permis d'échafauder des hypothèses quant aux causes des anomalies, hypothèses que nous avons par la suite vérifiées et validées. Allant plus loin, nous avons élaboré une méthodologie capable d'éliminer toutes les anomalies que nous avons pu identifier. Le pourcentage de boucles, cycles et diamants restants après un tel 'nettoyage' des traces est une fraction infime des anomalies détectées au départ, de 1% à 4% selon les cas. Nous avons également mis en place, en open-source sur le web, un outil `traceroute` amélioré utilisant partiellement cette méthodologie, et bien moins susceptible de produire de faux liens que l'outil `traceroute` classique. L'outil ainsi développé s'appelle *Paris traceroute* et suscite déjà un certain intérêt dans la communauté réseaux. Quelques articles et cours universitaires citant notre outil ont déjà été soumis, et son existence s'est fait connaître sur le Web à travers les forums et même déjà dans certains cours universitaires.

Synthèse

L'Internet peut se voir à plusieurs échelles, de la vue la plus globale à la plus microscopique, c'est-à-dire du point de vue du théoricien des graphes au point de vue de l'ingénieur réseau. Ces deux manières d'étudier l'Internet se ressemblent peu, et appartiennent à deux domaines de recherche relativement éloignés. Elles sont pourtant complémentaires, et cette thèse a été l'occasion de les rapprocher.

En pratique, la difficulté d'adopter une approche formelle, modélisatrice de l'Internet en prenant en compte toute sa complexité est rédhibitoire. La solution adoptée le plus souvent par les théoriciens est de s'affranchir des détails. Récemment, de nombreux travaux se sont concentrés pour démontrer la fausseté ou du moins le caractère approximatif des résultats trouvés dans cet état d'esprit. Si ces études sont rigoureuses, elles ne proposent pas de solution pour autant. Le consensus à l'heure actuelle est que quel que soit le modèle ou la mesure utilisés par une étude sur Internet, les résultats doivent être soumis à caution.

Nous avons tenté d'adopter une démarche scientifique adaptée. Dans tous nos travaux, de nombreux allers-retours entre théorie et pratique nous ont permis d'invalider de nombreuses approximations ou choix de modèles, et l'étude des déviations entre réalité et modèle ou réalité et mesure nous ont permis de mieux cerner les sources d'erreurs, et parfois à les corriger.

Notre contribution globale est de réduire le fossé entre théorie et pratique dans le contexte de l'étude de la topologie de l'Internet, ou plus modestement de faire évoluer les choses en ce sens. Plutôt que de trouver les failles existantes dans telle ou telle approche simplificatrice, nous avons cherché à introduire de la rigueur et à montrer qu'il était possible d'étudier l'Internet plus 'proprement'. Notre algorithme de génération de graphes aléatoires répond en effet au besoin de générer des graphes ayant *exactement* une séquence de degrés donnés tout en restant connexe, mais permet aussi d'enquêter *rigoureusement* sur le rôle de la distribution de degrés dans le comportement d'un grand réseau. L'inférence de la taille du réseau tente, quant à elle, de répondre à la question : "Est-il possible de déduire des informations *pertinentes* et *précises* sur Internet à partir des cartes `traceroute` ?". Enfin, notre étude des artefacts de la topologie `IP` créés par `traceroute` a permis de remplacer ce dernier par un outil capable de déterminer les routes avec une rigueur accrue.

Si nos apports sont très variés par leur nature, ils ont en commun cette volonté de poser des jalons aidant à analyser l'Internet de façon plus rigoureuse. Nous avons pu mesurer, en faisant face aux problèmes liés à la taille et la complexité du réseau, l'étendue du chemin qui reste à parcourir pour pouvoir traiter Internet comme un objet bien connu. La tâche est immense, et notre apport reste modeste à cet égard.

Ajoutons enfin que les problématiques rencontrées dans cette thèse peuvent pour beaucoup s'appliquer à d'autres grands réseaux d'interactions. Les réseaux de téléphonie fixe et portable en sont un exemple, mais des réseaux de nature plus variés partagent également des caractéristiques communes avec l'Internet. Nous espérons que les techniques développées dans cette thèse pourront servir à l'étude de ceux-ci.

Bibliographie

[1] Dimitris Achlioptas, Aaron Clauset, David Kempe, and Cristopher Moore. On the bias of traceroute sampling or, powerlaw degree distributions in regular graphs. In *Proc. ACM STOC*, 2005.

[2] Cedric Adjih, Leonidas Georgiadis, Philippe Jacquet, and Wojciech Szpankowski. Multicast tree structure and the power law. In *Proc. ACM-SIAM Symposium On Discrete Algorithms*, Janvier 2002.

[3] Cedric Adjih, Leonidas Georgiadis, Philippe Jacquet, and Wojciech Szpankowski. Multicast tree structure and the power law. *IEEE Transactions On Information Theory*, 52(4) :1508–1521, Avril 2006.

[4] William Aiello, Fan Chung, and Linyuan Lu. A random graph model for massive graphs. In *Proc. ACM STOC*, 2000.

[5] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1) :47–97, Janvier 2002.

[6] David G. Andersen, Nick Feamster, and Hari Balakrishnan. Topology Inference from BGP Routing Dynamics. In *Proc. ACM SIGCOMM Internet Measurement Workshop*, November 2002.

[7] Nadia Ben Azzouna, Christine Fricker, and Fabrice Guillemin. Modeling adsl traffic on an ip backbone link. Rapport de Recherche 4909 de l'INRIA, Août 2003.

[8] Fred Baker. RFC 1812 : Requirements for IP Version 4 routers, http://www.faqs.org/rfcs/rfc1812.html, Juin 1995.

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

[10] Paul Barford, Azer Bestavros, John Byers, and Mark Crovella. On the marginal utility of network topology measurements. In *Proc. ACM SIGCOMM*, 2001.

[11] Alain Barrat and Martin Weigt. On the properties of small-world network models. *The European Physical Journal B*, 13(3) :547–560, Février 2000.

[12] Marc Barthélemy. Betweenness centrality in large complex networks. *The European Physical Journal B*, 38 :163–168, 2004.

[13] Steven Bellovin. A technique for counting NATted hosts. In *Proc. ACM SIGCOMM Internet Measurement Workshop*, Novembre 2002.

[14] John Bunge and M. Fitzpatrick. Estimating the number of species : A review. *Journal of the American Statistical Association*, 88(421) :364–373, 1993.

[15] Ross Callon. RFC 1195 : Use of OSI IS–IS for Routing in TCP/IP and Dual Environments, http://www.faqs.org/rfcs/rfc1195.html. RFC 1195, Décembre 1990.

[16] Hyunseok Chang, Sugih Jamin, Z. Morley Mao, and Walter Willinger. An empirical approach to modeling inter-AS traffic matrices. In *Proc. ACM SIGCOMM Internet Measurement Conference*, Octobre 2005.

[17] Cisco. How does load balancing work ? See http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a0080094820.shtml.

[18] Aaron Clauset and Cristopher Moore. Accuracy and scaling phenomena in internet mapping. *Physical Review Letters*, 94 :018701, Janvier 2005.

[19] Jacomo Corbo and Thomas Petermann. Selfish peering and routing in the internet. *ArXiv Computer Science e-prints*, Octobre 2004.

[20] Luca Dall'Asta, José Ignacio Alvarez-Hamelin, Alain Barrat, Alexei Vázquez, and Alessandro Vespignani. Exploring networks with traceroute-like probes : theory and simulations. *Theoretical Computer Science*, 355 :6–24, 2006.

[21] Sven Dietrich, Neil Long, and David Dittrich. The history and future of distributed systems attack methods. In *IEEE Symposium on Security and Privacy*, Mai 2000.

[22] Registre Internet Régional pour l'Afrique, http://www.afrinic.net.

[23] Registre Internet Régional pour l'Asie et le Pacifique, http://www.apnic.net.

[24] Registre Internet Régional pour l'Amérique du Nord, http://www.arin.net.

[25] Berkeley Internet Name Domain, http://www.isc.org/sw/bind/.

[26] Registre Internet Régional pour l'Amérique du Sud, http://www.lacnic.net/en.

[27] Registre Internet Régional pour l'Europe, l'Asie Centrale et le Moyen-Orient, http://www.ripe.net/index.html.

[28] Benoit Donnet, Philippe Raoult, Timur Friedman, and Mark Crovella. Efficient algorithms for large-scale topology discovery. In *Proc. ACM SIGMETRICS*, 2005.

[29] Sergey N. Dorogovtsev and Jose Fernando F. Mendes. Evolution of networks. *Advances in Physics*, 51(4) :1079–1187, Juin 2002.

[30] Christos Douligeris and Arikaterini Mitrokotsa. Ddos attacks and defense mechanisms : classification and state-of-the-art. *Computer Networks*, 44(5) :643–666, Avril 2004.

[31] Bradley Efron. The jackknife, the bootstrap, and other resampling plans. *SIAM CBMS-NSF Monographs*, 38, 1982.

[32] Paul Erdös and Tibor Gallai. Graphs with prescribed degree of vertices. *Matematikai Lapok*, 11 :264–274, 1960.

[33] Paul Erdös and Alfréd Rényi. On random graphs. *Publicationes Mathematicae Debrecen*, 6 :290–297, 1959.

[34] Vijay Erramilli, Mark Crovella, and Nina Taft. An independent-connection model for traffic matrices. In *Proc. ACM SIGCOMM Internet Measurement Conference*, Octobre 2006.

[35] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *Proc. ACM SIGCOMM*, Septembre 1999.

[36] Tomás Feder, Adam Guetz, Milena Mihail, and Amin Saberi. A local switch markov chain on given degree graphs with application in connectivity of peer-to-peer networks. *preprint*, 2006.

[37] Jean-Michel Fourneau and Houssame Yahiaoui. Génération de topologies réalistes pour la simulation du routage interdomaine. In *Proc. ALGOTEL*, Mai 2005.

[38] Ove Frank. Sampling and inference in a population graph. *International Statistical Review*, 48(1) :33–41, Avril 1980.

[39] Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1) :35–41, Mars 1977.

[40] Christos Gkantsidis, Milena Mihail, and Ellen Zegura. The markov chain simulation method for generating connected power law random graphs. In *Proc. Workshop on Algorithm Engineering and Experiments*, 2003.

[41] Khim Yong Goh, B. Kahng, and D. Kim. Universal behavior of load distribution in scale-free networks. *Physical Review Letters*, 87(27) :278701, Décembre 2001.

[42] Irving John Good. On the population frequencies of species and the estimation of population parameters. *Biometrika*, 40 :237–264, 1953.

[43] Ramesh Govindan and Vern Paxson. Estimating router ICMP generation delays. In *Proc. Passive and Active Measurement Conference*, Mars 2002.

[44] Ramesh Govindan and Hongsuda Tangmunarunkit. Heuristics for internet map discovery. In *Proc. IEEE INFOCOM*, Mars 2000.

[45] Bamba Gueye. Localisation géographique des hôtes dans l'internet basée sur la multilatération. Thèse de Doctorat, 2006.

[46] Bamba Gueye, Artur Ziviani, Mark Crovella, and Serge Fdida. Constraint-based geolocation of internet hosts. In *Proc. ACM SIGCOMM*, 2004.

[47] Jean-Loup Guillaume and Matthieu Latapy. Bipartite structure of all complex networks. *Information Processing Letters*, 90, 2004.

[48] Jean-Loup Guillaume and Matthieu Latapy. Relevance of massively distributed explorations of the internet topology : Simulation results. In *Proc. IEEE INFOCOM*, 2005.

[49] Jean-Loup Guillaume, Matthieu Latapy, and Clémence Magnien. Comparison of failures and attacks on random and scale-free networks. In *Proc. International Conference on Principles Of Distributed Systems*, 2004.

[50] S. Louis Hakimi. On the realizability of a set of integers as degrees of the vertices of a linear graph. *SIAM Journal on Applied Mathematics*, 10(3) :496–506, 1962.

[51] Václav Havel. A remark on the existence of finite graphs. *Caposis Pest. Matematika*, 80 :496–506, 1955.

[52] Urs Hengartner, Sue B. Moon, Richard Mortier, and Christophe Diot. Detection and analysis of routing loops in packet traces. In *Proc. ACM SIGCOMM Internet Measurement Workshop*, Novembre 2002.

[53] Monika Rauch Henzinger and Valerie King. Randomized fully dynamic graph algorithms with polylogarithmic time per operation. *Journal of the ACM*, 46(4) :502–516, Juillet 1999.

[54] Jacob Holm, Kristian de Lichtenberg, and Mikkel Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. In *Proc. ACM STOC*, 1998.

[55] Bradley Huffaker, Daniel Plummer, David Moore, and k claffy. Topology discovery by active probing. In *Proc. Symposium on Applications and the Internet*, Janvier 2002.

[56] IANA. RFC 3330 : Special Use IPv4 Addresses, http://www.faqs.org/rfcs/rfc3330.html, Septembre 2002.

[57] Van Jacobson. Congestion avoidance and control. In *Proc. ACM SIGCOMM*, Août 1988.

[58] Van Jacobson. traceroute, ftp://ftp.ee.lbl.gov/traceroute.tar.gz, Février 1989.

[59] Mark Jerrum and Alistair Sinclair. Approximating the permanent. *SIAM Journal of Computing*, 18(6) :1149–1178, 1989.

[60] Juniper. Configuring load-balance per-packet action. From the JUNOS 7.0 Policy Framework Configuration Guideline, see http://www.juniper.net/techpubs/software/junos/junos70/swconfig70-policy/html/policy-actions-config11.html.

[61] Jon Kleinberg. The Small-World Phenomenon : An Algorithmic Perspective. In *Proc. ACM STOC*, 2000.

[62] Balachander Krishnamurthy and Jia Wang. Topology modeling via cluster graphs. In *Proc. ACM SIGCOMM Internet Measurement Workshop*, Novembre 2001.

[63] Maciej Kurant. Répertoire en ligne de fichiers d'associations IP-AS, http://icawww.epfl.ch/kurant/IP2AS/.

[64] Anukool Lakhina, John W. Byers, Mark Crovella, and Peng Xie. Sampling biases in IP topology measurements. In *Proc. IEEE INFOCOM*, 2003.

[65] Matthieu Latapy and Clémence Magnien. Measuring fundamental properties of real-world complex networks. *submitted*, 2006.

[66] John M. Lewis. On the complexity of the maximum subgraph problem. In *Proc. ACM STOC*, 1978.

[67] Damien Magoni and Mickaël Hoerdt. Internet core topology mapping and analysis. *Computer Communications*, 28(5) :494–506, Mars 2005.

[68] Priya Mahadevan, Dmitri Krioukov, Kevin Fall, and Amin Vahdat. Systematic topology analysis and generation using degree correlations. In *Proc. ACM SIGCOMM*, 2006.

[69] Zhuoqing Morley Mao, David Johnson, Jennifer Rexford, Jia Wang, and Randy Katz. Scalable and accurate identification of AS-level forwarding paths. In *Proc. IEEE INFOCOM*, Mars 2004.

[70] Zhuoqing Morley Mao, Jennifer Rexford, Jia Wang, and Randy Katz. Towards an accurate AS-level traceroute tool. In *Proc. ACM SIGCOMM*, Août 2003.

[71] Sergei Maslov, Kim Sneppen, and Alexei Zaliznyak. Pattern detection in complex networks : Correlation profile of the internet. *ArXiv Condensed Matter e-prints*, Mai 2002.

[72] Matthew Mathis and Jamshid Mahdavi. Forward acknowledgment : Refining tcp congestion control. In *Proc. ACM SIGCOMM*, Août 1996.

[73] Tony McGregor. An IP address to Autonomous System Number Converter, http://mna.nlanr.net/Software/IPAS.

[74] Geoffrey J. McLachlan and David A. Peel. *Finite Mixture Models*. Wiley & Sons, 2000.

[75] Ron Milo, Nadav Kashtan, Shalev Itzkovitz, Mark E. J. Newman, and Uri Alon. Uniform generation of random graphs with arbitrary degree sequences. *ArXiv Condensed Matter e-prints*, 2003.

[76] Michael Molloy and Bruce Reed. A critical point for random graphs with a given degree sequence. *Random Structures and Algorithms*, 6 :161–180, Mars 1995.

[77] Michael Molloy and Bruce Reed. The size of the giant component of a random graph with a given degree sequence. *Cambridge Journal of Combinatorics, Probability and Computing*, 7 :295–305, Septembre 1998.

[78] Tim Moors. Streamlining traceroute by estimating path lengths. In *Proc. IEEE Workshop on IP Operations and Management*, Octobre 2004.

[79] John Moy. RFC 2328 : OSPF Version 2, http://www.faqs.org/rfcs/rfc2328.html. RFC 2328, Avril 1998.

[80] Mark E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2) :167–256, 2003.

[81] Mark E. J. Newman, Steven H. Strogatz, and Duncan J. Watts. Random graphs with arbitrary degree distributions and their applications. *Physical Review E*, 64(2) :026118, Août 2001.

[82] Jae Dong Noh and Heiko Rieger. Random walks on complex networks. *Physical Review Letters*, 92(11) :118701, Mars 2004.

[83] Jean-Jacques Pansiot. Local and dynamic analysis of internet multicast router topology. *to appear in Annals of Telecommunications*, 2007.

[84] Jean-Jacques Pansiot and D. Grad. On routes and multicast trees in the internet. *ACM Computer Communication Review*, 28(1) :41–50, Janvier 1998.

[85] Romualdo Pastor-Satorras and Alessandro Vespignani. *Evolution and Structure of the Internet : A Statistical Physics Approach*. Cambridge University Press, 2004.

[86] Thomas Petermann and Paolo De Los Rios. Exploration of scale-free networks - do we measure the real exponents ? *The European Physical Journal B*, 38(2) :201–204, Mars 2004.

[87] Jon Postel. RFC 793 : Transmission Control Protocol (TCP) - spécification (traduction française) http://abcdrfc.free.fr/rfc-vf/rfc793.html.

[88] Jon Postel. RFC 791 : Internet Protocol (IP) - spécification (traduction française), http://abcdrfc.free.fr/rfc-vf/rfc791.html, Septembre 1981.

[89] Jon Postel. RFC 792 : Internet Control Message Protocol, http://www.faqs.org/rfcs/rfc792.html, Septembre 1981.

[90] Bruno Quoitin, Steve Uhlig, Cristel Pelsser, Louis Swinnen, and Olivier Bonaventure. Interdomain traffic engineering with BGP. *IEEE Communication Magazine*, 41(5), Mai 2003.

[91] Portail web des RFC (Request For Comments), http://www.rfc-editor.org/.

[92] RFC 2616 : Hypertext Transfer Protocol, http://tools.ietf.org/html/rfc2616, Juin 1999.

[93] RFC 4271 : A Border Gateway Protocol 4, http://www.faqs.org/rfcs/rfc4271.html.

[94] Liste (en ligne) des RFCs à propos de BGP, http://www.bgp4.as/rfc.

[95] Stefan Schmid and Roger Wattenhofer. Algorithmic models for sensor networks. In *Proc. IEEE International Parallel and Distributed Processing Symposium*, Avril 2006.

[96] Yuval Shavitt and Eran Shir. DIMES : Let the internet measure itself. *ACM Computer Communication Review*, 35(5) :71 – 74, Octobre 2005.

[97] Augustin Soule, Anukool Lakhina, Nina Taft, Konstantina Papagiannaki, Kave Salamatian, Antonio Nucci, Mark Crovella, and Christophe Diot. Traffic matrices : Balancing measurements, inference and modeling. In *Proc. ACM SIGMETRICS*, Juin 2005.

[98] Neil Spring, Mira Dontcheva, Maya Rodrig, and David Wetherall. How to resolve ip aliases. *UW CSE Technical Report*, 04-05-04, Mai 2004.

[99] Neil Spring, Ratul Mahajan, David Wetherall, and Thomas Anderson. Measuring ISP topologies with Rocketfuel. *IEEE/ACM Transactions On Networking*, 12(1) :2–16, Février 2004.

[100] Neil Spring, David Wetherall, and Tom Anderson. Scriptroute : A public internet measurement facility. In *Proc. USENIX Symposium on Internet Technologies and Systems*, 2002. see also http://www.cs.washington.edu/research/networking/scriptroute/.

[101] Alexandre O. Stauffer and Valmir C. Barbosa. A study of the edge-switching markov-chain method for the generation of random graphs. *ArXiv Computer Science e-prints*, 2006.

[102] W. Richard Stevens. *TCP/IP Illustrated, Volume 1 : The Protocols*, chapter 8, Traceroute Program. Addison-Wesley, 1994.

[103] R. Taylor. Constrained switchings in graphs. In *Proc. Australian Conference on Combinatorial Mathematics*, Août 1980.

[104] Steven K. Thompson. *Sampling*. Wiley & Sons, 1992.

[105] Mikkel Thorup. Near-optimal fully-dynamic graph connectivity. In *Proc. ACM STOC*, 2000.

[106] M. Toren. tcptraceroute, Avril 2001. See http://michael.toren.net/code/tcptraceroute/.

[107] Andras Veres and Miklos Boda. The chaotic nature of TCP congestion control. In *Proc. IEEE INFOCOM*, Mars 2000.

[108] Fabien Viger. Génération de graphes connexes aléatoires avec séquence de degrés donnée. Rapport de Stage de DEA, 2004.

[109] Fabien Viger, Alain Barrat, Luca Dall'Asta, Cun-Hui Zhang, and Eric D. Kolaczyk. Network inference from traceroute measurements : Internet topology 'species'. *ArXiv Computer Science e-prints*, 2005.

[110] Fabien Viger and Matthieu Latapy. Efficient and simple generation of random simple connected graphs with prescribed degree sequence. In *Proc. Computing and Combinatorics Conference*, Août 2005.

[111] Fabien Viger, Benjamin Orgogozo, Matthieu Latapy, and Damien Bobillot. Vers un radar pour l'Internet. In *Proc. Conference on Risks and Security of Internet and Systems*, Octobre 2005.

[112] Stanley Wasserman and Katherine Faust. *Social Networks Analysis : Methods and Applications*. Cambridge University Press, 1994.

[113] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393 :440–442, Juin 1998.

[114] Projet "Macroscopic Topology Measurements" de CAIDA, http://www.caida.org/analysis/topology/macroscopic/.

[115] Carte géographique mondiale des liens `IP` inter-cités, http://www.netdimes.org/maps/mapBrowser.html.

[116] Geobytes company, http://www.geobytes.com.

[117] MaxMind, service de géolocalisation des adresses `IP`, http://www.maxmind.com/app/ip-location.

[118] Netgeo, service de géolocalisation des adresses `IP`, http://www.netgeo.com/.

[119] Internet Movie Database, http://www.imdb.com/.

[120] Statistiques d'utilisation de l'internet par le site Internet World Stats.

[121] Topology project, electric engineering and computer science department, University of Michigan, http://topology.eecs.umich.edu.

[122] Projet DIMES pour la cartographie de l'Internet, http://www.netdimes.org.

[123] Internet mapping project at Lucent Bell Labs, http://www.cs.bell-labs.com/who/ches/map.

[124] Projet Route Views de l'Université d'Oregon, http://www.routeviews.org.

[125] SCAN project at the Information Sciences Institute http://www.isi.edu/div7/scan/.

[126] Projet Skitter pour la cartographie de l'Internet, http://www.caida.org/tools/measurement/skitter/.

[127] Projet Traceroute@Home, http://www.traceroute@home.net.

[128] Visualisation en ligne de la topologie du réseau académique américain Internet 2, http://weathermap.grnoc.iu.edu/abilene_jpg.html.

[129] Visualisation en ligne de la topologie du FAI Free, http://support.free.fr/reseau/.

[130] Topologie du réseau Géant 2 d'interconnexion des réseaux académiques européens, http://www.geant2.net/upload/pdf/786_GN2_Topology_Nov_06-2.pdf.

[131] Visualisation en ligne de la topologie du réseau académique français RENATER, http://www.renater.fr/Metrologie/map-Renater4/.

[132] Jianhong Xia, Lixin Gao, and Teng Fei. Flooding attacks by exploiting persistent forwarding loops. In *Proc. ACM SIGCOMM Internet Measurement Conference*, Octobre 2005.

[133] Moshe Zukerman, Timothy D. Neame, and Ronald G. Addie. Internet traffic modeling and future technology implications. In *Proc. IEEE INFOCOM*, Avril 2003.

[134] http://www.liafa.jussieu.fr/~fabien/generation.

[135] http://www.paris-traceroute.net.