Optimisation des infrastructures réseaux

Frédéric Giroire

UNIVERSITÉ CÔTE D'AZUR

Habilitation à Diriger des Recherches

présentée pour obtenir le grade de
HDR
Spécialité : Informatique

—

Optimisation des infrastructures réseaux

Un peu de vert dans les réseaux et autres problèmes de placement et de gestion de ressources

Frédéric Giroire

—

Soutenue le 23 octobre 2018 devant le jury composé de :

Rapporteurs :	Thomas BONALD	Professeur Telecom ParisTech
	Hervé RIVANO	Professeur INSA Lyon
	Laurent VIENNOT	Directeur de recherche Inria
Examinateurs :	Vania CONAN	Ingénieur Thales
	Marcelo DIAS DE AMORIM	Directeur de recherche CNRS
	Brigitte JAUMARD	Professeur Concordia University
	Guillaume URVOY-KELLER	Professeur Université Côte d'Azur
	Djamal ZEGHLACHE	Professeur Télécom Sud Paris

Contents

Chapter 1

Introduction

Je présente dans ce document les travaux de recherche que j'ai effectués depuis ma soutenance de thèse en novembre 2006. Au cours de ces années, je me suis principalement intéressé à l'optimisation des infrastructures de réseaux et de stockage. Ma démarche a été d'utiliser des outils théoriques pour résoudre des problèmes posés par l'introduction de nouvelles technologies ou de nouvelles applications. Mes outils proviennent principalement de la combinatoire et en particulier de la théorie des graphes, de l'algorithmique, de l'optimisation et des probabilités. Quand j'ai pu proposer de nouvelles méthodes de résolution, j'ai ensuite essayé d'évaluer leur impact pratique par évaluation numérique, simulation ou expérimentation de scénarios réalistes. J'illustre ma démarche sur deux exemples qui correspondent aux deux parties de ce document, d'une part, l'étude de l'efficacité énergétique des réseaux et, d'autre part, l'étude des nouveaux réseaux logiciels virtualisés.

1.1 Motivation

Contexte

Le visage d'Internet a considérablement changé au cours des dernières années, d'une petite communauté de spécialistes avec des besoins de recherche et d'ingénierie à un espace avec des milliards d'utilisateurs qui l'utilisent dans leur vie quotidienne. Le nombre d'utilisateurs connectés a atteint 3,1 milliards en 2017 [425], soit un peu plus de la moitié de la population mondiale, quand ce nombre n'était que de 2 milliards en 2010 (30% de la population mondiale) et de 1 milliard en 2005 (16%). Dans le même temps, les usages changent et se multiplient : pénétration très forte des téléphones mobiles dans les dernières années, adoption par le grand public d'applications comme les réseaux sociaux, distribution quasi total de la vidéo par les réseaux IP, apparition et développement de l'internet des objets ou IoT en bref pour *Internet of Things*. Ce nombre croissant d'utilisateurs ayant de nouveaux besoins et de nouveaux moyens d'être connectés à Internet modifie les caractéristiques du réseau et la dynamique du trafic de la manière suivante :

- Augmentation forte du trafic réseau
  - trafic moyen : Selon le rapport de Cisco [132] qui établit des prévisions de trafic pour la période 2016-2021, le trafic IP mondial a quintuplé au cours des 5 dernières années, et triplera dans les 5 prochaines années. Dans l'ensemble, le trafic IP se développera à un taux de croissance annuel composé de 24% de 2016 à 2021. Le trafic IP annuel

moyen par utilisateur atteindra 35 GB en 2021 quand il n'était que de 13 GB en 2016. Cette augmentation du trafic est principalement due à la migration de la diffusion des médias grand public de hors ligne vers en ligne (par exemple, la télévision qui passe maintenant principalement en IP via les box, ou les jeux vidéo qui sont maintenant majoritairement en ligne) ou de diffusion en broadcast vers une diffusion unicast (par exemple, TV replay au lieu de diffusion en direct). En conséquence, la planification de la capacité pour le réseau cœur, c'est-à-dire le processus permettant d'assurer une bande passante suffisante, de sorte que l'accord de niveau de service du réseau cœur (*backbone network*) engagé puisse être satisfait, devient coûteuse, plus fréquente et plus difficile à optimiser pour les opérateurs réseaux.

- trafic pic : En plus de la croissance du trafic, la planification de la capacité doit prendre en compte la charge maximale du trafic. Cela se traduit souvent par une augmentation de la capacité des liens lorsqu'ils atteignent une utilisation moyenne de 50%, [99], ce qui entraîne une capacité coûteuse et sous-utilisée pendant les périodes de repos prolongées. Le trafic Internet aux heures d'affluence augmente plus rapidement que le trafic Internet moyen. Le trafic Internet à l'heure de pointe (à savoir la période de 60 minutes la plus chargée dans une journée) a augmenté de 51 % en 2016, comparativement à une croissance de 32 % du trafic moyen. Il augmentera d'un facteur 4,6 entre 2016 et 2021 pendant que le trafic Internet moyen augmentera d'un facteur 3,2 [132]. Cela accentue les problèmes de coûts des infrastructures réseaux peu utilisées en moyenne.

- Changement important de la nature du trafic (Vidéo/mobile/M2M) qui change la structure des réseaux

  - **Trafic Vidéo.** Un peu plus de 70% du trafic internet est maintenant du trafic vidéo [132]. Ce trafic a d'abord été transporté majoritairement en utilisant des solutions distribuées, comme le pair-à-pair (ou P2P en bref pour *Peer-to-Peer*). En 2008, BitTorrent était le plus grand générateur de trafic IP et représentait 31% du trafic total. Les solutions P2P ont ensuite été progressivement remplacées (BitTorrent ne représente plus que 3% du trafic mondial en 2016) par de grands systèmes de vidéos à la demande (comme Youtube ou Netflix) qui distribuent leurs vidéos à partir de centres de données (*data centers*). Ainsi, Netflix représente à lui tout seul 35.2% du trafic des réseaux filaires nord américain [154].
    → **Évolution de l'architecture réseau.** En raison de cette prépondérance du trafic vidéo, la distribution de celui-ci a un impact profond sur la structure des réseaux et la topologie même d'Internet. Avec le basculement des solutions P2P vers celles des centres de données, nous observons actuellement un trafic de plus en plus concentré vers et à partir de centres de données, alors que ce trafic était plus réparti auparavant. Cette concentration est de plus accentuée par le développement fort des services en nuage ou *cloud*. Pour des raisons d'efficacité, ces centres de données sont de plus en plus intégrés dans les réseaux, pour former des Réseaux de Distribution de Contenus ou CDN en bref pour *Content Delivery Networks*. Ces CDNs permettent de distribuer les vidéos à partir d'emplacements proches des utilisateurs, améliorant la qualité d'expérience de ces derniers. Les CDNs transportent actuellement déjà 52% du trafic Internet global, et Cisco estime que ce nombre montera à 71% en 2021.
  - **Développement mobile.** Le trafic généré par *smartphones* va bientôt dépasser celui des PC. En 2016, les PC représentaient 46% du trafic IP total, mais plus que 25% en

2021. Au contraire, les smartphones représenteront 33% en 2021, pour seulement 12% en 2016. Le trafic PC va augmenter à un taux de 10 %, quand les trafics TV, tablettes, téléphones et machine-à-machine vont avoir des taux de croissance respectifs de 21, 29, 49 et 49 %. Le trafic sans-fil et mobile représentera plus de 63% du trafic IP total en 2021. En 2016, les appareils filaires représentaient alors 51% du trafic IP.

→ **Mobilité des utilisateurs.** Ces changements profonds ont d'importantes conséquences. L'utilisation d'équipements mobiles comme les ordinateurs portables et les téléphones cellulaires devient généralisée et les opérateurs développent des applications multimédias pour les nouveaux smartphones. Leurs utilisateurs doivent être de plus en mesure d'utiliser ces applications sans subir de perturbations, même quand ils se déplacent. Faire en voiture des calculs d'itinéraires en temps réels prenant en compte la circulation ou regarder une vidéo en *streaming* dans un train sont par exemple devenues des applications courantes. Il faut pouvoir proposer ce service sans interruption de trafic.

– **La montée en puissance des services en nuage.** Les entreprises ont adopté avec enthousiasme les services en nuage publics et privés, ce qui a entraîné une croissance sans précédent de ces services. Ceux-ci sont caractérisées par une grande souplesse : les entreprises veulent pouvoir accéder à des applications (*Software as a Service*), plateformes (*Platform as a Service*) ou infrastructures (*Infrastructure as a Service*) à la carte. Ils permettent aux entreprises de faire des économies substantielles en n'ayant pas besoin de gérer eux-même ces services qui sont mutualisés au niveau des opérateurs de nuages et en pouvant adapter les ressources payées à leurs demandes. Cependant, ils introduisent de nouveaux défis : de sécurité, comme les données des entreprises peuvent être stockées et traitées dans des serveurs distants, de *dépendance à la qualité du réseau* et de *gestion élastique des ressources informatiques, de stockage et de réseau*. En particulier, ces deux derniers défis entrainent une *convergence des infrastructures cloud et réseaux* et obligent les opérateurs cloud à trouver de *nouvelles méthodes de gestion dynamique des cloud et réseaux*.

– **Développement de l'IoT et des communications machines-à-machine ou M2M en bref pour *Machine-to-Machine*** A la suite du développement des réseaux de capteurs, de nouveaux équipements se sont connectés au réseau : compteurs électriques, bornes de parking, réfrigérateurs, ... formant un IoT et l'apparition de grands systèmes connectés M2M, en particulier pour optimiser la production industrielle ou pour faire du diagnostic médical. Le nombre d'appareils connectés aux réseaux IP sera plus que trois fois la population mondiale en 2021. Cisco estime qu'il y aura 27,1 milliards d'objets connectés en 2021, à comparer au 17,1 milliards en 2016. Les connexions M2M vont croître d'un facteur 2,4, de 5.8 milliard en 2016 à 13.7 milliards en 2021. Il y aura 1,75 connexions M2M pour chaque humain en 2021. L'apparition de ces nouveaux appareils connectés change la façon dont le trafic est transporté par les réseaux. Nous avons déjà mentionné la part qui augmente du trafic sans-fil dans le trafic global. Il y a aussi des changements à des niveaux plus élevés dans le réseau. Par exemple, 35% du trafic devrait être distribué par les réseaux métro en 2021, contre 22% aujourd'hui.
→ **Forte pression sur les réseaux d'accès.** En raison du développement conjoint du trafic mobile et des mondes IoT et M2M, le trafic distribué par ou provenant des réseaux d'accès augmente fortement, ce qui engendre une pression sur les réseaux d'accès et une saturation des liens des réseaux d'accès. Ce déplacement des goulots

d'étranglement dans le réseau change la nature des problèmes d'optimisation à étudier.

  - **Réseaux sociaux.** Les réseaux sociaux sont maintenant utilisés massivement. Sur les 7,4 milliards d'habitants, 3,42 milliards sont internautes (46%) et 2,31 milliards sont actifs sur les réseaux sociaux (31% de la population mondiale). Facebook, par exemple, recense 1,86 milliards d'utilisateurs actifs (au moins un connexion dans le mois dernier). Le temps moyen passé sur les réseaux sociaux serait de 1h20 par jour en France.
    → **Forte variation du trafic des utilisateurs.** La diffusion vidéo d'événements populaires et le développement du partage social créent des communautés d'utilisateurs souhaitant accéder à un contenu multimédia. Par conséquent, un grand nombre d'utilisateurs peuvent essayer de se connecter à la même ressource en même temps de façon totalement imprévisible. Cela conduit à une charge de trafic très dynamique et à des pics de trafic importants.

- **Nouvelles contraintes énergétiques.** L'efficacité énergétique des infrastructures réseaux (que ce soit de télécommunications, de centres de données, de FAI ou d'entreprises) est une préoccupation croissante, à la fois à cause de la forte croissance de la demande, des coûts énergétiques croissants et des inquiétudes au sujet des émissions de CO2. Dans l'étude [331], il est rapporté que le secteur de l'Information et de la Communication (TIC) est responsable de 2 à 10% de la consommation mondiale d'énergie, dont 51% est attribué à l'infrastructure des réseaux de télécommunications et des centres de données. Les émissions de CO_2 des TIC augmentent à un taux de 6% par an. Avec un tel taux de croissance, ils pourraient représenter 12% des émissions mondiales d'ici 2020 [230]. Dans ce contexte, les centres de données et réseaux de base connaîtront les taux les plus élevés de consommation d'énergie de croissance dans les années à venir [265]. Par conséquent, un objectif très important est la réduction de la consommation d'énergie pour l'exploitation et la gestion des réseaux existants, en particulier avec le développement de nouvelles applications exigeantes en ressources.

Les évolutions des pratiques des utilisateurs et des solutions techniques font que les contraintes sur les réseaux changent rapidement. De plus les réseaux sont confrontés à de nouveaux défis sociétaux. Par conséquent, de nouvelles solutions doivent être inventées pour la conception et la gestion des réseaux afin de tenir compte de ces nouvelles contraintes et nouveaux besoins (applications de plus en plus exigeantes, par exemple en débit comme la vidéo haute résolution, en réactivité comme les jeux vidéos (*gaming*), grandes masses de données à traiter, ..., mobilité et demandes dynamiques).

Difficultés/Verrous.

Il existe deux difficultés majeures pour mettre en place les nouvelles solutions nécessaires.

- **Passage à l'échelle.** Comme discuté, les réseaux et le trafic réseau évoluent rapidement et leurs tailles augmentent constamment, en raison du développement de nouveaux services cloud, mobiles ou de l'internet des objets par exemple. Il est donc important de développer des méthodes (algorithmes, protocoles, ou outils d'optimisation) qui passent à l'échelle et qui puissent continuer à fonctionner avec cette augmentation planifiée.

- **Ossification d'Internet.** Aujourd'hui, malgré leur adoption généralisée, les réseaux traditionnels sont complexes et très difficiles à gérer. Il est difficile de les configurer selon des politiques prédéfinies, et encore plus de les reconfigurer pour répondre aux pannes ou à des variations rapides du trafic. Chaque équipement réseau a en effet son propre langage de configuration et la configuration est souvent faite à la main. Dans ces conditions, la politique suivie par les opérateurs est souvent de ne pas toucher à ce qui marche.

Evolution des paradigmes réseaux.

Pour répondre à ce développement fort du trafic réseau et à ces changements profonds de nature du trafic et des applications, les opérateurs et la communauté réseau se sont adaptés et ont proposé différentes solutions.

- **Architecture réseau. Distribué versus centralisé. P2P puis CDN.** Les premières réponses pour passer à l'échelle face à la demande croissante de partage de vidéo ont été le développement de solutions distribuées performantes. En particulier, les solutions P2P ont connu un succès exceptionnel dans les années 2000. Elles représentaient entre 43% et 70% du trafic mondial selon les régions. L'avantage des solutions P2P est qu'elles passent à l'échelle naturellement puisqu'un nouvel utilisateur avec de nouvelles demandes apporte aussi en même temps de nouvelles ressources de stockage et de bande passante. Cela permet de réduire le coût de la bande passante pour un diffuseur. Au cours de ces dernières années, je me suis intéressé à l'étude des systèmes P2P, et en particulier des systèmes de stockage distribués. Les solutions P2P ont ensuite progressivement été remplacées comme solution principale de diffusion vidéo. La distribution de vidéo se fait désormais principalement à partir du cloud et des CDNs. C'est le modèle de Youtube et Netflix par exemple qui représentent aujourd'hui à eux deux 50% du trafic réseau aux États-Unis.

- **Nouveaux protocoles.** La prépondérance du trafic vidéo a fait réfléchir les chercheurs sur de nouveaux protocoles. En effet, le fait que de nombreux utilisateurs demandent le même contenu dans des intervalles de temps rapprochés fait que le modèle classique serveur-client + routage IP n'est pas très efficace au sens où un même contenu est envoyé plusieurs fois sur les mêmes liens réseau. Le paradigme des réseaux centrés sur le contenu ou CCN en bref pour *Content Centric Networks* a été proposé pour remédier à cette inefficacité [309]. L'idée était de demander au réseau un contenu (et non de demander un contenu à une adresse IP) qui ensuite pouvait parvenir de différents endroits du réseau. En parallèle, chaque routeur ou commutateur réseau aurait une petite mémoire cache pour stocker certains contenus. Quand une demande parvient à un routeur, celui-ci regarde d'abord s'il a le contenu, si oui, il le retourne directement, sinon, il fait remonter la demande et garde dans une table d'intérêt l'information qu'on lui a demandé ce contenu. Quand le contenu repasse ensuite par lui, il sait le re-router et il peut décider de garder ce contenu en cas de nouvelle demande. Cela permet à terme de servir les contenus populaires d'endroits très proches des utilisateurs et de finalement faire une sorte de multicast avec des contenus qui ne sont pas regardés exactement en même temps. Ce paradigme n'a pas été vraiment mis en pratique en raison de difficultés intrinsèques (comment effectuer un routage par contenu avec un nombre de contenus énormes) et en raison de la difficulté de changer des protocoles réseaux qui marchent.

- **De nouveaux paradigmes: réseaux logiciels et virtualisation réseau.** Pour lutter contre l'ossification des réseaux de télécommunication, les réseaux logiciels ou SDN en bref pour *Software Defined Network* ont été proposés. Les architectures SDN séparent les fonctions de contrôle et de transmission des données du réseau. Cela permet de rendre directement programmable le contrôle du réseau et d'abstraire l'infrastructure sous-jacente des applications et des services réseau. Cette programmabilité permet de gérer dynamiquement le réseau et facilite l'introduction de nouveaux services qui sont presque impossibles à mettre en place dans les réseaux actuels. Dans le même temps, une autre révolution se met en place, la virtualisation des fonctions réseaux ou NFV en bref pour *Network Function Virtualization*. Similairement à ce qui est advenu pour les applications dans les centres de données qui sont exécutées dans des machines virtuelles, les services réseaux (pare-feu, optimisation du trafic, équilibrage de charge,...), au lieu d'être mis en œuvre par des équipements spécialisés, les *middleboxes*, peuvent maintenant être "virtualisés" et être exécutés par un logiciel s'exécutant dans une machine virtuelle. *Ces paradigmes se développent fortement et pénètrent l'industrie rapidement* en raison de leur nombreux avantages en termes de coût, flexibilité, efficacité énergétique : Google a déployé un réseau entre ses centres de données, nommé B4, en utilisant les technologies SDN. Cela lui a permis d'obtenir plusieurs avantages, notamment une gestion efficace du réseau, des cycles d'innovation des réseaux et des services plus faciles et plus rapides, une meilleure utilisation du réseau ainsi qu'une réduction des dépenses d'exploitation (OPEX) et des dépenses d'investissement (CAPEX) [218]. AT&T, le plus gros opérateur réseau américain, vise que 75% de ses fonctions réseau soient logicielles d'ici à 2020[1] et a déjà atteint 40% aujourd'hui [133]. Orange, le plus gros opérateur français, aussi a aussi introduit une offre SDN couvrant 75 pays, permettant à des entreprises d'instancier à la demande des services réseau [130]. Au cours des dernières années, Huawei a déployé 560 projets commerciaux SDN/NFV dans le monde entier [131].

Au cours de ces dernières années, j'ai participé à l'étude de ces nouvelles méthodes pour les évaluer et déterminer les façons les plus efficaces de les mettre en œuvre. En effet ces nouvelles technologies répondent à des besoins (passage à l'échelle pour le P2P, diminution de la redondance réseau du trafic video pour le CCN, programmabilité, baisse des coûts et prise en compte de la dynamicité pour le SDN et la virtualisation réseau), mais introduisent aussi une complexité et des problématiques nouvelles (gestion distribuée et nouveaux codages réseaux pour le P2P, problème de routage pour le CCN, unique point de défaillance pour le SDN, optimisation du placement de ressources virtuelles).

1.2 Résumé de mes travaux et résultats marquants

Je travaille principalement sur des problèmes d'algorithmique et de combinatoire avec pour champ d'application les réseaux de télécommunications. Typiquement un réseau de télécommunication est modélisé par un graphe. Un sommet peut représenter un processeur, un routeur, un équipement radio, un site ou une personne. Une arête correspond à une connexion entre les éléments représentés par les sommets (connexion logique ou physique). Des informations supplémentaires peuvent être associées aux sommets (nombre de ports, coût de l'équipement) ou aux arêtes (poids qui correspond à une longueur, un coût, une capacité). Des modèles divers peuvent être définis en fonction de l'application et cette modélisation est une tâche importante.

[1] http://about.att.com/innovation/sdn

Comme expliqué précédemment, mes recherches ont suivi l'évolution des technologies réseaux et se sont principalement organisées autour de trois axes thématiques : d'abord l'amélioration des performances énergétiques des réseaux et l'étude et la modélisation des systèmes de stockage pair-à-pair, puis plus récemment l'utilisation des nouveaux paradigmes des réseaux logiciels et de la virtualisation réseau pour mettre en pratique des protocoles et algorithmes efficaces. Tout au long de mes recherches et en parallèle de mes travaux pour des applications, j'ai aussi eu comme fil rouge l'approfondissement des outils utilisés dans les travaux de l'équipe, principalement la théorie des graphes. Je présente ici un résumé de mes travaux sur ces axes ainsi que sur un dernier domaine sur lequel j'ai moins longtemps travaillé, à savoir l'algorithmique pour la sécurité des réseaux en utilisant des analyse de traces.

Après ce résumé de mes travaux principaux, mon document s'organise autour de deux parties qui correspondent à deux focus sur mes travaux : l'un sur l'efficacité énergétique, dans la partie I, et l'autre sur les réseaux logiciels virtualisés, dans la partie II.

1.2.1 Efficacité énergétique (Partie I - Histoire verte)

Contexte. Ces dernières années, j'ai travaillé sur les thématiques d'économie d'énergie dans les réseaux. Ce travail s'est fait d'abord principalement au sein de l'ANR-JCJC Dimagreen[2] dont j'étais le porteur (2009-2012), puis du projet TREND, Towards Real Energy-efficient Network Design[3] dont Inria est devenue institution collaborative en 2011 et pour lequel j'étais responsable du côté Inria (2010-2013). L'objectif des projets a été d'introduire et d'analyser des methodes de conception et de gestion des réseaux économes en énergie dans le but de réduire la facture énergétique des télécommunications. Les projets étaient décomposés en trois tâches principales : l'évaluation de performance et campagnes de mesures, la conception de réseaux et la gestion de réseaux. J'ai principalement travaillé à proposer de nouveaux algorithmes de routage minimisant la consommation énergétique des réseaux cœur.

J'ai rencontré différents groupes industriels intéressés par le sujet dont Alcatel Lucent, Monaco Telecom et Orange. Cela nous a en particulier permis d'effectuer une campagne de mesures, en 2012, sur une plateforme d'Orange labs à Sophia Antipolis sur des outils de compression et d'optimisation TCP (Wide Area Network Optimization Controlers ou WOC en bref) qui a mené à un nouvel algorithme de routage efficace en énergie qui utilise l'élimination de redondance (Section 3.2) ou d'étudier comment effectuer une distribution de contenus efficace en énergie sur la topologie du réseau d'Orange et avec des données de trafic réalistes (Section 3.3).

Depuis récemment, nous étudions l'utilisation du nouveau paradigme des réseaux logiciels et de la virtualisation réseau dans le but de mettre en pratique certaines méthodes proposées pour économiser l'énergie.

Pour traiter ce sujet, j'ai utilisé des techniques diverses allant donc des mesures à des outils de théories des graphes, en passant par l'algorithmique et l'optimisation combinatoire à base de programmation linéaire entière.

Plan. La première partie de ce document décrit comment se sont déroulés mes travaux au cours de ces dernières années. Dans un premier temps, nous avons étudié le problème de base du routage efficace en énergie ou EAR en bref pour *Energy Aware Routing*. Nous avons étudié différents scénarios pratiques correspondant aux différentes sections du chapitre 3. Le premier scénario pratique était d'estimer combien d'énergie pouvait être économisée dans les réseaux des FAI tout en conservant de bonnes propriétés en terme de longueur des routes et de tolérance

[2] http://www-sop.inria.fr/mascotte/DIMAGREEN/wiki/
[3] http://www.fp7-trend.eu/

aux pannes en section 3.1. Le deuxième scénario étudie le cas où des outils d'élimination de redondance pour le trafic sont utilisés, voir la section 3.2. Enfin, le troisième et dernier scénario, section 3.3, est l'étude de la distribution de contenus, en particulier vidéo. Ces problèmes pratiques ont soulevé des problèmes fondamentaux de théorie des graphes, en particulier, la construction de sous-graphes avec peu d'arêtes et tolérant les demandes de trafic. Nous exposons nos travaux sur cette question dans le chapitre 4. Enfin, comme les opérateurs réseau sont réticents à mettre en place les solutions efficaces en énergie proposées, nous nous sommes attelés à voir comment les mettre en pratique en utilisant les nouveaux paradigmes des réseaux logiciels et de la virtualisation réseau dans le chapitre 8. En effet, la séparation du plan de contrôle dans les réseaux logiciels permet de mettre en place les changements rapides de configuration réseau qui sont indispensables pour un usage économe en énergie.

De premiers résultats (chapitre 3).

Routage efficace en énergie ou éteindre des équipements (section 3.1 du chapitre 3). Des études montrent que la charge en trafic des routeurs n'a qu'une faible influence sur leur consommation énergétique. Par conséquent, la consommation dans les réseaux est fortement liée au nombre d'équipements du réseau activés (interfaces, châssis, etc). Dans un objectif de minimisation de l'énergie dans les réseaux, il est intéressant de *minimiser le nombre (pondéré) d'équipements utilisés lors du routage, puis de mettre en veille ceux qui ne le sont pas. C'est le principe de l'EAR.* Comme la charge de trafic n'a qu'une petite influence sur la consommation des routeurs, le principe de l'EAR est de mettre en mode veille des équipements réseaux (liens réseau) qui ne sont pas utilisés.

Dans nos travaux initiaux [Ci56, ch5], nous avons considéré une architecture simplifiée où un lien entre deux routeurs relie deux interfaces. Quand un lien n'est pas activé, les deux interfaces correspondantes peuvent être éteintes. Par conséquent, afin de réduire la consommation d'énergie, l'objectif est de trouver un routage qui minimise le nombre de liens utilisés et satisfait toutes les demandes. En condition normale d'utilisation, le gain en énergie pour les interfaces des réseaux cœur est de l'ordre de 30%. Nous étudions enfin l'impact de ces solutions efficaces en énergie sur la tolérance aux pannes et sur la longueur moyenne des routes.

Élimination de redondance (section 3.2 du chapitre 3). Pour économiser de l'énergie, l'idée est donc d'agréger les demandes sur un sous-ensemble des liens du réseau pour pouvoir éteindre les autres liens. Nous avons aussi étudié l'utilisation d'une technique complémentaire, l'élimination de redondance des données (RE). Cette technique, mise en place au niveau des routeurs, permet d'identifier et d'éliminer le contenu redondant au cours des transferts réseaux. Ainsi, la capacité des liens est virtuellement accrue et les demandes peuvent être agrégées de nouveau. Pour évaluer cette technique, nous avons conduit des expérimentations en collaboration avec Orange Labs sur une de leur plateforme. Nous avons estimé le coût énergétique pour mettre en place la compression et le gain potentiel en quantité de trafic. Il y a donc un compromis entre les deux et il est important de déterminer à quels endroits du réseau mettre en place la RE. Pour ce faire, nous avons modélisé le problème sous forme de programmes linéaires et proposé des méthodes heuristiques pour le résoudre dans [Ci50, J17]. Les simulations sur un jeu de topologies de réseaux montrent que cette technique permet d'économiser 30% supplémentaires de la consommation énergétique des liens.

Distribution de contenus (section 3.3 du chapitre 3). Comme discuté ci-dessus, la distribution de contenus, et en particulier de vidéos, représente l'essentiel du trafic. Le trafic réseau qu'elle engendre a des caractéristiques particulières et est très répétitif. Il a donc été proposé de rajouter des caches à l'intérieur des nœuds réseaux afin d'améliorer la distribution de contenus

et de réduire la congestion des réseaux. Cela a aussi l'avantage de diminuer leur consommation énergétique. Nous présentons deux travaux dans ce contexte.

Dans la section 3.3.1, nous étudions l'impact de l'utilisation de caches réseaux (in-network caches) et de leur coopération avec les CDN sur l'énergie consommée par le routage. Nous modélisons ce problème par un programme linéaire en nombres entiers et proposons une heuristique en temps polynomial pour le résoudre efficacement. L'objectif est de trouver un routage réalisable qui minimise la consommation énergétique du réseau tout en satisfaisant les demandes de contenus. Nous exhibons les valeurs des paramètres (tailles des caches, popularités des données, ...) pour lesquelles ces caches sont utiles. Des expérimentations montrent qu'en plaçant un cache sur chaque routeur d'un réseau cœur pour stocker le contenu le plus populaire, ainsi qu'en choisissant le meilleur serveur pour chaque demande traitée par un CDN, environ 20% de l'énergie du cœur peuvent être économisés, dont 16% du gain sont dus aux seuls caches [Ci49, J15].

Dans la section 3.3.2, nous étudions comment réduire la consommation énergétique du réseau d'un fournisseur d'accès internet en déterminant la meilleure infrastructure possible de diffusion. Nous proposons un algorithme pour décider de façon optimale où stocker les contenus (caching) à l'intérieur du réseau de l'opérateur [Ci45]. Nous évaluons notre solution avec deux études de cas basées sur les informations de deux opérateurs. Nos résultats montrent que la conception de l'infrastructure de distribution de contenus selon des critères d'efficacité énergétique apporte des économies substantielles, à la fois en terme d'énergie et de bande passante nécessaire pour le point de peering de l'opérateur. De plus, nous avons étudié l'impact des caractéristiques du contenu et de différents modèles de consommation énergétique. Enfin, nous donnons des indications pour la conception des réseaux du futur, efficaces en énergie.

Outils théoriques et sous-graphes minimaux (chapitre 4). La minimisation du nombre de liens utilisés dans un réseau tout en satisfaisant la charge nous a amené à définir des problèmes de théorie des graphes. Ainsi, nous avons étudié le problème de déterminer des sous-graphes minimaux en nombre d'arêtes qui permettent de satisfaire les demandes pour des graphes classiques tels que arbres, cycles, grilles, graphes aléatoires. L'intérêt est double. Certains réseaux se rapprochent de graphes classiques, par exemple, certains réseaux d'accès sont organisés en grilles carrées. D'autre part, cette étude donne une idée de comment le problème se comporte et de combien on peut espérer gagner, ou, du moins, de bonnes bornes sur les performances à attendre.

L'indice de transmission (forwarding index) d'un graphe est le minimum, sur tous les routages possibles de toutes les demandes, de la charge maximale d'une arête. Cette métrique est d'un grand intérêt puisqu'elle capture la notion de congestion globale de manière précise : moins l'indice de transmission est élevé, moins la congestion est importante. Je suis donc parti à la recherche de (sous-)graphes couvrants avec un nombre fixé d'arêtes qui ont un indice de transmission minimum. J'ai effectué deux études qui correspondent aux deux sections de ce chapitre. Dans la section 4.2, nous avons déterminé les sous-graphes couvrants d'une grille carrée avec un indice de transmission minimum [Ci40, J13]. Dans la section 4.3, nous étudions la question de conception suivante [Ci39] : étant donné un nombre e d'arêtes et un nombre n de sommets, quels sont les graphes avec le plus petit indice de transmission que nous pouvons construire ?

Aller vers la mise en pratique (chapitre 8). Mon objectif ici est d'explorer le potentiel des nouveaux paradigmes réseaux comme les réseaux logiciels et la virtualisation réseau pour concevoir des solutions efficaces en énergie pour les réseaux de télécommunication et de centres

de données.

Efficacité énergétique et réseaux logiciels (section 8.2 du chapitre 8). Ces dernières années, de nombreuses applications ont été construites en utilisant SDN, comme par exemple, l'équilibrage de charge de serveurs, la migration de machines virtuelles, de l'ingénierie du trafic et du contrôle d'accès. Je me suis intéressé à l'utilisation de SDN pour mettre en place l'EAR. L'avantage de SDN est de pouvoir faire des collectes de matrices de trafic en temps réel qui permettent de calculer des solutions de routage minimum en terme de consommation énergétique, puis de pouvoir changer dynamiquement les règles de routage des commutateurs. Cependant, les travaux précédents sur l'EAR ont supposé que la table de routage d'OpenFlow [420] (le protocole de communication utilisé principalement pour contrôler un routeur dans les solutions SDN) pouvait stocker un nombre infini de règles. En pratique, cette hypothèse n'est pas vraie. En effet, la table de routage est implémentée avec de la mémoire TCAM (pour *Ternary Content Addressable Memory*) qui est à la fois chère, gourmande en énergie et de taille limitée. J'ai proposé une méthode d'optimisation pour minimiser la consommation énergétique d'un réseau cœur tout en respectant les contraintes d'espace mémoire disponible pour les règles de routage. Je montre qu'il est ainsi possible d'économiser presque autant d'énergie quand le nombre de règles n'est pas limité [Ci43, J10].

Virtualisation pour l'efficacité énergétique (section 8.3 du chapitre 8). La virtualisation des fonctions réseaux permet d'implémenter les services réseaux à la demande et de s'adapter aux changements du trafic. Couplée au SDN, elle permet une grande flexibilité pour gérer les flots et permet de réduire les coûts et, en particulier, les coûts énergétiques. Dans [Ci33, J6], j'ai proposé un modèle de décomposition résolu ensuite par génération de colonnes pour effectuer une optimisation jointe de l'EAR et du placement de fonctions réseaux virtuelles.

Mise en pratique (section 8.4 du chapitre 8). J'ai enfin étudié la mise en pratique de l'EAR. Tout d'abord, j'ai considéré l'introduction progressive des technologies SDN dans des réseaux SDN hybrides. J'ai proposé différents mécanismes pour ne pas introduire de pertes de paquets lors des changements de routage adaptant la consommation énergétique du réseau à la demande [Ci35, J7]. Les solutions proposées ont été testées par émulation sur des plateformes logicielles 8.4.3 et matérielles en section 6.7. J'ai pu vérifier que l'on *pouvait mettre en œuvre des politiques de routage dynamiques efficaces en énergie sans augmenter significativement les délais ni les pertes de paquets.*

Collaborations. La distribution de contenus efficace en énergie a été le sujet de la thèse de Remigiusz Modrzejewski et l'efficacité énergétique avec les réseaux logiciels, celui de la thèse de Nicolas Huin. J'ai aussi eu la chance de pouvoir compter sur l'aide de mes co-auteurs : J. Araujo, *E. Bonetto, L. Chiaraviglio, R. Gonzalez, C. Guerrero, E. Le Rouzic*, J. Moulierac, *F. Musumeci*, Y. Liu, R. Modrzejewski, S. Pérennes, T. K. Phan, *F. Roudaut*, I. Tahiri, *A. Bianco, F. Idzikowski, F. Jimenez, C. Lange, J. Montalvo, A. Pattavina A. Valenti, W. Van Heddeghem, and Y. Ye.*

1.2.2 Réseaux logiciels virtualisés (Partie II - Histoire virtuelle)

Aujourd'hui, malgré leur adoption généralisée, les réseaux traditionnels sont complexes et très difficiles à gérer. Il est difficile de les configurer selon des politiques prédéfinies, et de les reconfigurer pour répondre aux pannes et à la dynamique du trafic. Pour empirer les choses, les réseaux actuels sont également intégrés verticalement : les plans de contrôle et de données sont regroupés. SDN est un paradigme émergent qui promet de changer cet état des choses,

en brisant l'intégration verticale, avec *(i)* le découplage des plans de contrôle et de données, *(ii)* un contrôleur qui a une vue centralisée du réseau, *(iii)* l'ouverture des interfaces entre les équipements du plan de contrôle (contrôleurs) et ceux du plan de données, et *(iv)* la programmabilité du réseau par des applications externes [210].

Dans le même temps que l'apparition des réseaux logiciels, nous assistons à une seconde révolution majeure, la virtualisation des réseaux, et en particulier la virtualisation de fonctions réseaux (NFV). Le paradigme NFV est une approche émergente dans laquelle les fonctions réseau ne sont plus exécutées sur du matériel dédié appelé *middlebox*, mais peuvent être exécutées sur des serveurs génériques situés dans de petits centres de données [174]. Les exemples de fonctions réseau comprennent les pare-feu (*firewall*), l'équilibrage de charge (*load balancing*), le filtrage du contenu et l'inspection approfondie des paquets (*deep packet inspection*). Cette technologie vise à traiter les principaux problèmes de l'infrastructure *middlebox* des entreprises d'aujourd'hui, tels que le coût, la rigidité des capacités, la complexité de la gestion et les défaillances [237]. L'un des principaux avantages de cette approche est que les fonctions réseau virtuelles (ou VNF en bref pour *Virtual Network Functions*) peuvent être instanciées et mises à l'échelle à la demande sans qu'il soit nécessaire d'installer de nouveaux équipements.

L'apparition de ces deux paradigmes pose de nombreux nouveaux problèmes de protocoles, d'algorithmique et d'optimisation : routage dynamique, placement de ressources virtuelles, optimisation conjointe du routage et du placement de ressources, tolérances aux pannes des ressources virtuelles, ... Dans ce contexte, j'ai étudié différents problèmes pour les réseaux logiciels virtualisés : la compression des tables de routage SDN multi-dimensionnelles dans le chapitre 6 et le placement de chaînes de fonctions réseau virtuelles dans le chapitre 7.

Compresser des tables de routages SDN multi-dimensionnelles (Chapitre 6). Les règles de routage utilisées dans les réseaux SDN sont plus complexes que celles des réseaux classiques. Dans OpenFlow 1.3, le routage d'un paquet peut s'effectuer en utilisant jusqu'à 40 champs différents. La complexité de ces règles nécessite l'utilisation de Ternary Content Adressable Memory (TCAM) qui est malheureusement plus chère et plus gourmande en énergie que le type de mémoire utilisé sur les réseaux classiques. La taille des tables de routage pouvant être stockées s'en retrouve alors grandement réduite (de l'ordre de 750 à 4000 règles [423]). Cette limitation est une contrainte importante pour le déploiement des réseaux logiciels.

Ce problème a été attaqué dans la littérature, comme discuté en section 6.2, en utilisant différentes stratégies, comme la compression de tables de routages [196, 173] ou la distribution des règles de routage [200].

Dans le chapitre 6, nous examinons une méthodologie de compression dans laquelle n'importe quel champ de l'en-tête d'un paquet peut être compressé. C'est une avancée importante car cela permet une compression plus efficace des tables de routage et permet de mettre en œuvre des politiques de routage avancées, comme l'équilibrage de charge (load balancing) et/ou des politiques de qualité de services. Dans la suite, nous considérons la compression de règles pour deux champs, les sources et destinations. Mais, notre solution peut cependant être aussi appliquée sur d'autres champs tels que le ToS (Type de Service), le protocole de transport, ... et peut se généraliser à un nombre plus élevé de champs.

Nous étudions d'abord le problème théorique de compresser les tables de routage SDN en section 6.3. La nouveauté est que ces tables sont multi-dimensionnelles et la difficulté vient du fait que l'ordre dans lequel les règles sont écrites devient crucial dans ce contexte. Nous avons d'abord déterminé la complexité du problème, répondant ainsi à une question ouverte de [367]. Nous avons ensuite proposé des algorithmes d'approximation, puis des algorithmes à complexité paramétrée fixe (FPT) pour résoudre le problème [Ci42, J9, 146].

Nous présentons ensuite MINNIE, une solution de routage SDN qui utilise la compression de règles en section 6.6. Puisque le problème de compression de table est NP-Complet, MINNIE implémente une heuristique de compression qui crée trois tables compressées différentes, utilisant seulement l'agrégation par source pour la première, par destination pour la deuxième, ou la règle par défaut pour la dernière, et choisit la plus petite des trois tables. Cette heuristique est une 3-approximation du problème de compression avec ordre (Section 6.5.1). Puis MINNIE utilise une heuristique de routage où la métrique des poids dépend de l'utilisation des liens et de la taille des tables. Nous présentons les résultats de MINNIE obtenus par simulation sur plusieurs *topologies de centres de données* parmi les plus communes. Nous montrons que notre solution passe à l'échelle et peut *gérer plus d'un million de flots différents avec moins de 1000 entrées* dans sa table de routage et avec un temps de compression négligeable.

Enfin, nous avons utilisé une plateforme matérielle pour tester à la fois nos algorithmes de compression et la possibilité pour les réseaux SDN d'implémenter efficacement des algorithmes d'optimisation. Cette plateforme nous a permis d'émuler un *4-fat tree*, l'une des architectures de centre de données les plus courantes et de tester la mise en pratique des technologies logicielles. En effet, le fait de centraliser les décisions de routage en plaçant un algorithme d'optimisation dans le contrôleur pourrait apporter divers problèmes : augmentation du délai en raison de potentiels contacts avec le contrôleur (en particulier pour le premier paquet d'un flot), saturation du CPU ou des liens avec le contrôleur, augmentation du taux de perte, ... Les résultats de la section 6.7 montrent que notre solution est capable de minimiser le nombre d'entrées dans les commutateurs, *tout en gérant avec succès la dynamique des requêtes et en maintenant la stabilité des réseaux.*

Placement optimal de chaînes de services réseaux (Chapitre 7). Le modèle des réseaux programmables virtualisés permet aux opérateurs de télécommunications d'offrir des services réseaux complexes et flexibles. Un service se modélise alors comme une chaîne de fonctions réseaux (firewall, compression, contrôle parental,...) qui doivent être appliquées séquentiellement à un flot de données. Ces chaînes sont appelées chaînes de fonctions de services ou SFC en bref pour *Service Function Chains*. Le problème qui se pose alors est comment router les demandes et placer les fonctions réseau virtuelles des chaînes de services par lesquelles doivent passer les demandes. Les fonctions objectifs peuvent être multiples :

Dans un premier travail (Section 7.1), nous essayons de trouver le meilleur compromis entre l'utilisation de la bande passante et le nombre d'emplacements pour héberger les fonctions réseau. Nous proposons un modèle de génération de colonnes pour le routage et le placement de chaînes de service. Nous sommes les premiers à proposer un modèle exact qui passe à l'échelle. Nous montrons au travers d'expérimentations poussées que nous pouvons résoudre le problème de façon optimale en moins d'une minute pour des réseaux ayant une taille allant jusqu'à 65 nœuds et 16 000 requêtes. Nous étudions aussi le compromis entre l'utilisation de la bande passante et le nombre de nœuds capables d'héberger des fonctions réseau [Ci36, J8]. Finalement, nous avons étudié la tolérance aux pannes de ces systèmes [Ci30].

Dans un second travail (Section 7.2), nous étudions le problème du placement de fonctions de services qui consiste à déterminer sur quels nœuds localiser les fonctions afin de satisfaire toutes les demandes de service, de façon à minimiser le coût de déploiement. Nous montrons que le problème peut être ramené à un problème de Set Cover, même dans le cas de séquences ordonnées de fonctions réseau. Cela nous permet de proposer deux algorithmes d'approximation à facteur logarithmique, ce qui est le meilleur facteur possible. Finalement, nous évaluons les performances de nos algorithmes par simulations. Nous montrons ainsi qu'en pratique, des solutions presque optimales peuvent être trouvées avec notre approche [Ci32].

Collaboration. Ce sujet est celui des thèses d'Andrea Tomassilli et de Giuseppe di Lena. Mes co-auteurs ont été les suivants : F. Havet, N. Huin, *B. Jaumard*, *D. Lopez-Pacheco*, J. Moulierac, S. Pérennes, *M. Rifai*, A. Tomassilli, *G. Urvoy-Keller*.

1.2.3 Étude des systèmes distribués (Travaux annexes non présentés dans l'HdR)

Ces dernières années, j'ai travaillé sur l'analyse et la conception de systèmes distribués, principalement sur les systèmes de stockage, puis plus récemment sur les systèmes de diffusion de vidéos (streaming).

Systèmes de stockage pair-à-pair

Je me suis intéressé aux systèmes de stockage pair-à-pair au sein de l'ANR SPREADS[4] 2007-2010 et du projet IST/FET AEOLUS 2005-2010[5]. Ces systèmes sont économiques à opérer mais leur nature hautement distribuée pose des questions sur leur fiabilité, la disponibilité des données et leur confidentialité. Plusieurs efforts ont été faits pour construire des systèmes distribués auto-régulants à grandes échelles. Cependant, peu de modèles analytiques ont été proposés pour estimer le comportement du système (durée de vie des données, utilisation de ressources, par exemple bande passante) et pour comprendre les compromis entre les paramètres du système. En collaboration avec Stéphane Pérennes et Julian Monteiro principalement, j'ai proposé de nouveaux modèles pour analyser ces systèmes.

Les systèmes de stockage de données pair-à-pair à grande échelle sont depuis longtemps envisagés, du moins en théorie, comme un moyen d'apporter un stockage hautement fiable à faible coût. Pour atteindre une forte fiabilité, ces systèmes pair-à-pair codent les données des utilisateurs par un ensemble de fragments redondants et les distribuent sur les pairs (voir Figure 1.1). Cette redondance doit être constamment surveillée et maintenue par un processus de reconstruction en raison d'occurrences continues de pannes ou de départs de pairs. Les performances du système dépendent de nombreux paramètres qui doivent être bien réglés, comme le facteur de redondance, le code utilisé, la fréquence de la réparation de données et la taille des blocs de données. Ces paramètres ont un impact sur la quantité de ressources (la bande passante, l'espace de stockage, ...) nécessaires pour obtenir une certaine fiabilité (probabilité de perdre des données).

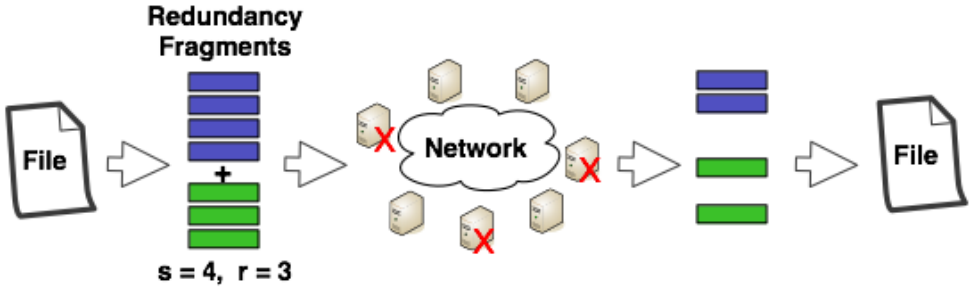


Figure 1.1: Les fichiers ou données sont découpés en blocs de données. Chaque bloc de donnée est ensuite divisé en s fragments initiaux auxquels sont ajoutés r fragments de redondance. Chaque groupe de s fragments parmi les $s + r$ permet de reconstruire le bloc de données d'origine.

[4] http://spreads.fr/
[5] http://aeolus.ceid.upatras.gr/

Nous avons étudié et utilisé différentes techniques pour analyser et prévoir les performances des systèmes de stockage à grande échelle. Elles vont de l'analyse formelle (chaînes de Markov et modèles fluides) à des simulations et expérimentations (en utilisant la plateforme Grid5000). En comparant à des simulations, nous avons montré que les modèles à base de chaînes de Markov donnent une approximation correcte du comportement moyen des systèmes, mais ne peuvent capturer leurs variations au cours du temps. Notre contribution principale est un nouveau modèle stochastique basé sur une approximation fluide qui capture les variations du système quand les autres modèles de la littérature donnent seulement des indications sur le comportement moyen du système. Nous avons aussi utilisé des modèles de files d'attente pour estimer le temps de reconstruction quand la bande passante est très limitée. Additionnellement, nous avons étudié différentes méthodes pour distribuer les fragments chez les pairs (stratégies de placement) et un code hybride qui réduit la consommation de bande passante. Enfin, nous avons conduit des expérimentations avec le code de la startup Ubistorage[6] en utilisant la plateforme Grid5000 pour valider nos résultats.

Comportement moyen et guide pratique. Dans [Ci55], nous modélisons un système de stockage en utilisant un modèle Markovien. Ce modèle nous permet de prendre en compte l'effet des pannes de disques ainsi que le temps pris par le processus d'auto-réparation. Nous confirmons que la stratégie de réparation paresseuse peut être employée pour amortir le coût de réparation en bande passante. Nous avons ensuite déduit des formules mathématiques closes qui estiment le comportement moyen du système. Ces formules donnent une bonne intuition de sa dynamique. Notre contribution est un guide pratique pour les concepteurs et administrateurs de systèmes de stockage pour choisir le bon jeu de paramètres.

Capturer les variations. Dans [Cn78, Ci60], nous proposons et étudions un modèle fluide qui estime les variations de la consommation en bande passante et la probabilité de perdre des données. Ces variations sont causées par la perte simultanée d'un grand nombre de blocs de données quand un pair subit une panne ou quitte le système.

Distribution du temps de réparation. La vitesse de réparation d'un bloc de donnée est cruciale pour sa survie. Cette vitesse est déterminée principalement par la quantité de bande passante disponible. Les reconstructions concurrentes sont en compétition pour cette bande passante. Dans [Ci48], nous proposons un nouveau cadre analytique qui prend en compte cette corrélation. Principalement, nous introduisons un modèle de file d'attente dans lequel les reconstructions sont effectuées par les pairs à une cadence qui dépend de la bande passante disponible. Notre modèle permet d'obtenir une estimation précise de la probabilité de perdre des données quand la bande passante est limitée. En outre, nous étudions l'efficacité de différentes politiques d'ordonnancement pour les reconstructions.

Stratégies de placement de données. Dans [J20] nous avons étudié l'impact de différentes stratégies de placement de données sur les performances du système. Cette étude est motivée par les systèmes de stockage pair-à-pair qui stockent les données dans les voisins logiques (par exemple, leurs voisins dans une table de hachage distribuée ou DHT en bref pour *Distributed Hash Table*). Nous avons utilisé des simulations et des modèles combinatoires pour montrer que, sans contrainte de ressources, le comportement moyen est le même quelle que soit la politique de placement utilisée. Cependant, les variations d'utilisation de la bande passante sont beaucoup plus fortes pour des politiques locales (pour lesquelles les données sont stockées sur les voisins logiques). Quand la bande passante est limitée, ces fortes variations entrainent un temps de réparation beaucoup plus long et donc réduisent la durée de vie des données.

[6] http://www.ubistorage.com/

Codage hybride. Le processus de reconstruction des systèmes de stockage utilisant les codes d'effacement (*erasure code*) comme Reed-Solomon consomme un surplus de bande passante (en comparaison avec une simple réplication des données). Dans [Ci52], nous étudions des codes hybrides qui mixent les codes d'effacement et la réplication. L'idée est de garder dans le système un réplica des données en même temps que les fragments des codes d'effacement. Nous avons modélisé ces systèmes en utilisant des chaînes de Markov, puis trouvé des formules closes pour approximer l'utilisation de bande passante et la durée de vie des données. Nous avons montré que les systèmes hybrides ont un meilleur compromis que les systèmes simples entre bande passante, utilisation d'espace disque et durée de vie.

Systèmes live de diffusion de vidéos pair-à-pair

Dans ces systèmes, une source diffuse une vidéo à un ensemble d'utilisateurs qui veulent assister à un événement en temps réel. La diffusion de vidéo peut être faite au travers d'une architecture classique client/serveur ou avec un architecture distribuée, par exemple pair-à-pair. Les solutions distribuées sont très efficaces pour les scénarios de diffusion en direct dans lesquels les utilisateurs regardent la vidéo au même moment. En effet, la bande passante de ces utilisateurs peut être utilisée pour transférer la vidéo aux autres utilisateurs, diminuant la charge de la source.

Les réseaux pair-à-pair sont de deux types, avec un réseau logique (overlay) structuré ou non-structuré. Dans le premier type, les nœuds sont organisés selon un (ou plusieurs) arbres logiques, appelés arbre(s) de diffusion. La source de la vidéo est la racine et la vidéo est distribuée à partir de la source vers les feuilles, les parents transférant la vidéo à leurs enfants. Dans un réseau non-structuré, l'arbre n'est pas défini explicitement. Les nœuds qui ont des tranches du fichier vidéo (chunks) les transfèrent de façon opportuniste à d'autres utilisateurs qui ne les ont pas. Ce deuxième type de systèmes est le plus utilisé parce qu'ils peuvent gérer facilement le churn, c'est-à-dire l'arrivée et le départ d'utilisateurs, qui est très fréquent dans les systèmes de diffusion en direct. *Ce churn est le problème principal de ces systèmes.* Les systèmes structurés ont le désavantage que le churn casse les arbres de diffusion. Cependant, nous pensons qu'ils peuvent être en fait très efficaces. Si leur structure pouvait être maintenue en utilisant des protocoles de réparation distribués très simple, même en cas de churn fréquent, cela permettrait de garder les avantages des systèmes structurés, à savoir un taux de diffusion optimal et une continuité de la diffusion avec de petits buffers, tout en étant résistant au churn,

Notre but est donc de proposer des mécanismes distribués de reconstruction pour des systèmes de diffusion d'événements en direct, de développer des modèles formels pour comprendre ces systèmes, de montrer qu'ils peuvent être efficacement simulés, et enfin qu'ils peuvent être très efficaces en pratique.

Dans [Ci47], nous proposons et analysons un algorithme local simple pour équilibrer un arbre. En particulier, en raison de limitations de bande passante, un arbre de diffusion efficace doit veiller à ce que les degrés des nœuds soient bornés. Par ailleurs, pour minimiser le délai de la diffusion en continu, la profondeur de l'arbre de diffusion doit également être contrôlée. Nous proposons ici un algorithme de réparation distribué simple dans lequel chaque nœud exécute des opérations locales en fonction de son degré et de la taille des sous-arbres de ses enfants. Nous effectuons ensuite une analyse de la complexité dans le pire cas de son temps de reconstruction.

Nous avons continué l'étude de ses systèmes lors de la thèse de Nicolas Huin. Nous avons proposé de nouveaux protocoles de reconstruction de l'arbre de diffusion et commencé une analyse de leur complexité moyenne. Nous montrons qu'il est possible de mettre en place des protocoles structurés simples de diffusion qui sont très efficaces même quand le churn est important : délai

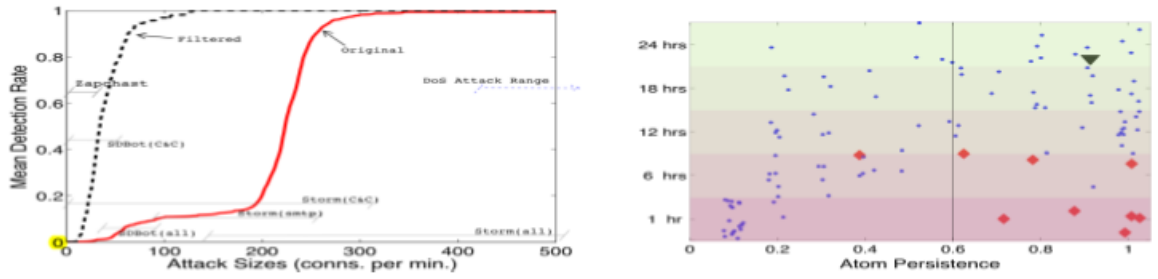
et utilisation de la bande passante quasi-optimale, tout en assurant que les interruptions de la diffusion dues à la reconstruction soient imperceptibles pour l'utilisateur [Ci41]. Ces protocoles ont été testés dans [R83] sur des traces réelles du système de diffusion Twitch [417].

Collaborations : Ce sujet a été celui de la thèse de Julian Monteiro. Mes coauteurs ont été les suivants : S. Caron, O. Dalle, N. Huin, D. Mazauric, N. Nisse, S. Pérennes, A. Tomassilli.

1.2.4 Algorithmes pour la sécurité des réseaux et analyse de traces (Travaux annexes non présentés dans l'HdR)

Pendant ma thèse, j'ai travaillé sur l'analyse d'algorithmes probabilistes pour estimer la cardinalité (le nombre d'éléments distincts) de très grands ensembles de données [Ci67, Ci65, J26]. Ces algorithmes sont utiles pour détecter les attaques par déni de services [T81]. J'ai ensuite commencé à m'intéresser à la sécurité des réseaux d'entreprises durant un postdoc dans les laboratoires de recherche d'Intel à Berkeley (US) au cours de l'année 2007. En collaboration principalement avec Nina Taft et Jaideep Chandrashekar, j'ai travaillé sur des méthodes pour assurer la sécurité des utilisateurs (*end hosts*, ordinateurs au départ ou à l'arrivée de connexions, à distinguer des routeurs ou passerelles) dans des environnements typiques d'entreprises. La recherche est basée sur la personnalisation des méthodes de protection et l'adaptation au cours du temps grâce à l'utilisation de profils construits à partir des communications passées des utilisateurs.

Profils réseaux. La première phase du projet a consisté en l'analyse des comportements variés de ces utilisateurs en utilisant une large collection de traces de plus de 300 machines d'entreprises. Nous avons montré dans [Ci64] que les comportements diffèrent fortement entre les machines et, pour un même utilisateur, en fonction de son environnement (travail ou domicile par exemple). Il faut donc individualiser le profil pour définir automatiquement un niveau de sécurité qui corresponde à l'utilisateur et à son environnement. Ce profilage permet de proposer des méthodes plus performantes de détection des anomalies que ceux des systèmes de détection actuels (ou IDS en bref pour *Intrusion Detection Systems*) comme SNORT ou BRO, qui utilisent principalement des règles pour reconnaître des attaques déjà connues à l'avance (*signature based rules*). En particulier, le profilage permet de fixer des valeurs seuils plus adaptées pour définir des comportements anormaux.



(a) Average Detection Rate for **bursts** (b) Persistence for botnet C&C atoms

Figure 1.2: Détection du trafic de botnets en utilisant la persistance d'atomes réseau.

Détection d'anomalies. Dans la deuxième phase, nous avons travaillé sur la détection d'anomalies. Nous avons ainsi étudié comment utiliser la diversité des utilisateurs de réseaux d'entreprise pour améliorer les méthodes de détection d'intrusions [Ci62]. Nous avons ensuite proposé une

méthode exploitant la temporalité du trafic réseau pour détecter des botnets, une des menaces principales pour les réseaux actuels [Ci59]. Nous introduisons la notion d'*atomes de trafic* qui agrègent les destinations et services avec lesquels un utilisateur communique. Nous calculons ensuite la *persistance* de ces atomes qui est une mesure de régularité temporelle que nous avons définie. Les atomes les plus persistants sont whitelistés durant une phase d'apprentissage. Nous traquons ensuite l'apparition de nouveaux atomes persistents pour identifier des destinations suspectes pouvant constituer un canal de Command&Control du botnet. Nous calculons cette persistance à plusieurs échelles de temps en même temps. Notre méthode ne nécessite aucune connaissance à-priori des adresses, ports ou protocoles utilisés par le canal de Command&Control du botnet, ni ne requière d'inspection du corps des paquets IP (*payload*). Nous avons évalué notre méthode en utilisant les traces d'utilisateurs de réseaux d'entreprise dont j'ai parlé plus haut, ainsi que des traces de trafic de différents botnets. Nous avons démontré que la méthode identifie correctement le trafic de C&C, même s'il est très furtif. Cette détection a de plus un taux très faible de faux positifs. Enfin, l'utilisation des whitelistes pour filtrer le trafic permet d'améliorer fortement les performances des détecteurs d'anomalies traditionnels (pour la détection de déni de service par exemple). Cette étude a mené au dépôt d'un brevet [B1].

Collaborations. Sur ce sujet, j'ai pu compter sur l'aide de mes co-auteurs : *D. Barman, J. Chandrashekar, M. Faloutsos, L. Huang, G. Iannaccone, K. Papagiannaki, E. Schooler et N. Taft.*

1.2.5 Approfondissement des outils combinatoires (Travaux annexes non présentés dans l'HdR)

L'étude de problèmes réseaux m'amène à étudier différents problèmes combinatoires. J'ai déjà mentionné les problèmes de recherche de sous-graphes minimaux. Ainsi plus généralement, j'ai étudié plusieurs problèmes de graphes suite à des questions d'allocations de fréquences, de protection de réseaux et d'optimisation de bande passante. On peut citer des problèmes de coloration de graphes, d'étiquetage d'arêtes, de recherche de circuits dans une grille et de capture de fugitifs dans le graphe du web. Parfois, la motivation réseaux est absente : recherche d'enveloppes convexes de graphes. Pour ces problèmes, j'ai indiqué après leurs descriptions des questions ouvertes qui m'intéressent.

Coloration impropre pondérée. Dans [J23], nous étudions un nouveau problème de coloration motivé par un problème pratique d'allocation de fréquences qui généralise les modèles d'interférences classique. Dans les réseaux sans-fil, un nœud interfère avec d'autres, à un niveau dépendant de nombreux paramètres : la distance entre les nœuds, la topographie physique, les obstacles, etc. Nous modélisons cela par un graphe arête-valué (G, w) où le poids d'une arête représente le bruit (ou l'interférence) entre ces deux extrémités. L'interférence totale au niveau d'un nœud est alors la somme de tous les bruits entre ce nœud et les autres nœuds émettant avec la même fréquence (utilisant la même couleur). Une k-coloration t-impropre pondérée de (G, w) est une k-coloration des nœuds de G (assignation de k fréquences) telle que l'interférence en chaque nœud n'excède pas un certain seuil t. Nous étudions le problème de la détermination du nombre chromatique t-impropre pondéré d'un graphe arête-valué (G, w), qui est le plus petit entier k tel que (G, w) admette une k-coloration t-impropre pondérée : nous donnons la complexité algorithmique, une généralisation du théorème de Lovász sur le nombre chromatique des graphes l-impropres [411] et nous résolvons le problème pour différentes grilles.

Aller plus loin. Différentes questions restent à explorer : en particulier, l'étude d'autres modèles d'interférences et d'autres graphes comme les hypercubes.

Étiquetage d'arêtes. Un bon étiquetage des arêtes d'un graphe est un étiquetage de ses arêtes tel que, pour toute paire de sommets (x, y), il n'existe pas deux chemins de x vers y dont les étiquettes sont croissantes. Cette notion a été introduite dans [324] pour résoudre des problèmes d'allocation de fréquences pour des classes particulières de graphes. Dans [J22], nous tentons de caractériser la classe des graphes qui admettent un bon étiquetage. Tout d'abord, nous produisons des familles infinies de graphes pour lesquelles aucun bon étiquetage n'existe. Nous montrons par la suite que le problème de décider si un graphe admet un bon étiquetage est NP-Complet. Finalement, nous donnons de larges classes de graphes admettant un bon étiquetage : les forêts, les graphes planaires extérieurs (outerplanar graphs) sans triangles, les graphes planaires de maille au moins 6, les graphes subcubiques sans triangles ni $K_{2,3}$.

Aller plus loin. De nombreuses questions restent à traiter, par exemple : existe-t-il un algorithme polynomial pour décider si un graphe planaire admet un bon étiquetage ? quel est le nombre maximum d'arêtes pour un bon graphe avec n sommets ?

Web caching et jeux sur les graphes. Considérons un internaute qui va d'une page Web à une autre en suivant les liens qu'il rencontre. Pour éviter que l'internaute ne (s'im)patiente, il est important d'essayer de précharger les documents avant que l'internaute ne les demande. Cependant, le coût d'un tel pré-téléchargement ne doit pas excéder le gain en temps qu'il génère. Ainsi, il faut minimiser la bande passante utilisée pour le pré-téléchargement tout en s'assurant que l'impatient internaute n'attende jamais. Dans [Ci51, Cn75, J19, J18], nous modélisons ce problème sous forme d'un jeu de type *Cops and Robber* dans les graphes. En particulier, étant donnés un graphe G qui représente le graphe du Web et une page Web de départ $v_0 \in V(G)$, nous définissons l'*indice de contrôle* de G, $ic(G, v_0) \in \mathbb{N}$, qui modélise la vitesse minimum de téléchargement suffisante pour que l'internaute partant de v_0 n'attende jamais quoi qu'il fasse. Nous considérons le problème de décider si $ic(G, v_0) \leq k$ et démontrons plusieurs résultats de complexité. En particulier, décider si $ic(G, v_0) \leq 2$ est NP-difficile si G est cordal, et décider si $ic(G, v_0) \leq 4$ est PSPACE-complet si G est un graphe orienté acyclique.

Aller plus loin. Il nous reste à caractériser plus précisément le coût de la connexité (l'ensemble des sommets marqués doit être connexe). Existe-t-il un graphe G et $v_0 \in V(G)$ tels que $ic(G, v_0)+1 < cic(G, v_0)$? Existe-t-il f, une constante ou une fonction de n tel que $cic(G, v_0) \leq f \cdot ic(G, v_0)$ pour tout graphe G de n sommets et $v_0 \in V$?

Convexité. J'ai aussi étudié une notion de convexité dans les graphes introduite dans [404]. Dans [Ci53, J21], nous nous concentrons sur la question de la complexité du calcul de l'enveloppe minimum d'un graphe dans le cas de diverses classes de graphes.

Étant donné un graphe $G = (V, E)$, l'intervalle $I[u, v]$ entre deux sommets $u, v \in V$ est l'ensemble des sommets qui appartiennent à un plus court chemin entre u et v. Pour un ensemble $S \subseteq V$, on note $I[S]$ l'ensemble $\bigcup_{u,v \in S} I[u, v]$. Un ensemble $S \subseteq V$ de sommets est dit *convexe* si $I[S] = S$. L'*enveloppe convexe* $I_h[S]$ d'un sous-ensemble $S \subseteq V$ de G est défini comme le plus petit ensemble convexe qui contient S. $S \subseteq V$ est une *enveloppe* de G si $I_h[S] = V$. Le *nombre enveloppe* de G, noté $hn(G)$, est la cardinalité minimum d'une enveloppe du graphe G.

Nous montrons que décider si $hn(G) \leq k$ est un problème NP-complet dans la classe des graphes bipartis répondant ainsi à une question ouverte de [316] et nous prouvons que $hn(G)$ peut être calculé en temps polynomial pour les co-bipartis, $(q, q-4)$-graphes et cactus. Nous montrons aussi des bornes supérieures du nombre enveloppe des graphes en général, des graphes sans triangles et des graphes réguliers.

Aller plus loin. Un grand nombre de problèmes restent ouverts. Déterminer la complexité pour les graphes d'intervalles et les graphes planaires. D'autres types de convexités dépendant

du type de chemins considérés sont aussi à étudier : convexité monophique pour les chemins induits, P_3-convexité.

Collaborations : Ces thématiques ont été celles de la thèse de Julio Araujo. J'ai aussi eu la chance de pouvoir compter sur le travail de mes co-auteurs : J.-C. Bermond, *V. Campos*, N. Cohen, *F. Fomin*, F. Havet, *A. Jean-Marie*, D. Mazauric, R. Modrzejewski, N. Nisse, S. Pérennes, L. Sampaio, R. Soares.

1.3 Méthode

Mon approche a été d'utiliser des outils théoriques de différentes natures pour mieux appréhender la difficulté des nouveaux problèmes posés et pour proposer des méthodes efficaces pour les traiter. J'ai utilisé des méthodes provenant de l'algorithmique et la théorie de la complexité, de la théorie des graphes, d'optimisation, et d'analyse probabiliste. J'ai ensuite testé l'impact de ces méthodes en pratique. Ma méthode générale pour comprendre un nouveau problème peut être résumée de la façon suivante et est illustrée par des exemples tirés principalement des deux sujets que j'ai développés dans les deux parties de ce manuscrit, l'efficacité énergétique des réseaux et le développement des nouveaux réseaux logiciel virtualisés.

- **Complexité algorithmique.** La première étape est d'étudier la complexité algorithmique du nouveau problème. Est-il polynomial, NP-complet, ... ? Ensuite, si le problème est NP-complet, peut-il être approché ?

  Pour un problème de minimisation ayant une solution optimale de valeur z^*, un *algorithme d'approximation* de facteur $\rho > 1$ (i.e. un algorithme ρ-approché) est un algorithme donnant une solution de valeur z , avec la garantie que $z \leq \rho z^*$.

  Il existe différentes classes d'algorithmes d'approximation. On dit que le problème admet un Schéma d'Approximation en Temps Polynomial ou PTAS en bref pour *Polynomial-Time Approximation Scheme* s'il existe un algorithme polynomial d'approximation à n'importe quel facteur $(1+\varepsilon)$, $\varepsilon > 0$. Un problème est dans APX s'il existe un algorithme d'approximation en temps polynomial avec un facteur d'approximation constant pour le résoudre.

  Exemple (histoire verte) : nous montrons dans la première section de la première partie que le problème de routage efficace en énergie considéré dans la partie I n'est pas dans APX.

- **Proposer des méthodes efficaces de résolution.** Quand la complexité du problème a été déterminée, il s'agit ensuite de proposer des méthodes efficaces pour le résoudre. Ces méthodes sont de différentes natures et sont discutées ci-dessous.

  - **Avec garanties théoriques :** Quand le problème est polynomial, il est souvent possible de proposer un algorithme exact qui peut résoudre même de grandes instances. Quand le problème est NP-complet, les algorithmes (exponentiels) exacts ne peuvent souvent résoudre que de petites instances. Il devient intéressant de proposer des algorithmes d'approximation en temps polynomial quand c'est possible. Une autre solution est de réduire l'explosion combinatoire à un seul paramètre, c'est l'idée des algorithmes FPT pour *fixed-parameter tractable*, traduit littéralement en soluble à paramètre fixé.
    Un problème paramétré (Q, κ) est dans la classe FPT s'il admet un algorithme qui résout le problème en temps proportionnel à $f(\kappa(x)) \times poly(|x|)$, où $poly(.)$ est une

fonction polynomiale, $|x|$ est la taille de l'instance x, et f est une fonction quelconque calculable. Ces algorithmes sont très intéressants en pratique quand le paramètre κ a une petite valeur dans les instances pratiques (par exemple, si c'est le diamètre d'un réseau...)

Exemple (histoire virtuelle) : nous proposons en section 7.2 un algorithme à facteur logarithmique pour le problème de placement des chaînes de fonctions de services étudié dans le chapitre 7.

Exemple (histoire virtuelle) : nous proposons des algorithmes FPT pour le problème de compression de table de routage multi-dimensionelles des réseaux logiciels en section 6.5.2 du chapitre 6.

– **Méthodes d'optimisation.** La méthode d'optimisation utilisée principalement dans ce manuscrit est la modélisation sous la forme d'un programme linéaire ou LP en bref pour *Linear Program* ou d'un programme linéaire à nombre entiers ou ILP en bref pour *Integer Linear Program*.
Les résultats principaux utilisés sur ces programmes linéaires sont i) qu'un LP (variables fractionnaires) peut-être résolu en temps polynomial en le nombre de ses variables et de ses contraintes ; ii) qu'il existe des méthodes très efficaces pour les résoudre, à savoir la méthode du simplexe pour les LP, même si elle est en théorie exponentielle dans le pire cas, elle est très souvent linéaire dans le nombre de contraintes, ou les méthodes théoriquement polynomiales comme la méthode de l'ellipsoïde ou celle des points intérieurs. Pour les ILP, les techniques de *branch and bound and cut and price* sont souvent très efficaces comme elles permettent de tronquer l'arbre de recherche ; iii) qu'il existe des théories pour borner l'écart entre la meilleure solution trouvée et l'optimal , en particulier la théorie de la dualité et enfin iv) qu'il existe des solveurs CPLEX, glpk, Coin, ... pour résoudre efficacement en pratique un programme linéaire.
Ces méthodes permettent parfois d'obtenir en pratique des solutions optimales aux problèmes d'optimisation considérés. Quand les instances considérées sont trop grandes pour être résolues, il est possible d'obtenir de bonnes solutions pour lesquelles on connait l'écart à l'optimal. Il est aussi possible d'accélérer la résolution en utilisant des techniques comme la génération de colonnes. Nous en discutons ci-dessous dans la partie décomposition.

Exemple (histoire verte) : Le problème de routage efficace en énergie est modélisé comme un ILP dans la section 3.1.3, en ajoutant des possibilité d'élimination de redondance dans la section 3.2 et de caching dans la section 3.3.

Exemple (histoire virtuelle) : Notez qu'il existe de nombreuses modélisations sous forme de programmes linéaires d'un problème. Il s'agit souvent de trouver une modélisation la plus efficace possible. Le problème de placement de chaînes de services consiste à placer des fonctions virtuelles dans un ordre *donné. Il existe différentes façons de modéliser ces contraintes d'ordres. Une façon classique est d'avoir des variables binaires disant qu'une fonction est placée avant une autre et ensuite d'empêcher des cycles de dépendances. Nous avons proposé une modélisation à base de graphes en couches qui est beaucoup plus efficace, voir la section 7.1 du chapitre 7.*

– **Algorithmes heuristiques.** De nombreux problèmes pratiques sont trop compliqués ou ont trop de paramètres pour qu'il soit possible de trouver des algorithmes avec des garanties théoriques ou de les résoudre exactement en utilisant des méthodes

d'optimisation. Il arrive que l'instance du problème soit si grosse que les solveurs de LP ne puissent même pas la charger en mémoire. Dans ce cas, il est important de proposer des algorithmes qui marchent le mieux possible en pratique, on appelle ces algorithmes des algorithmes heuristiques ou tout simplement des heuristiques. Ces algorithmes doivent ensuite être validés comme ils n'ont pas de garantie théorique. Cette validation peut être effectuée de plusieurs manières. On peut comparer leurs solutions

  * à des solutions optimales quand elles sont possibles à calculer, soit parce que les instances sont petites et que les méthodes d'optimisation exacte permettent de les résoudre, soit parce que les instances considérées sont particulières et qu'une solution optimale peut être calculée théoriquement.
  * aux solutions d'autres algorithmes heuristiques (en particulier proposés auparavant dans la littérature) et montrer que l'algorithme proposé se comporte mieux.

Exemple (histoire verte) : pour résoudre le problème EAR, nous introduisons une heuristique gloutonne qui est validée en étant comparée i) à un algorithme à choix aléatoires ii) à des solutions optimales pour des instances de tailles petites et moyennes.

– **Décomposition du problème.** Quand on ne sait pas trouver de solutions efficaces pour un problème, il est souvent intéressant de *décomposer le problème en sous-problèmes que l'on réussit à résoudre efficacement.* Pour certains sous-problèmes, il sera souvent possible de proposer des algorithmes avec garanties théoriques. Ces décompositions sont naturelles en algorithmique où un problème de routage et allocation de ressources se divisera naturellement en deux sous-problèmes, mais existe aussi en optimisation. Nous avons en particulier utilisé des modèles de décompositions que nous avons ensuite résolu en utilisant la génération de colonnes. L'idée est de diviser le problème en un problème maitre avec peu de variables (ou colonnes) et un second problème, appelé problème de *pricing*, qui permet de trouver des colonnes intéressantes à rajouter au problème maitre. La méthode est efficace quand le problème de pricing peut être résolu par un algorithme efficace.

  Exemple (histoire virtuelle) : Dans le chapitre 6, nous nous sommes intéressés au problème de routage dans les réseaux logiciels. Nous avons divisé le problème en deux : le problème de routage et le problème de compression de tables. Sur le problème de compression de tables, nous avons pu proposer un algorithme avec facteur constant d'approximation, section 6.5.1.

  Exemple (histoires verte et virtuelle) : Pour résoudre les problèmes de routage avec chaînes de services, nous avons utilisé la génération de colonnes, sections 8.3 et 7.1. Notre sous-problème de pricing est la recherche d'un plus court chemin (contraint ou non en fonction des variantes) dans un graphe en couches. Il existe des algorithmes très efficaces que nous avons utilisés pour résoudre ce sous-problème. Cela nous a permis de résoudre des instances beaucoup plus larges que celles résolues dans la littérature. Nous avons aussi proposé en section 7.1 une heuristique pour minimiser le nombre de ressources virtuelles utilisées, dans laquelle le placement de ces ressources étaient géré par une variante de l'algorithme K-mean.

- **Classe de graphes particulières.** Les réseaux ont souvent des propriétés particulières qui sont dues aux impératifs considérés lors de leurs conceptions ou à l'historique de leur formation. Par exemple, les réseaux télécom doivent pouvoir tolérer des pannes de liens.

Ils sont donc quasiment tout le temps 2-connexes. De même beaucoup de réseaux sociaux ont une distribution des degrés en loi de puissance et sont petit monde (c'est-à-dire que la distance moyenne entre deux sommets croît logarithmiquement avec leurs nombres de sommets). Il est souvent possible d'exploiter ces propriétés particulières pour proposer des algorithmes plus efficaces que pour des réseaux quelconques. Par exemple, par exemple le problème classique Minimum Vertex Cover, qui consiste à couvrir toutes les arêtes d'un graphe par un sous-ensemble de sommets de taille minimum, est polynomial dans les graphes planaires alors qu'il est NP-complet dans les graphes généraux.

Exemple : Le problème EAR est NP-complet et non-APX et donc difficile à résoudre pour des instances de tailles moyennes. Nous avons par contre résolu en section 4.2 le problème pour les grilles qui sont un type de réseaux utilisés pour certains cas pratiques et certaines modélisations.

- **Tester en pratique.** Les méthodes proposées doivent souvent être validées. C'est le cas en particulier pour les algorithmes heuristiques, mais notez aussi que les garanties théoriques ne sont souvent que des analyses du pire cas et, qu'en pratique, les résultats moyens donnés par les algorithmes sont souvent bien meilleurs. Les méthodes peuvent être testées sur des jeux de données (topologies réseaux, matrices de trafic) synthétiques (par exemple générés aléatoirement) ou provenant de cas pratiques (topologies d'un opérateur réseau et traces de trafic réseau). Différents niveaux de validation, plus ou moins proches d'une mise en production, peuvent être envisagés : évaluations probabilistes, évaluations numériques, simulations, émulations ou expérimentations.

  - **Évaluation théorique et modèles probabilistes.** Les systèmes pratiques peuvent souvent être bien représentés par des modèles probabilistes. Le temps d'arrivées de flots, la popularité de vidéos, les occurrences de pannes sont par exemple souvent donnés par une distribution de probabilité (construite à partir de données réelles). L'état d'un système peut ensuite se modéliser par une chaîne de Markov qui donne les probabilités de passer d'un état à l'autre. Par exemple, l'état d'un cache réseau peut être donné par un modèle de file d'attente. Le résultat principal est que, sous certaines conditions comme l'irreductibilité (tout état peut être atteint à partir de m'importe quel état) et l'apériodicité, ces systèmes admettent une distribution stationnaire unique qui décrit bien leur état. De plus, ces systèmes convergent vers cette distribution stationnaire qui peut-être calculée en utilisant des équations de stabilité.

    Exemple (histoire annexe P2P) : Je me suis intéressé à des façons de stocker des fichiers de façon distribuée sur les pairs d'un réseau pair-à-pair. Pour éviter les pertes de données, celles-ci sont encodées puis répliquées. Nous avons testé l'efficacité de différentes méthodes de codage et de réparation des données en modélisant ces systèmes avec des chaînes de Markov.

  - **Évaluation numérique, simulations ou émulation sur des topologies et du traffic réel.** C'est la méthode que nous avons le plus employée. Le principe est de se placer dans des scénarios réalistes avec des topologies de réseaux et du trafic existants. Mes jeux de données préférés proviennent de la bibliothèque SNDlib [280] qui répertorie 26 topologies, avec les capacités des liens, les matrices de trafic de chaque réseau, et pour certaines, leur évolution dans le temps avec une valeur toutes les 5 minutes. J'ai aussi utilisé pendant mes travaux des architectures réseaux provenant

directement d'opérateurs comme Orange et des traces de trafic réseau de portable Intel. Les modèles d'optimisation et les algorithmes sont ensuite évalués numériquement ou simulés sur ces jeux de données. J'ai aussi parfois participé à des évaluations par émulation. A la différence de la simulation qui vise à imiter un modèle abstrait, l'émulation vise à reproduire fidèlement le comportement physique d'un matériel par un logiciel. Il est ainsi possible de tester l'impact d'une nouvelle méthode sur les protocoles existants et de mesurer des métriques comme les délais réels et les pertes de paquets qui sont souvent fortement abstraits dans les modèles.

Exemple (histoire verte) : Nous avons évalué numériquement en section 3.1.6 le programme linéaire et simulé les algorithmes heuristiques proposés dans le chapitre 8 pour obtenir un routage efficace en énergie sur un ensemble de topologies de réseaux et pour les matrices de trafic tirées de SNDlib.

Exemple (histoire verte) : Dans la dernière étape de notre périple vert, nous discutons des méthodes pour mettre en pratique les méthodes efficaces en énergie discutées précédemment. Ces méthodes impliquent de changer dynamiquement les configurations réseaux au cours de la journée pour s'adapter à l'utilisation des ressources. Pour convaincre des opérateurs, il est important de vérifier que cela ne génère pas de perte de paquets par exemple. Nous avons testé les méthodes proposées en section 8.4.2 par émulation en utilisant Mininet[7] en section 8.4.3.

– **Expérimentations.** La dernière étape avant la mise en production est d'effectuer des expérimentations matérielles pour tester les méthodes proposées. Ces expérimentations peuvent avoir des échelles très différentes : cela peut consister en le test d'un équipement, par exemple pour obtenir son profil énergétique, l'utilisation d'une petite plateforme expérimentale, ou le déploiement de son code sur une plateforme existante comme Grid5000[8] ou PlanetLab[9].

Exemple (histoire verte) : Dans la section 3.2, nous avons étudié l'emploi de méthode d'élimination de redondance pour réduire la consommation énergétique d'un réseau. Pour fixer les paramètres de nos modèles d'optimisation, nous avons testé expérimentalement l'utilisation d'un équipement réseau qui effectue cette élimination de redondance, un WOC pour Wan Optimization Controller. Nous avons testé la consommation énergétique du WOC pour différents volumes de trafic réaliste créé avec un générateur spécialisé d'Orange. Cela nous a permis d'utiliser ensuite un modèle énergétique réaliste.

Exemple (histoire virtuelle) : Dans le chapitre 6, nous nous sommes intéressés à des méthodes de compression de tables de routage. Pour vérifier que ces méthodes peuvent être utilisées en pratique, nous avons utilisé une petite plateforme SDN. Cela a permis de vérifier qu'elles n'entrainaient pas de grands délais sur les paquets, qu'elles ne généraient pas de pertes de paquets et que le contrôleur SDN n'était pas saturé en CPU ou bande passante.

[7] http://mininet.org/
[8] https://www.grid5000.fr/
[9] https://www.planet-lab.org/

1.4 Encadrement

Comme mentionné précédemment, la plupart de mes travaux ont été fait en collaboration avec des étudiants. Sur mes différents projets, j'ai ainsi eu la chance d'encadrer, officiellement ou non, les doctorants, post-doctorants et étudiants de master 2 suivants. J'indique le pourcentage estimé des charges d'encadrement pour chaque étudiant.

Encadrement de thèses (en cours) :

- Co-directeur de Giuseppe di Lena (à 33%), avec Thierry Turletti, DR inria, et Chidung Lac, Orange Labs Lannion, bourse CIFRE du laboratoire commun Orange-Inria. *Resilience of virtualized networks.* Début : mars 2018. ***Histoire virtuelle.***

- Co-directeur de Thibaud Trolliet (à 50%), avec Arnaud Legout, CR Inria. *Analysis of large social networks.* Début : octobre 2017.

- Co-directeur d'Andrea Tomassilli (à 70%) avec Stéphane Pérennes, DR CNRS. *Next generation virtualized networks.* Début : octobre 2016. ***Histoire virtuelle.***

Encadrement de thèses (soutenues) :

- Directeur de Nicolas Huin (à 70%) avec Dino Lopez, équipe SIGNET, laboratoire I3S. 2014-2017. Soutenue le 28 septembre 2017.
  Energy-efficient Software Defined Networks. ***Histoires verte et virtuelle.***
  Devenir : Postdoc à l'université Concordia, Montréal, Canada. Puis, chercheur au Huawei Research Lab, Paris.

- Co-directeur de Julio Araujo (à 50%) avec Jean-Claude Bermond et Claudia Linhares, University of Ceara, Brésil. 2009-2012. Soutenue le 13 septembre 2012.
  Graph Coloring and Graph Convexity. **Histoires verte et annexe outils combinatoires.**
  Devenir : Professeur assistant à l'Université du Ceara, Fortaleza, Brésil.

- Encadrement de la thèse de Remigiusz Modrzejewski (à 70%) avec Jean-Claude Bermond. 2010-2013. Membre du jury de thèse. Soutenue le 24 octobre 2013.
  Content Distribution and Storage. ***Histoires verte et annexe P2P.***
  http://www-sop.inria.fr/members/Remigiusz.Modrzejewski/thesis.html
  Devenir : Employé chez Google, Dublin, Irlande.

- Encadrement de la thèse de Julian Monteiro (33%) avec Olivier Dalle, MCF Université Côte d'Azur, et Stéphane Pérennes, DR CNRS . Membre du jury de thèse. Soutenance : 16 novembre 2010.
  Modeling and Analysis of P2P Data Storage Systems. ***Histoire annexe P2P.***
  http://www.ime.usp.br/~jm/thesis.php
  Devenir : Chercheur et responsable d'équipe à Cittati Tecnologia. Co-Fondateur et responsable technologique de la startup Lejour.

Encadrement de postdocs

- Luca Chiaraviglio a effectué un postdoc de septembre 2012 à mars 2013 sur l'économie d'énergie dans les réseaux. ***Histoire verte.***
  Devenir : Professeur assistant (Tenure-Track) à l'Université de Rome Tor Vergata.

- ▷ Yaning Liu a effectué un postdoc de mars 2011 à février 2012 sur la même thématique, financée sur l'ANR DIMAGREEN. ***Histoire verte.***
  Devenir : Chercheur à JCP-Consult R&D pour la recherche et le management de projets européens.

Encadrement de stages J'ai aussi encadré une dizaine de stages, principalement de Master 2, que je liste ci-dessous :

- ▷ Badr Jouhar, Master 2 IFI, parcours Ubinet. March-August 2018. *Joint optimization of network and servers in a datacenter.*
- ▷ Adrien Gausseran, Master 2 RIF. March-August 2018. *Network slicing for 5G.*
- ▷ Rohit Agrawal, Master 2 IFI, parcours Ubinet. March-August 2017. *Joint optimization of network and servers in a datacenter.*
- ▷ Andrea Tomassilli, Master 2 IFI, parcours Ubinet. March-August 2016. *Study of a Distributed Live Video Streaming System.*
- ▷ Stefano Ponziani, Master 2 IFI, parcours Ubinet. March-August 2016. *Random models for directed social networks like Twitter.*
- ▷ Nicolas Huin, Master 2 RIF, France. March-August 2014. *Study of a Distributed Live Video Streaming System.*
- ▷ Remigiusz Modrzejewski, Master 2 IFI, parcours Ubinet, France. March-August 2010. *Live Streaming in Peer-to-Peer Systems.*
- ▷ Sandeep Kumar Gupta, IIT Delhi, India. May-July 2009 *How Bandwidth Availability Affects Data Lifetime in Distributed Storage Systems.*
- ▷ Nikhil Arora, IIT Delhi, India, May-July 2009. *Implementation and evaluation of a peer-to-peer storage system over the AEOLUS (Algorithmic Principles for Building Efficient Overlay Computers / European project) testbed.*
- ▷ Stéphane Caron, ENS Paris. June-August 2009. *Surveying Different Placement Policies in P2P Storage Systems.*
- ▷ Julio Araujo, Master 2, University of Ceara, Brazil, March-April 2009. *Good edge-labelling of graphs.*

Part I

Un peu de vert dans les réseaux

Chapter 2

Introduction

Motivation

Contexte. Compte tenu de l'augmentation du coût de l'énergie et de la forte croissance de la demande, le besoin de solutions économes en énergie est devenu un impératif pour les gouvernements, les entreprises et les particuliers. Les projections pour la consommation énergétique des ICT sont qu'elle représentera 14% de la consommation globale d'électricité en 2020 [188, 195], alors qu'elle ne représentait que 4.7% en 2012 [192]. En 2015, cette consommation était de 1700 TWh représentant un coût proche de 250 milliards d'euros. Elle se divise en quatre catégories principales : la consommation des appareils terminaux (ordinateurs portables, téléphone, télévisions connectées,...) 55%, celle des réseaux 19%, celles des centres de données 11%, et celle utilisée par la production de tous ces équipements 20% [129]. Dans les prochaines années, il est prévu que la consommation électrique des réseaux et centres de données augmentera bien plus que celles des appareils terminaux. [129] estime que les parts respectives dans la consommation des quatre catégories considérées sera en 2030 de 14% pour les appareils terminaux, 13% pour les réseaux, 58% pour les centres de données (en incluant la consommation des réseaux d'accès pour traiter le trafic vers et à partir des centres de données) et 15% pour la production. Par conséquent, un objectif très important est la réduction de la consommation d'énergie pour l'exploitation et la gestion des réseaux existants.

Pour réduire la consommation d'énergie induite par Internet, plusieurs techniques peuvent être utilisées :

- Au niveau du matériel : la consommation globale du réseau peut être réduite par des progrès technologiques dans la création d'équipements réseau, par exemple, des systèmes de refroidissement plus efficaces ou l'introduction de différents états matériels en fonction du niveau d'activité.

- Au niveau de la conception (design) du réseau : les réseaux doivent être conçus pour répondre aux nouveaux besoins et aux applications des utilisateurs. Par exemple, des changements de topologies réseaux et une réplication bien répartie des données sont un moyen de réduire les ressources utilisées par les réseaux.

- Au niveau de la gestion du réseau : lorsque le réseau a déjà été conçu, de nouvelles politiques d'utilisation des ressources, par exemple de routage, prenant en compte la consommation d'énergie peuvent être introduites.

Dans mes travaux, je me suis principalement concentré sur les deux derniers points, des changements de conception et de gestion des réseaux, qui peuvent être attaqués en utilisant des outils de théorie des graphes, d'optimisation et d'algorithmique.

État de l'art

Si l'efficacité énergétique a toujours été une préoccupation dans la conception du matériel, il a été considéré dans le passé comme un problème d'ingénierie ou confiné dans des domaines spécifiques : par exemple, dans les réseaux de capteurs, des politiques de routage intelligentes et des protocoles cryptographiques légers utilisant peu d'énergie ont été proposés pour sauvegarder la batterie limitée des appareils. *Vers la fin des années 2000*, avec la très forte croissance du trafic réseau et l'augmentation du coût de l'énergie, une pression est apparue pour trouver des méthodes prenant en compte la consommation énergétique des réseaux. Ce domaine de recherche, à un stade précoce de développement, a attiré l'attention de la *communauté réseau* : en 2009, l'efficacité énergétique a été ajoutée dans les domaines d'intérêt de grandes conférences généralistes comme Infocom (Power Control & Management) ou spécialisées comme ONDM (efficacité énergétique dans les réseaux optiques) ou de nouveaux workshops, par exemple, GreenComm, mis en place par les opérateurs de réseaux qui voulaient trouver des méthodes pour réduire leurs coûts énergétiques. Les *fabricants de matériel réseau* ont également montré leur intérêt pour l'étude des réseaux verts. En effet, sous la pression de la hausse des coûts de l'énergie et des normes environnementales de plus en plus rigides, les gouvernements et les entreprises du monde entier ont renforcé les budgets d'énergie et d'émissions, créant ainsi la demande de nouvelles générations d'équipements de télécommunications économes en énergie. Par exemple, dans un rapport de 2008 [281], les employés de Juniper étudient comment réaliser en pratique une mesure objective de l'efficacité énergétique dans le monde des télécommunications. Dans [348], Cisco montre comment réduire considérablement la consommation d'énergie et de refroidissement en créant une plate-forme intégrée de six périphériques différents (routeurs, commutateurs, points d'accès sans fil ...).

Traditionnellement, les réseaux de télécommunication ont été conçus pour maximiser la bande passante disponible et son exploitation. Cette politique tend à minimiser les coûts de remplacement qui surviennent lorsque les technologies sont mises à jour pour répondre à une augmentation du nombre d'utilisateurs ou du trafic échangé. Cependant, les utilisateurs accèdent au réseau à différents moments de la journée et les applications utilisées sont différentes en termes de trafic échangé. Les réseaux sont dimensionnés pour les demandes de trafic les plus élevées, de sorte que la quantité de données circulant sur le réseau est normalement bien en deçà des débits de données maximaux réalisables [380].

La consommation d'énergie des périphériques réseau dépend normalement de la capacité installée et de la technologie sous-jacente [343]. Par conséquent, un périphérique réseau consomme une quantité d'énergie largement indépendante de la quantité actuelle de données que l'appareil est en train de traiter.

L'idée principale de la conception et de l'exploitation de réseaux efficaces en énergie est de réduire l'écart entre l'utilisation du réseau et la capacité offerte. A partir des travaux pionniers de Gupta et al [370], différentes solutions sont étudiées pour réduire le gaspillage d'énergie, ou, de manière équivalente, pour rendre la consommation du réseau proportionnelle à la charge de trafic [255, 272]. Les approches proposées peuvent être divisées en deux grandes catégories : (1) des approches proportionnelles en énergie qui s'adressent à des dispositifs individuels et tentent d'atteindre la proportionnalité énergétique en adaptant la vitesse (et la capacité) des dispositifs

à la charge réelle, et (2) des approches utilisant des modes veille qui affectent le réseau dans son ensemble et approchent la proportionnalité de la charge en distribuant soigneusement le trafic dans le réseau afin que certains dispositifs soient pleinement utilisés et que d'autres dispositifs deviennent inactifs et soient mis en mode veille. En particulier, les modes veille sont motivés par le fait que la consommation d'énergie des appareils actuels est pratiquement indépendante de la charge [343, 235]. Par conséquent, éteindre les appareils permet d'économiser une quantité constante d'énergie. Cependant, les modes veille introduisent un niveau de complexité supplémentaire dans le réseau. En fait, la coordination entre les dispositifs est nécessaire. En particulier, le trafic doit être redirigé des appareils qui vont être éteints vers d'autres appareils qui restent allumés. Le choix des appareils à éteindre est un problème ouvert. Ensuite, le réseau doit mettre en œuvre des mécanismes pour réagir à l'augmentation du trafic, c'est-à-dire en choisissant les appareils qui doivent être mis sous tension. Enfin, le réseau doit mettre en place des mécanismes pour garantir la qualité de service (QoS) pour les utilisateurs lorsque des approches d'économie d'énergie sont mises en place.

Verrous scientifiques et techniques

Les principales difficultés pour réussir à rendre les réseaux plus efficaces en énergie sont les suivantes :

- **Mesures.** Afin de pouvoir évaluer l'utilisation d'énergie d'un réseau entier, la consommation d'énergie des différents éléments du réseau, comme par exemple celle d'un routeur, doit être étudiée. Cette étude est très difficiles pour plusieurs raisons : la multiplicité des équipements réseaux et de leurs paramètres de configurations, leur coût, l'accessibilité à ces équipements, en particulier dans leurs conditions normales d'utilisation. Cette étude est très importante car elle sert de base pour proposer des fonctions de coût réalistes spécifiques aux équipements du réseau qui seront ensuite utilisées dans les modèles d'optimisation.

- **Optimisation.** L'étude de la consommation d'énergie des différents composants du réseau fournira des fonctions de coût réalistes pour les différents appareils. Ces fonctions ont très peu de chance d'avoir une forme classique pour plusieurs raisons : les résultats préliminaires indiquent que dans une plage de fonctionnement très intensive, un routeur augmente plus que linéairement sa consommation. De même, à l'opposé du spectre, un routeur qui achemine un trafic très faible et peu dense gaspillerait une grande partie de ses ressources. La difficulté qui apparaît dans ces conditions est que la plupart des travaux de la littérature supposent une fonction de coût convexe plus facile à manipuler. Quoi qu'il en soit, cela nous laisse sans recette pour résoudre nos problèmes. C'est pourquoi, nous voulons proposer des outils spécifiques pour résoudre ces problèmes. Par conséquent, trouver une politique de routage efficace pour ce type de fonctions de coût implique de résoudre des problèmes d'optimisation très difficiles.

- **Solutions dynamiques.** Enfin, à la base même de la famille de solutions efficaces en énergie que nous avons étudiée, se trouve le processus dynamique d'adaptation de l'utilisation des ressources à la dynamique des demandes et du trafic. Cependant les réseaux actuels sont très difficiles à configurer et leurs opérateurs très réticents à envisager des solutions dynamiques. Il faudra donc proposer et évaluer de nouvelles méthodes permettant de mettre en pratique de telles solutions.

Plan

Au cours de ces dernières années, j'ai effectué de nombreux travaux dans le but d'améliorer l'efficacité énergétique des réseaux, en m'intéressant à des aspects théoriques pour comprendre le cœur de la difficulté du problème, mais aussi à des aspects pratiques pour voir comment ces travaux pourraient être vraiment mis en œuvre par les opérateurs réseaux.

- Dans un premier temps, j'ai formalisé dans le chapitre 3 un problème simplifié de minimisation de l'énergie nécessaire pour transporter le trafic réseau. J'ai ensuite étudié sa complexité, puis analysé certains scénarios pratiques.
- Ce faisant, j'ai rencontré certains problèmes fondamentaux de théorie des graphes, comme la recherche de sous-graphes minimaux en nombre d'arêtes supportant la charge des demandes que j'étudie dans le chapitre 4.
- Enfin, je me suis demandé comment mettre en pratique les solutions efficaces en énergie proposées en m'appuyant sur les nouveaux paradigmes des réseaux logiciels et de la virtualisation réseaux dans le chapitre 8. Le chapitre n'apparait pas directement dans ce manuscrit et en constitue le dernier chapitre car il utilise les notions présentées dans la partie II qui est consacrée aux technologies SDN et NFV.

Ces travaux ont en partie été faits dans le cadre de l'ANR DIMAGREEN[1] (DesIgn and MAnagement of GREEN networks with low power consumption), puis du projet européen TREND, dont Inria (et en particulier COATI) était devenue une institution collaborative.

Ils sont le fruit d'une collaboration d'une part avec des collègues de COATI (à l'époque MASCOTTE), en particulier J. Moulierac, de six doctorants de l'équipe, D. Mazauric, B. Onfroy, T. K. Phan, R. Modrzejewski, I. Tahiri et J. Araujo ; d'autre part avec des chercheurs d'Orange Labs, Frédéric Roudaut, à Sophia Antipolis, Esther Le Rouzic, à Lannion et des chercheurs d'autres universités, *E. Bonetto, L. Chiaraviglio, R. Gonzalez, C. Guerrero, B. Jaumard, F. Musumeci, Y. Liu, A. Bianco, R. Gonzalez, F. Idzikowski, F. Jimenez, C. Lange, J. Montalvo, A. Pattavina, A. Valenti, W. Van Heddeghem, and Y. Ye.*

Ils ont donné lieu à un chapitre de livre [ch5], 6 publications dans des journaux internationaux [J17, J15, J13, J7, J10, J6], 10 dans des conférences internationales [Ci56, Ci50, Ci45, Ci49, Ci44, Ci40, Ci39, Ci43, Ci35, Ci33] et une dans une conférence nationale [Cn76].

[1] http://www-sop.inria.fr/teams/mascotte/Contrats/DIMAGREEN/wiki/

Chapter 3

De premiers résultats

Dans ce chapitre, je m'intéresse aux gains énergétiques qui peuvent être obtenus dans les réseaux en jouant sur le routage (principalement) et sur la mise en cache de certaines données (de façon secondaire). Je présente d'abord le problème principal étudié, à savoir le routage efficace en énergie ou EAR en bref pour Energy Aware Routing et je fournis ensuite une étude de sa complexité. Ensuite, j'étudie trois scénarios réseaux différents qui correspondent à mes trois sections. Dans la première, la section 3.1, je m'intéresse aux gains énergétiques obtenus pour des topologies réseaux cœur classiques. Dans une deuxième, la section 3.2, je montre comment améliorer ces gains en utilisant l'élimination de redondance dans le trafic réseau. Enfin, dans la dernière section, la section 3.3, je m'attaque à deux scénarios de diffusion de contenus et montre les gains énergétiques dans ce contexte.

3.1 Difficulté du problème de base et premiers résultats sur des réseaux ISP

Ce travail est une collaboration avec deux anciens doctorants du groupe COATI (à l'époque MASCOTTE), D. Mazauric, maintenant CR Inria Sophia Antipolis et B. Onfroy, actuellement Ingénieur à Avisto, et avec J. Moulierac. Il a donné lieu aux publications [Ci56, Cn76, ch5].

3.1.1 Introduction

Comme discuté, l'économie d'énergie dans les réseaux peut être accomplie en utilisant différentes techniques. Dans ce travail, nous considérons une méthode algorithmique qui est de changer le routage pour pouvoir mettre en veille des équipements réseaux. En effet, plusieurs études [345, 306] montrent que la charge de trafic des routeurs n'a qu'une faible influence sur leur consommation d'énergie. Par conséquent, le facteur dominant pour expliquer cette consommation est le nombre (pondéré par la consommation) d'éléments de réseau allumés : interfaces, plates-formes, routeurs, ... Afin de minimiser l'énergie utilisée pour router le trafic, nous devons donc essayer d'utiliser le moins possible d'éléments réseau. Néanmoins, dans la plupart des réseaux, les PoPs ou même les routeurs ne peuvent pas être désactivés. En effet, ils sont la source ou la destination des demandes. Pour cette raison, nous ne considérons que la mise en veille de liens réseau. Nous étudions une architecture simplifiée où une connexion entre deux routeurs est représentée par un lien joignant deux interfaces réseau. Mettre en veille un lien réseau signifie mettre en veille ses deux interfaces réseau.

Notre méthode consiste à changer le routage pour pouvoir éteindre un nombre maximal de liens. En effet, les réseaux actuels sont très largement sur-dimensionnés (i) pour permettre de tolérer des pannes réseau et pouvoir re-router le trafic dans le cas où certaines routes ne sont plus disponibles (ii) pour pouvoir tolérer l'augmentation constante du trafic au cours du temps. Typiquement, un lien d'un réseau FAI est utilisé entre 30 et 50% de sa capacité. Il existe donc une marge importante pour re-router le trafic de façon à l'agréger sur un petit nombre d'équipements. Les équipements non-utilisés seront ensuite mis en veille. Autrement dit, ce problème revient à effectuer un routage des demandes en minimisant le nombre d'arêtes utilisées dans la topologie.

Nous définissons d'abord formellement le problème et nous le modélisons comme un programme linéaire à nombres entiers. Ensuite, nous prouvons que ce problème n'est pas dans APX, c'est-à-dire qu'il n'y a pas d'algorithme d'approximation à facteur constant en temps polynomial pour le résoudre, à moins que P=NP. Nous proposons un algorithme heuristique pour ce problème. Nous montrons ensuite le gain en termes de nombre d'interfaces réseau (conduisant à une réduction globale d'environ 33 MWh pour un réseau cœur de taille moyenne) pour un ensemble de technologies réseau existantes : nous constatons que pour presque toutes les topologies, plus d'un tiers des interfaces réseau peut être épargné. Enfin, nous discutons de l'impact du routage éco-énergétique sur la longueur des routes et sur la tolérance aux pannes.

Le reste du chapitre est organisé comme suit.

- Dans la section 3.1.3, nous présentons d'abord formellement le problème et nous le modélisons comme un programme linéaire en nombres entiers.
- Dans la section 3.1.4, nous rappelons la complexité de problèmes connexes et nous prouvons que le problème ne peut être approché à un facteur constant, même pour seulement deux demandes et si tous les liens ont la même capacité.
- Au vue de la difficulté du problème, nous proposons des algorithmes heuristiques pour pouvoir résoudre le problème sur des instances de tailles moyennes ou grandes dans la section 3.1.5.
- Nous étudions les gains énergétiques pour un ensemble de réseaux cœur existants. Nous exhibons qu'*au moins un tiers des interfaces réseau peuvent être désactivées* dans la section 3.1.6. Enfin, nous discutons l'impact de ces solutions éconergétiques sur la *longueur des routes* et la *tolérance aux pannes* du réseau.

3.1.2 Etat de l'art

Mesure des consommations d'énergie. Plusieurs campagnes de mesure de la consommation énergétique du réseau ont été réalisées au cours des dernières années, voir par exemple [337, 306] et [345]. Leurs auteurs affirment que la consommation des appareils réseau est en grande partie indépendante de leur charge. En particulier, dans [345], les auteurs s'intéressent à la consommation d'énergie des routeurs. Ils observent que pour la série populaire Cisco 12000, la consommation à une charge de 75% est seulement 2% de plus qu'à l'état de repos (770W vs. 755W). Dans [306], les auteurs montrent par l'expérimentation que l'énergie consommée dépend du nombre de ports actifs. Désactiver les ports inutilisés sur une carte de ligne réduisent la consommation d'énergie de l'appareil. Les valeurs obtenues au cours de l'expérimentation montrent que la consommation d'une linecard 4-port Gigabit ethernet (100W) est d'environ un quart de la consommation du système de base global (430W).

La minimisation de l'énergie. Dans [345], les auteurs modélisent ce problème comme un programme linéaire en nombres entiers. La fonction objectif est une somme pondérée du nombre de plateformes et d'interfaces. Ils montrent combien d'énergie peut être économisée sur différents réseaux avec ce modèle. Cependant, ils ne donnent pas d'intuitions, d'explications, ni de formules pour leurs résultats. Dans [284], les auteurs proposent un reroutage à différentes couches dans les réseaux IP-over-WDM pour faire des économies d'énergie tandis que [304] étudie l'impact de la technologie pour un routage économe en énergie. Dans [336, 371, 322], les chercheurs ont proposé des techniques telles que la mise en veille de sous-composants inactifs (cartes de ligne, ports, etc.), ainsi que l'adaptation du taux de transfert des paquets en fonction du trafic dans les réseaux locaux. Dans [290], les auteurs proposent une modulation des configurations radio dans les réseaux fixes sans fil à large bande pour réduire la consommation d'énergie.

3.1.3 Modélisation du problème

Le réseau est modélisé par un graphe pondéré $G = (V, E)$ avec c_e représentant la capacité de l'arête $e \in E$. L'ensemble des demandes est $D = \{\mathcal{D}_{st} > 0; (s,t) \in V \times V, s \neq t\}$ avec $\mathcal{D}_{st}$ le volume de trafic de s à t. Une demande $\mathcal{D}_{st}$ doit être routée via un chemin élémentaire de s à t. Un routage valide des demandes est un ensemble de chemins dans G pour chacune des demandes $\mathcal{D}_{st} \in D$ tel que pour chaque arête $e \in E$, le volume total de trafic passant par e n'excède pas la capacité c_e. Un problème de décision classique est de déterminer si un tel routage existe ou non. Formellement :

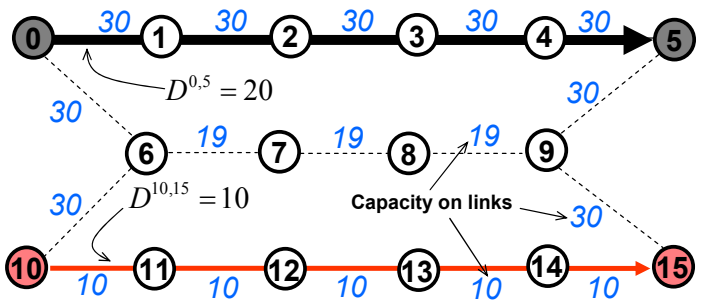
Définition 1. Etant donnés un graphe pondéré $G = (V, E)$ et un ensemble de demandes D, le Routing Problem consiste à décider si il existe un routage valide des demandes de D dans G.

Le problème que nous étudions a la contrainte supplémentaire de trouver un routage minimisant le nombre d'arêtes utilisées. En effet, nous étudions une architecture réseau simplifiée dans laquelle une liaison (A, B) relie deux routeurs A et B à travers 2 interfaces réseau, une pour A et une pour B. C'est pourquoi, afin de minimiser la consommation d'énergie du réseau, nous voulons désactiver les interfaces réseau. Le degré d'un nœud dans le graphique correspond au nombre d'interfaces réseau sur le routeur. Si une interface réseau est désactivée, l'autre interface à l'extrémité de la liaison n'est plus utile et peut également être désactivée. Par conséquent, l'objectif de notre problème est de minimiser le nombre de liens actifs dans le réseau. Formellement :

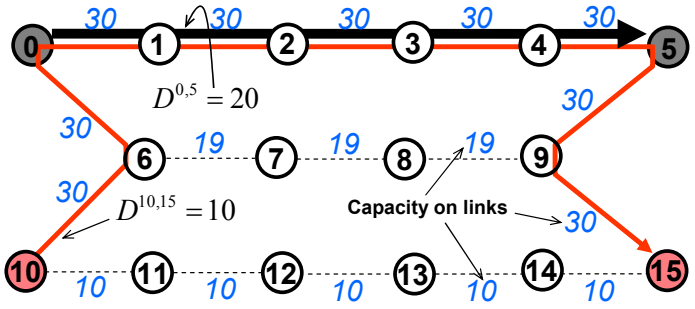
Définition 2. Etant donnés un graphe pondéré $G = (V, E)$ et un ensemble de demandes D, le Minimum Edge Routing Problem consiste à trouver un ensemble de cardinalité minimum $E^* \subseteq E$ tel que il existe un routage valide des demandes D dans $G^* = (V, E^*)$.

In Section 3.1.3, we present some simple instances of the Minimum Edge Routing Problem and the corresponding solutions. In Section 3.1.3, we describe a linear program for our problem.

Example. Considérons le graphe $G = (V, E)$ représenté en figure 3.1 composé de 16 sommets et de 17 arêtes. Les nombres entiers en bleu sur les arêtes représentent les capacités des liens réseau. Il y a deux demandes de trafic $0 \to 5$ et $10 \to 15$ avec des volumes $D^{0,5} = 20$ Gb et $D^{10,15} = 10$ Gb. Traditionnellement, les opérateurs réseau utilisent le routage par plus courts chemins pour minimiser la bande passante utilisée et les délais. Le routage par plus courts chemins, comme indiqué dans la figure 3.1a, utilise 10 liens actifs. Ainsi les 7 autres liens qui sont inactifs peuvent être mis en mode veille. Cependant, il existe des routages qui permettent de mettre plus de liens en veille. En particulier, la solution du Minimum Edge Routing Problem est donnée dans la Fig. 3.1b. Cette solution EAR permet de mettre 8 liens en veille, donc la consommation d'énergie est encore plus réduite.



(a) Shortest path routing: sleep 7 links



(b) EAR: sleep 8 links

Figure 3.1: Exemples, d'une part, d'un routage par plus courts chemins et, d'autre part, d'un routage économe en énergie, solution de MINIMUM EDGE ROUTING PROBLEM.

Programme linéaire en nombres entiers

Le MINIMUM EDGE ROUTING PROBLEM peut être modélisé comme un problème de flots *entier* à plusieurs commodités. Nous notons f_{uv}^{st} le volume de trafic de l'arête uv correspondant à la demande $\mathcal{D}_{st}$ passant de u vers v. Nous notons $f_{uv} = \sum_{st \in V \times V} f_{uv}^{st}$. Pour chaque arête $e \in E$, nous introduisons une variable binaire x_e selon si l'arête est utilisée ou non : $x_e = 0$ si $f_{uv} + f_{vu} = 0$ et $x_e = 1$ si $f_{uv} + f_{vu} > 0$.

La *fonction objectif* est alors

$$\min \sum_{e \in E} x_e$$

avec :

contraintes de flots : $\forall (s,t) \in V \times V, \forall u \in V,$

$$\sum_{v \in N(u)} f_{vu}^{st} - \sum_{v \in N(u)} f_{uv}^{st} = \begin{cases} -\mathcal{D}_{st} & \text{if} \quad u = s, \\ \mathcal{D}_{st} & \text{if} \quad u = t, \\ 0 & \text{otherwise.} \end{cases}$$

contraintes de capacité : $\forall e = (u,v) \in E,$

$$\sum_{d \in \mathcal{D}} \left(f_{uv}^{d} + f_{vu}^{d} \right) \leq x_e c_e.$$

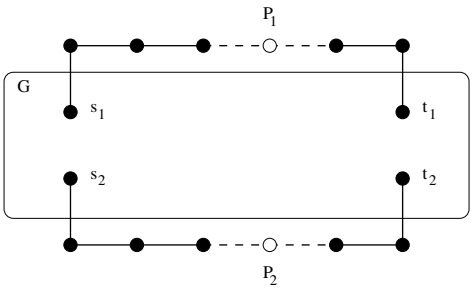


Figure 3.2: Construction de G' à partir de G utilisée dans la preuve du théorème 1.

Les contraintes de flots sont les contraintes usuelles de conservation de flots. Les contraintes de capacité indiquent que pour chaque arête $e \in E$, le montant total des demandes passant par e ne dépasse pas la capacité c_e.

3.1.4 Inapproximabilité

Le Minimum Edge Routing Problem présenté dans cet article est un cas particulier de différents problèmes d'optimisation de réseaux bien connus : routage à coût minimum [410], problème de flot minimum concave [400], problème de flot minimum avec des fonctions de coût en escalier [382]. En recherche opérationnelle, ce problème peut être considéré comme un cas particulier du problème Fixed charge transportation problem [412, 97], où le coût de l'unité de débit sur une arête est nul. Notez que ce problème est NP-Hard [400] : le nombre de sous-graphes possibles à tester est fortement exponentiel pour la plupart des graphes. De plus, même pour un sous-graphe donné (lorsque l'ensemble des arêtes à utiliser est donné), la faisabilité d'un problème de flots intégral multi-commodité doit être évaluée. Ce problème plus simple correspond au Routing Problem et il est connu pour être NP-complet même pour deux commodités [409]. Enfin, notez que c'est aussi le pire cas des fonctions en escalier, car la plupart des approximations par linéarisation sont très loin d'être une solution réalisable.

Nous prouvons maintenant que le Minimum Edge Routing Problem n'est pas dans APX (et donc c'est un problème NP-hard) même pour deux instances spéciales (Theorem 1 et Theorem 1 et Theorem 2). Cela signifie qu'il n'y a pas d'algorithme d'approximation à facteur constant en temps polynomial pour le Minimum Edge Routing Problem, à moins que $P = NP$.

Théorème 1. *Le problème* Minimum Edge Routing Problem *n'est pas dans APX même pour deux commodités.*

Démonstration. Soit $G = (V, E)$ un graphe pondéré non dirigé. Considérons le cas de deux commodités $D_{s_1t_1} > 0$ et $D_{s_2t_2} > 0$. Soit k un entier positif. Nous construisons un graphe $G'_k = (V', E')$ comme suit : nous commençons avec une copie de G ; nous ajoutons un chemin P_1 entre s_1 et t_1 composé de $x > k|E|$ arêtes et un chemin P_2 entre s_2 et t_2 également composé de $x > k|E|$ arêtes. Chaque arête de P_1 et P_2 a une capacité infinie. Nous considérons le même ensemble de demandes. G' est représenté dans la figure 3.2. Supposons maintenant qu'il y a une constante $k \geq 1$, telle qu'il existe un algorithme en temps polynomial trouvant un sous-ensemble $E^k \subseteq E$ assurant que $|E^k|/|E^*| \leq k$, où E^* est une solution optimale de Minimum Edge Routing Problem. Nous nommons E'^k la solution trouvée pour G' par cet algorithme et nous avons $|E'^k|/|E'^*| \leq k$. Si $|E'^k| \geq x$, cela signifie qu'il n'y a pas de solution pour le

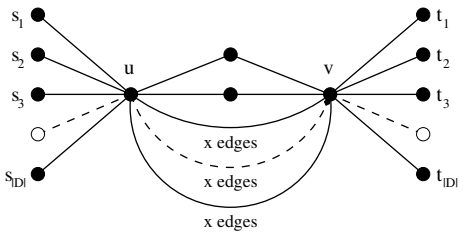


Figure 3.3: Le graphe $G = (V, E)$ utilisé dans la preuve du théorème 2. Chaque arête $e \in E$ a comme capacité $c_e = c$.

Routing Problem avec G. En fait, une telle solution utiliserait au plus $|E|$ arêtes de G, et une k-approximation utiliserait moins de x arêtes. Si $|E'^k| < x$, cela signifie qu'il existe une solution à ce problème. Ainsi, en utilisant cette construction, on obtient un algorithme en temps polynomial résolvant le Routing Problem avec G, c'est-à-dire un algorithme en temps polynomial pour décider s'il y a un routage des demandes D dans G respectant les contraintes de capacités. Une contradiction, à moins que $P = NP$. □

Théorème 2. *Le problème* Minimum Edge Routing Problem *n'est pas dans APX même quand chaque arête a une capacité fixée c.*

Démonstration. Supposons qu'il existe une constante $k \geq 1$, telle qu'il existe un algorithme en temps polynomial trouvant un sous-ensemble $E^k \subseteq E$ pour le problème Minimum Edge Routing Problem assurant que $|E^k|/|E^*| \leq k$. Rappelez-vous que E^* indique une solution optimale du problème Minimum Edge Routing Problem. Considérons le graphe pondéré non dirigé $G = (V, E)$ représenté dans la figure 3.3. Chaque arête $e \in E$ a une capacité $c_e = c$. Les $|D|$ demandes $D_{s_1t_1}, D_{s_2t_2}, \ldots, D_{s_{|D|}t_{|D|}}$ sont telles que $\sum_{i=1}^{|D|} D_{s_it_i} = 2c$. De u à v, il y a des chemins disjoints composés de 2 arêtes chacun et $|D|$ chemins disjoints avec $x > k|E|$ arêtes chacun. A cause de la k-approximation, s'il y a une solution n'utilisant pas de longs chemins, alors notre algorithme en temps polynomial renvoie nécessairement une telle solution. Sinon, il retourne une solution avec des arêtes appartenant à ces chemins. Notez que le problème de trouver s'il existe une partition des requêtes en deux ensembles de même poids c est un problème NP-complet (Partition Problem). Notre algorithme d'approximation en temps polynomial le résoudrait donc, une contradiction, à moins que $P = NP$. □

Ces deux résultats négatifs motivent la conception d'algorithmes heuristiques pour le Minimum Edge Routing Problem dans la Section 3.1.5 et l'étude des limites théoriques pour des instances particulières dans [ch5] qui donnent des informations pour les réseaux généraux.

3.1.5 Heuristique

Nous proposons dans cette section une heuristique pour le Minimum Edge Routing Problem. Less Loaded Edge Heuristic commence par trouver un routage valide dans $G = (V, E)$ en utilisant une heuristique routant les demandes de maniète gloutonne par des plus courts chemins. La métrique utilisée est $c(e)/r(e)$ où $r(e)$ désigne la capacité résiduelle. Initialement

$r(e) = c(e), \forall e \in E$. L'idée est ensuite de supprimer l'arête $e \in E$ dont le ratio $c(e)/r(e)$ est le plus petit possible (dans un certain sens l'arête qui est la moins chargée). Nous recalculons un routage valide et procédons de la même façon pour diminuer le nombre d'arêtes. Si aucun routage valide n'est trouvé nous réinsérons l'arête supprimée et continuons le processus de suppression des arêtes. Notons qu'une arête ne peut pas être supprimée deux fois. Nous obtenons finalement un ensemble d'arêtes E' et un routage valide dans $G' = (V, E')$.

La seconde heuristique, RANDOM HEURISTIC, est utilisée comme base de comparaison pendant les simulations. La seule différence avec la première est que les liens à supprimer sont sélectionnés uniformément au hasard (et non pas en fonction de leur charge). Le routage est exécuté de la même manière que pour LESS LOADED EDGE HEURISTIC.

Algorithm 2.1 LESS LOADED EDGE HEURISTIC

Require: Un graphe non dirigé pondéré $G = (V, E)$ dans lequel chaque arête $e \in E$ a une capacité initiale c_e et une capacité résiduelle r_e (qui dépend des demandes utilisant e). Un ensemble de demandes $\mathcal{D}$ et chaque demande a une charge $\mathcal{D}_{st}$.

$\forall e \in E, r_e = c_e$
Calculer un routage réalisable des demandes avec l'algorithme 2.2
while Arêtes peuvent être enlevées **do**
  Enlever l'arête e' de valeur $\frac{c(e')}{r(e')}$ minimum qui n'a pas déjà été choisie.
  Calculer un routage réalisable avec l'algorithme 2.2
  Si aucun routage réalisable n'existe, alors remettre e' dans G
end while
renvoyer le sous-graphe G.

Algorithm 2.2 Heuristique pour le ROUTING PROBLEM

Require: Un graphe non dirigé pondéré $G = (V, E)$ dans lequel chaque arête $e \in E$ a une capacité initiale c_e et une capacité résiduelle r_e (qui dépend des demandes utilisant e). Un ensemble de demandes $\mathcal{D}$ et chaque demande a une charge $\mathcal{D}_{st}$.

Trier les demandes uniformément au hasard
while $\mathcal{D}_{st}$ est une demande de D sans encore de routage **do**
  Calculer un plus court chemin SP_{st} selon la métrique $\frac{c_e}{r_e}$ sur les arêtes
  Affecter la route SP_{st} à la demande $\mathcal{D}_{s,t}$
  $\forall e \in SP_{st}, r_e = c_e - \mathcal{D}_{st}$
end while
renvoyer le routage (s'il existe) affecté au demande de D.

Les deux heuristiques sont évaluées par simulation en section 3.1.6 et comparées aux solutions données par le programme linéaire de la section 3.1.3.

3.1.6 Résultats pour des réseaux cœur

Nous présentons dans cette section les gains énergétiques potentiels dans des scénarios pratiques. Nous regardons combien d'interfaces peuvent être économisées pour dix réseaux classiques et pour différentes ratios capacité/demande λ. Nous étudions ensuite l'impact de ces routages efficaces en énergie sur la longueur des routes et la tolérance aux pannes.

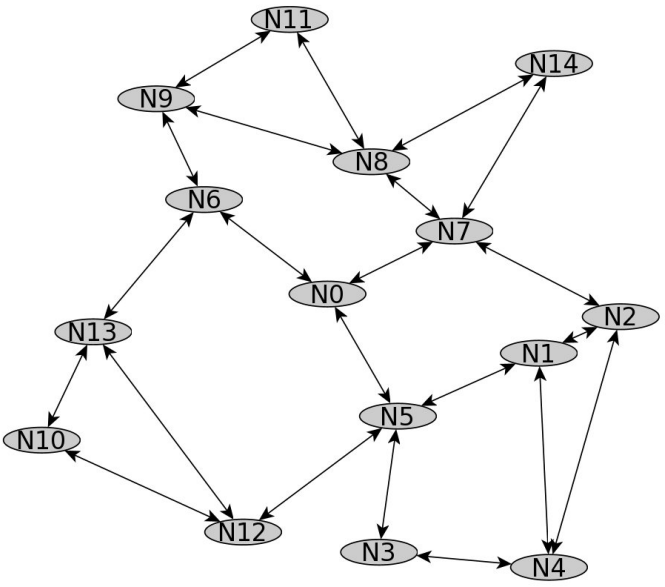


Figure 3.4: Réseau *Atlanta* avec 15 sommets and 22 arêtes.

SNDLib Topologies

Topologies. Nous utilisons dix réseaux extraits de la librairie SNDLib (http://sndlib.zib.de). Ces réseaux correspondent à des topologies US (*Atlanta*, voir la figure 3.4), européennes (*Nobel EU, Cost266*) ou nationales (*Nobel Germany, France*). Leurs tailles s'étendent de 15 à 54 sommets et de 22 à 89 arêtes, comme résumé dans le tableau 3.1. Pour ces 10 topologies, nous avons calculé des routages éconergétiques pour différents ratios capacité/demande λ. Le plus petit ratio considéré, λ_1, correspond à la capacité minimale pour router toutes les demandes en utilisant toutes les arêtes. Ce ratio correspond à un réseau sans sur-dimensionnement, c'est-à-dire pour lequel OF=1 (pour Overprovisioning Factor). Le plus grand ratio considéré est λ_{tree} et correspond à la capacité nécessaire pour router sur un sous-graphe minimum, à savoir un arbre couvrant (dans ce cas, $OF = \frac{\lambda_1}{\lambda_{\text{tree}}}$).

Validation des heuristiques. Nous n'avons pu exécuter le programme linéaire en nombres entiers que pour le plus petit réseau, *Atlanta*, dont les résultats sont présentés dans la figure 3.5. Nous voyons que Less Loaded Edge Heuristic fonctionne bien, atteignant la valeur optimale la plupart du temps, quand l'heuristique aléatoire a besoin d'un plus grand λ pour atteindre la même valeur. En fait, CPLEX prend déjà plusieurs heures sur *Atlanta* pour résoudre le problème pour un rapport capacité/demande. Nous présentons donc les résultats trouvés par l'heuristique (qui ne prend que des dizaines de ms) dans les tableaux 3.1 et 3.2.

Économie d'interfaces réseaux. Nous donnons le pourcentage d'interfaces économisées pour différentes valeurs de OF dans le tableau 3.1 (a). Un facteur de 1 signifie que nous utilisons le rapport minimum capacité/demande nécessaire pour acheminer toutes les demandes (correspondant à la valeur λ_1), alors que, par exemple, un facteur de 2 signifie que nous avons le double de la valeur λ_1. Notez que dans la plupart des réseaux cœur d'aujourd'hui, le sur-dimensionnement est fortement utilisé car c'est un moyen efficace et facile de fournir une protection contre les pannes : les liens sont souvent utilisés entre 30 et 50 % de leur capacité.

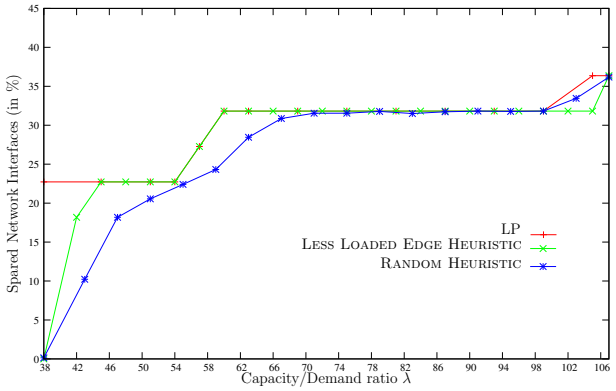


Figure 3.5: Interfaces réseau économisées (en %) pour le réseau *Atlanta*.

			Facteur de surdimensionnement OF				Arbre	
	$\|V\|$	$\|E\|$	1	2	3	4	OF	Gain
Atlanta	15	22	0%	32%	36%	36%	2.66	36%
New York	16	49	2.0%	59%	63%	67%	5.2	69%
Nobel Germany	17	26	0%	35%	39%	39%	2.75	39%
France	25	45	0%	42%	44%	47%	3.13	47%
Norway	27	51	12%	43%	47%	47%	4.71	49%
Nobel EU	28	41	12%	32%	34%	34%	2.76	34%
Cost266	37	57	3.5%	32%	35%	37%	3.68	37%
Giul39	39	86	0%	45%	50%	52%	8.25	56%
Pioro40	40	89	0%	53%	54%	55%	5.12	56%
Zib54	54	80	0%	30%	33%	33%	4.71	34%

Table 3.1: Interfaces économisées (en %) selon la valeur du facteur de surdimensionnement OF et (b) Évaluation de λ_1 et λ_{tree}.

Tout d'abord, nous notons que sur certaines des dix topologies, dès que le routage est faisable ($OF = 1$), certaines interfaces réseau peuvent déjà être désactivées (12% pour *Norway* et *Nobel EU*). En effet, dans ce cas, les arêtes importantes du réseau (les arêtes dans la coupe minimale par exemple) sont pleinement utilisées, mais en même temps les arêtes à la périphérie sont moins utilisées et certaines peuvent être désactivées. Avec un facteur de sur-dimensionnement de 2, environ un tiers des arêtes peuvent être épargnées (et même 53% pour le réseau *Pioro40*).Avec des facteurs plus importants (3 ou 4), le gain n'est pas aussi important, mais certaines interfaces réseau peuvent être mises en veille (par exemple 36 % pour Atlanta). Nous montrons dans le tableau 3.1 (a), dans les 2 dernières colonnes, la valeur de OF pour laquelle l'arbre est atteint ainsi que le pourcentage d'interfaces réseau mise en veille (SNE). Les valeurs sont directement liées à la densité du réseau. Par exemple, la Norvège a besoin d'un facteur de $4,71$ ($\lambda_{\text{tree}} = 4,71 \times \lambda_1$) pour atteindre l'arbre avec des 26 liens (au lieu de 51), épargnant 49% arêtes. Par conséquent, plus la densité est grande, plus le nombre d'interfaces réseau qui peuvent être désactivées est grand.

En conclusion, pour tous les réseaux étudiés, *entre un tiers et la moitié des interfaces réseau*

	Sur-dimensionnement			
	1	2	3	λ_{arbre}
Atlanta	1.00	1.19	1.25	1.25
New York	1.01	1.24	1.26	1.32
Nobel Germany	1.00	1.11	1.18	1.18
France	1.00	1.10	1.12	1.16
Norway	1.02	1.17	1.18	1.25
Nobel EU	1.08	1.14	1.24	1.25
Cost266	1.04	1.11	1.19	1.32
Giul39	1.00	1.18	1.21	1.50
Pioro40	1.00	1.25	1.32	1.42
Zib54	1.00	1.02	1.07	1.11

(a) Longueur des routes

	Sur-dimensionnement		
	1	2	3
Atlanta	2.35	1.09	1.00
New York	4.90	1.24	1.19
Nobel Germany	2.35	1.04	1.00
France	2.48	1.02	1.01
Norway	2.61	1.14	1.04
Nobel EU	1.82	1.07	1.00
Cost266	2.47	1.12	1.07
Giul39	3.68	1.41	1.14
Pioro40	4.06	1.12	1.09
Zib54	2.16	1.05	1.01

(b) Tolérance aux pannes

Table 3.2: Impact du routage efficace en énergie sur (a) la longueur des routes (facteur de stretch multiplicatif moyen) et sur (b) la tolérance aux pannes du réseau (nombre moyen de chemins disjoints).

peuvent être épargnés pour les facteurs habituels de sur-dimensionnement. De plus, lorsque le meilleur routage ne peut être trouvé par le programme linéaire en nombres entiers (grandes topologies), *l'heuristique proposée se rapproche des solutions optimales.*

Impact sur le réseau

Nous pensons que les opérateurs de réseaux ne mettront en œuvre un routage économe en énergie que si l'impact sur les autres paramètres est limité. Nous discutons ici de l'impact sur la longueur des routes et sur la tolérance aux pannes de l'heuristique que nous proposons.

Impact sur la longueur des routes. Lorsque nous éteignons certains composants d'un réseau, nous économisons de l'énergie mais, en même temps, nous routons les demandes sur des chemins plus longs. Le *stretch multiplicatif* est défini comme le ratio entre la longueur moyenne des routes dans le nouveau routage divisé par la longueur moyenne des routes avec l'ancien routage (en utilisant toutes les arêtes). Les résultats pour les topologies SNDLib sont donnés dans le tableau 3.2 (a). Le stretch d'une valeur 1 correspond aux cas où aucune arête n'a pu être mise en veille et donc le routage n'est pas affecté. Nous constatons que, comme prévu, la tendance générale est que lorsque le facteur de sur-dimensionnement augmente, les trajectoires s'allongent.

Néanmoins, nous remarquons que les routages obtenus ont un impact limité sur l'augmentation de la longueur des routes. Pour *Zib54*, l'accroissement est de 11% pour le cas extrême où le routage se fait avec un arbre (34% d'économie). En général, l'augmentation pour ce cas extrême varie de 11% à 50% avec une moyenne de 27%. Concernant le réseau le plus impacté *Giul39* avec un accroissement de 50% de la longueur des routes, l'économie est de 56%. Nous observons qu'une économie d'interfaces de 45% est atteinte pour $OF = 2$ avec une augmentation de la longueur des routes de seulement 18%.

Impact sur la tolérance aux pannes. Nous mesurons dans le tableau 3.2 (b) l'impact sur la tolérance aux pannes en calculant *le nombre moyen de chemins disjoints* entre deux sommets. Ce nombre varie de 1.82 à 4.90 pour les réseaux complets. En routant les demandes via un arbre, ce nombre est 1 par définition (une seule route existe entre n'importe quelle paire de sommets).

Cette valeur est presque atteinte avec $OF = 3$. Nous observons enfin que cette diminution est rapide car pour les dix réseaux, cette valeur est inférieure à 1.41 pour $OF = 2$.

Discussion sur la technologie nécessaire. Dans le cas d'une panne entre deux routeurs, il est possible que toutes les demandes ne puissent plus être routées sur les interfaces active. Il peut ainsi être nécessaire d'activer certaines interfaces réseau pour trouver un nouveau routage pour certaines demandes. Ainsi, cette étude montre que l'utilisation de ces solutions d'efficacité énergétique est conditionnée par l'existence de technologies permettant une mise en marche rapide des interfaces réseau. Les fabricants d'interfaces réseau ont travaillé et travaillent à la conception de ce type d'interfaces [334]. D'autre part, l'arrivée de nouvelles technologies comme les réseaux logiciels devraient permettre de réagir très rapidement aux pannes et donc permettre la mise en place de ces solutions éco-énergétiques, comme étudié au chapitre 8.

En attendant la mise en place de ces nouveaux matériels ou technologies, nous avons aussi proposé dans [ch5] une solution pour un routage tolérant aux pannes. L'idée est d'utiliser des sous-graphes de liens actifs pour lesquels il est possible de trouver deux chemins disjoints par demande. Par conséquent, l'impact des défaillances de liaisons sur le réseau sera réduit parce qu'un chemin de protection avec une capacité suffisante sera disponible.

3.1.7 Conclusion et perspectives

Dans ce travail, nous présentons à travers une architecture simplifiée le problème de la minimisation de la consommation d'énergie dans les réseaux. Nous montrons les résultats de non-approximation. Les simulations sur des topologies réelles montrent que le gain d'énergie est significatif lorsque certaines interfaces réseaux peuvent être désactivées.

- *Au moins un tiers des interfaces réseau peuvent être mises en veille pour les valeurs de demandes habituelles.*

- Pour un réseau cœur de taille moyenne, cela conduit à une réduction de la consommation d'énergie d'environ 33 millions de dollars par an (pour *Cost266* avec 37% d'interfaces mises en veille). Pour cette estimation, nous considérons un scénario où les interfaces désactivées de notre architecture simplifiée sont des ports ethernet 4-Gb. Nous pensons que cela correspond à une capacité raisonnable pour les réseaux cœurs. Nous utilisons les valeurs de consommation données dans [345] : 100 W pour ces interfaces.

- La longueur des routes augmente, mais pas trop : en moyenne 27 % pour presque toutes les topologies étudiées.

- La *tolérance aux pannes* peut être réalisée avec l'utilisation de technologies de mise en marche rapide *quick switching-on* ou en ajoutant des contraintes de chemins disjoints au problème.

Dans le cadre de travaux futurs, nous prévoyons tout d'abord d'étudier une fonction de coût plus détaillée correspondant à une architecture de routeur plus complexe. C'est en particulier fait dans la section 8.3 du chapitre 8.

D'un point de vue pratique, nous aimerions mener des expériences en laboratoire sur de petites topologies de réseau pour mesurer dans la pratique la performance du routage efficace en énergie proposé. Les questions sont multiples : déploiement, passage d'un routage à un autre sans perte de paquets, modification des protocoles de routage, tolérances aux pannes... Au cours de ces dernières années, j'ai étudié un certain nombre de ces questions :

- Nous avons effectué différentes expérimentations et mesures, dont une est présentée dans la section suivante, la section 3.2.
- Nous avons étudié la mise en pratique des solutions efficaces en énergie tout en évitant des écueils discutés ci-dessus, en utilisant les nouvelles technologies des réseaux logiciels dans la section 8.4.

D'un point de vue théorique, il existe de nombreux défis, en particulier :

- Le problème de déterminer le meilleur arbre couvrant (et de trouver la valeur λ_{tree}), qui d'un premier abord pourrait sembler simple est en fait NP-complet. Cela se prouve de façon analogue aux théorèmes 1 et 2. Mais, nous aimerions savoir s'il est possible d'obtenir une approximation de λ_{tree} à une constante multiplicative près, c'est-à-dire si ce problème est dans APX.
- Plus généralement, est-il possible de concevoir des méthodes pour trouver les meilleurs sous-graphes de topologies particulières ? Je me suis attaqué à ce sujet qui est le thème du chapitre 4.

3.2 Utiliser l'élimination de redondance

Ce travail a été effectué en collaboration avec Joanna Moulierac, Truong Khoa Phan, ancien doctorant de COATI et maintenant Research Associate dans le Department of Electronic and Electrical Engineering, University College London (UCL), UK, et Frédéric Roudaut, Orange Labs, Sophia Antipolis, France. Ce travail a été fait en particulier dans le cadre du Network Boost Project (Orange Labs). Il a donné lieu aux publications [Ci50, J17].

Récemment, *le routage efficace en énergie* (EAR) a gagné en popularité dans le milieu de la recherche en réseaux. L'idée est que les demandes de trafic sont redirigées sur un sous-ensemble des liens du réseau, ce qui permet à d'autres liens d'être mis en veille afin d'économiser de l'énergie. Dans cet article, nous proposons GreenRE - un nouveau modèle EAR avec le support de l'élimination de la redondance des données (RE). Cette technique, activée au sein des routeurs, peut virtuellement augmenter la capacité des liens réseau. Basé sur des expériences réelles sur une plateforme Orange Labs, nous montrons que l'utilisation de la RE augmente la consommation d'énergie des routeurs. Par conséquent, il est important de déterminer quels routeurs doivent activer la RE et quels liens doivent êtres mis en mode veille afin de minimiser la consommation d'énergie du réseau. Nous modélisons le problème en tant que Programme Linéaire en Nombres Entiers et proposons des algorithmes heuristiques gloutons pour les grands réseaux. Des simulations sur plusieurs topologies de réseaux montrent que le modèle GreenRE peut réaliser 37 % d'économies d'énergie supplémentaires par rapport au modèle EAR classique.

3.2.1 Introduction

En général, la capacité des liens est la principale contrainte du problème EAR. Dans ce travail, nous partons de l'hypothèse que les routeurs peuvent éliminer le trafic de données redondantes et, par conséquent, augmenter virtuellement la capacité des liaisons réseau. Par conséquent, un plus grand nombre de flux de trafic peut être rerouté et un plus grand nombre de liens peuvent être mis en veille pour économiser de l'énergie. Bien qu'aujourd'hui les routeurs ne peuvent pas supprimer directement le contenu répété des transferts réseau, il existe un dispositif commercial utilisé dans les entreprises ou les petits FAI pour éliminer la redondance du trafic, le WAN Optimization Controller (WOC), [101, 350, 94]. Afin d'identifier la consommation d'énergie directement induite par la RE, nous réalisons des expériences pratiques sur le WOC. Parce que l'idée principale des routeurs exécutant la RE est similaire à la fonctionnalité WOC (voir Section 2.2), nous pensons que lorsqu'un routeur élimine la redondance du trafic, il consomme également de l'énergie supplémentaire comme le WOC. En résumé, les contributions de ce travail sont les suivantes :

- Nous avons effectué des expériences pratiques pour déterminer la consommation énergétique d'un WOC.
- Nous définissons et formulons GreenRE - un nouveau modèle EAR en tant que Programme Linéaire en Nombres Entiers (PLNE).
- Nous proposons et évaluons un algorithme heuristique glouton qui peut être utilisé pour les réseaux à grandes échelles.
- Par simulation, nous présentons les économies d'énergie sur des topologies réelles de réseaux.

Le reste de ce document est structuré comme suit. Nous résumons les travaux connexes dans la section 2. Dans la section 3, nous modélisons GreenRE comme ILP, puis nous proposons un algorithme heuristique gourmand. Les résultats de la simulation sont présentés à la section 4. Enfin, nous concluons le travail à la section 5.

3.2.2 Etat de l'art

Routage efficace en énergie classique (EAR)

Le problème EAR de la minimisation du nombre de liens actifs sous contraintes de QoS peut être formulé avec précision à l'aide de la programmation linéaire en nombres entiers (PLNE). Cependant, ce problème est connu pour être NP-Hard [Ci56], et actuellement des solutions exactes ne peuvent être trouvées que pour les petits réseaux. Par conséquent, de nombreux algorithmes heuristiques ont été proposés pour trouver des solutions admissibles pour les grands réseaux [Ci56, 252].

Réduction de la charge réseau

Le trafic Internet présente une grande quantité de redondance lorsque différents utilisateurs accèdent au même contenu ou à des contenus similaires. Par conséquent, plusieurs travaux [330, 329, 279] ont exploré comment éliminer la redondance du trafic sur le réseau. Spring et al. [379] a développé le premier système pour supprimer les octets redondants de tout flux de trafic. Suite à cette approche, plusieurs fournisseurs commerciaux ont introduit le WAN Optimization Controller (WOC) - un dispositif qui peut supprimer le contenu dupliqué des transferts réseau [101, 350, 94]. Les WOC sont installés sur les sites individuels de petits ISP ou d'entreprises pour offrir des RE de bout en bout entre des paires de sites. Comme le montre la figure 3.6,

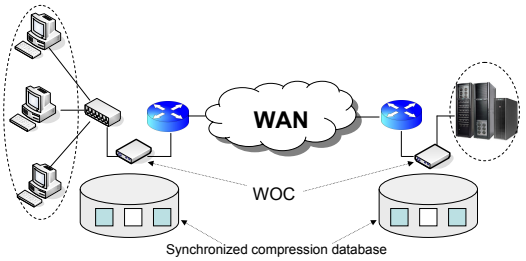


Figure 3.6: Reduction of end-to-end link load using WOC

les motifs de données précédemment envoyés sont stockés dans la base de données des WOC du côté émetteur et du côté récepteur. Une technique utilisée pour synchroniser les bases de données lors du peering WOCs peut être trouvée dans [350]. Chaque fois que le WOC du côté émetteur remarque le même motif de données provenant des hôtes émetteurs, il envoie une petite signature à la place des données d'origine (processus d'encodage). Le WOC récepteur récupère ensuite les données d'origine en recherchant la signature dans sa base de données (processus de décodage). Comme la taille des signatures n'est que de quelques octets, l'envoi de signatures au lieu de données réelles permet de réaliser d'importantes économies de bande passante.

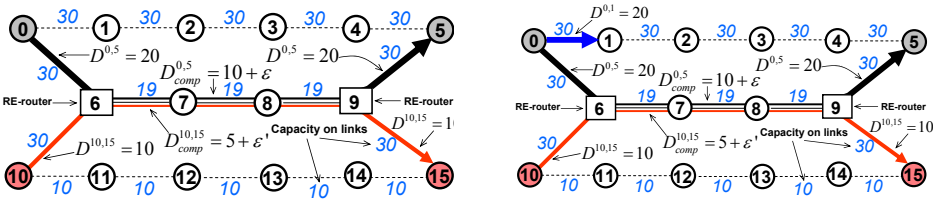
Récemment, le succès du déploiement des WOCs a motivé les chercheurs à explorer les avantages du déploiement des RE dans les routeurs sur l'ensemble de l'Internet [330, 329, 329, 279]. Les techniques de base utilisées ici sont similaires à celles utilisées par le WOC : chaque routeur sur le réseau dispose d'un cache local pour stocker les données envoyées précédemment et

utilisées pour encoder et décoder les paquets de données plus tard. Évidemment, cette technique nécessite de lourds calculs et une grande mémoire pour le cache local. Cependant, Anand et al. ont montré que sur un CPU de bureau de 2,4 GHz avec 1 Go de RAM, le prototype peut fonctionner à 2,2 Gbps pour l'encodage et à 10 Gbps pour le décodage des paquets [329]. De plus, ils croient qu'un débit plus élevé peut être atteint si le prototype est implémenté en dur (hardware). Plusieurs traces de trafic réel ont été recueillies pour montrer que jusqu'à 50 % de la charge de trafic peut être réduite avec le support RE [330, 329, 329, 279].

Dans la section suivante, nous proposons GreenRE - le premier modèle de routage de *routage efficace en énergie* avec le support RE. Nous montrons que RE, qui a été initialement conçue pour économiser de la bande passante, est également susceptible de réduire la consommation d'énergie du réseau.

3.2.3 Routage efficace en énergie classique avec RE

Dans le modèle GreenRE, la RE est utilisée pour augmenter virtuellement la capacité des liaisons réseau. Un inconvénient est que, comme indiqué dans [Ci50], quand un routeur exécute RE, il consomme plus d'énergie que d'habitude. Ceci introduit un compromis entre l'activation de RE sur les routeurs et la mise en veille des liens. Nous montrons que c'est une tâche non triviale de trouver quels routeurs doivent effectuer les RE et quelles liaisons doivent être mises en veille pour minimiser la consommation d'énergie d'un réseau coeur



(a) sleep 10 links, enable 2 RE-routers (b) sleep 9 links, enable 2 RE-routers

Figure 3.7: GreenRE avec 50% de trafic redondant.

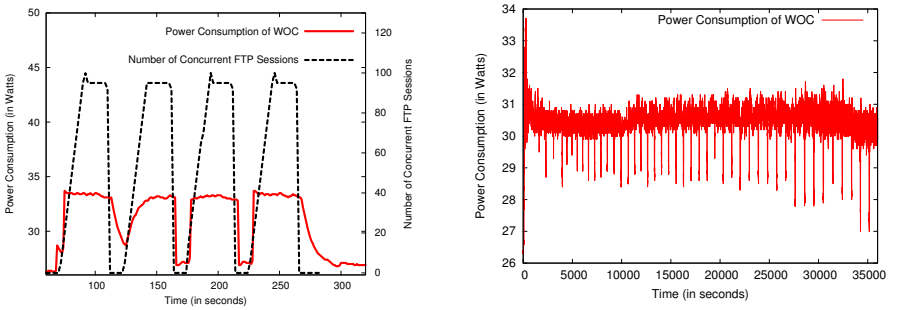
Pour un exemple, nous nous référons à la figure 3.7a avec deux demandes de trafic $D^{0,5} = 20$ Gb et $D^{10,15} = 10$ Gb. Nous considérons qu'un routeur RE coûte 30 Watts [Ci50] et qu'un lien consomme 200 Watts [252]. Supposons que 50% du trafic est redondant et que le service RE est activé au niveau du routeur 6 et du routeur 9, donc les flux de trafic sont de $0 \rightarrow 5$ et $10 \rightarrow 15$ pour les liens de passage (6, 7, 8, 9) sont réduits à $(10 + \epsilon)$ Gb et $(5 + \epsilon')$ Gb où ϵ, ϵ' désignent la taille totale des signatures utilisées pour chaque flux. En réalité, chaque signature n'a que quelques octets de taille [350], donc ϵ, ϵ' est petit et le routage représenté dans la figure 3.7a est réalisable sans congestion. En conséquence, la solution GreenRE permet de mettre en veille 10 liens avec 2 routeurs RE qui économisent $(10 \times 200 - 2 \times 30) = 1940$ Watts, comparé à $8 \times 200 = 1600$ Watts de la solution EAR (Fig. 3.1b). Il est à noter que, dans certains cas extrêmes, GreenRE aide même à trouver une solution de routage réalisable alors qu'elle est impossible pour l'EAR classique. Par exemple, si nous ajoutons une troisième demande du routeur 0 à 1 avec un volume de 20 Go, alors figures 3.7b est une solution réalisable. Cependant, sans routeurs RE, aucune solution réalisable n'est trouvée parce qu'il n'y a pas assez de capacité pour acheminer les trois demandes.

Pour résoudre le problème, nous avons proposé une formulation ILP et un algorithme heuristique dans [Ci50, J17].

3.2.4 Expérimentation et résultats de simulation

Consommation énergétique avec les WOCs

Plusieurs résultats d'économies de bande passante en utilisant les WOCs peuvent être trouvés dans [350]. Nous avons également réalisé des expériences sur la plate-forme réseau du projet Network Boost d'Orange Labs (la réprésentation de la plateforme de test se trouve dans [95]). Nous avons installé deux WOCs, chacun au lien d'accès des deux sites (appelons-les site A et site B). Ces deux sites sont reliés par un backbone composé de 4 routeurs. Nous établissons des connexions FTP pour télécharger des fichiers du site A vers le site B. Comme le montre la



(a) With concurrent FTP sessions (b) With increasing size of uploaded files

Figure 3.8: Consommation énergétique avec les WOCs.

figure 3.8a, la consommation d'énergie du WOC est augmentée (de 26 Watts à 34 Watts) avec le nombre de sessions FTP simultanées. Pour l'expérience suivante, nous ne gardons qu'une seule session FTP et laissons le WOC exécuter RE pendant 10 heures pendant lesquelles les tailles des fichiers téléchargés sont augmentées. Les résultats montrent que le WOC consomme environ 30 watts en moyenne (Fig. 3.8b). Par conséquent, par souci de simplicité, nous utilisons une valeur moyenne de la consommation d'énergie (30 watts) pour représenter le coût supplémentaire que le routeur doit supporter pour réaliser la fonction RE.

Simulation Results with GreenRE

Nous résolvons le modèle GreenRE avec le solver IBM CPLEX 12.4 [98]. Tous les calculs ont été effectués sur un processeur Intel Core i7 de 2,7 Ghz avec 8 Go de RAM. Nous avons étudié dix topologies de réseaux réels classiques extraites de SNDLib [280]. Leurs tailles vont de 15 à 54 sommets et de 22 à 89 liens, comme le résume le tableau 1. Selon les résultats des travaux mentionnés dans la section 2.2, nous utilisons un facteur de non-redondance $\gamma = 50\%$ pour toutes les simulations. Dans le pire des cas et à des fins de comparaison avec les travaux antérieurs [Ci56, Ci50], tous les liens ont la même capacité C et les demandes sont de tous vers tous (ou *all-to-all*, c'est-à-dire qu'un routeur doit envoyer du trafic à tous les routeurs restants sur le réseau) avec le même volume de trafic D pour chaque demande.

Comparaison avec l'heuristique H-GreenRE de [Ci50] Nous proposons dans cet article une nouvelle heuristique basée sur la formulation ILP appelée H_{ILP}-*GreenRE*. Pour comparer avec l'heuristique *H-GreenRE* proposée dans le travail précédent [Ci50], deux scénarios de simulation ont été réalisés pour les dix topologies de réseaux (nous trions les réseaux par ordre croissant du nombre de sommets).

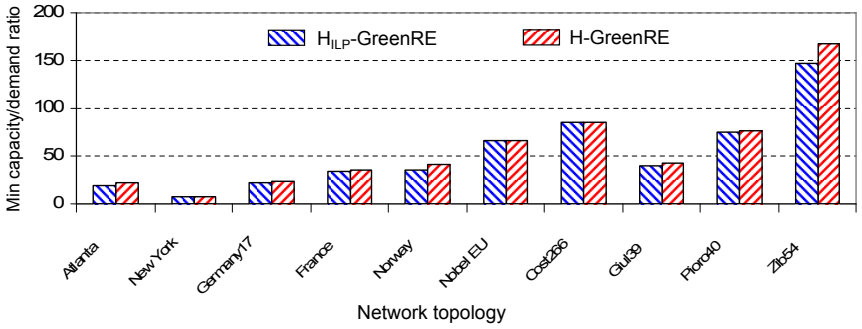


Figure 3.9: Comparaison de λ_{minRE} entre H_{ILP}-GreenRE et H-GreenRE

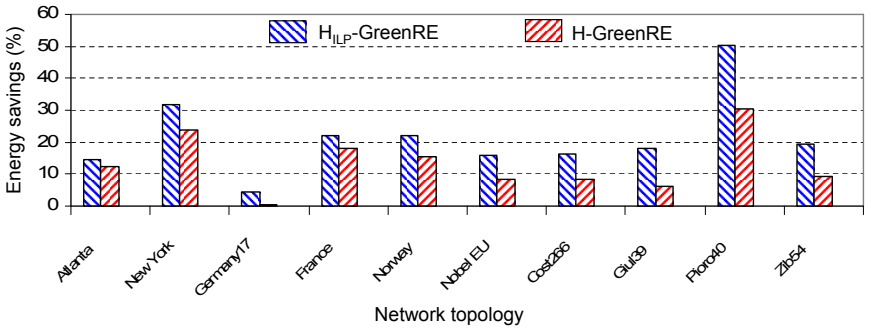


Figure 3.10: Comparaison des économies d'énergie de H_{ILP}-GreenRE et H-GreenRE

Tout d'abord, nous trouvons les valeurs minimales du rapport capacité/demande λ_{minRE} qui permettent à chaque algorithme heuristique de trouver une solution de routage réalisable *avec le support des routeurs RE*. Notez que, λ représente le niveau de charge de trafic sur le réseau. Une petite valeur de λ signifie que la charge de trafic sur le réseau est élevée (par exemple le trafic aux heures de pointe), il est donc difficile de trouver une solution réalisable en raison du manque de capacité (voir l'exemple dans la Fig. 3.7b). Par conséquent, l'algorithme heuristique qui peut trouver un routage réalisable avec une valeur plus petite de λ_{minRE} est le meilleur. La figure 3.9 montre que H_{ILP}-GreenRE peut trouver des solutions réalisables avec des valeurs plus petites de λ_{minRE} que H-GreenRE. Par exemple, pour le réseau d'Atlanta, H_{ILP}-GreenRE trouve une solution avec λ_{minRE}.19 lorsque H-GreenRE est avec λ_{minRE}.22 : c'est-à-dire, par exemple, pour une capacité de liens de 10 Gbit/sec, la première heuristique réussit à router une demande globale de 10/19 =0.53 Gbit/sec pour chaque demande et la seconde heuristique, une demande de seulement 10/22 = 0.45 Gbit/sec. En résumé, H_{ILP}-GreenRE trouve des solutions réalisables proches des limites inférieures de λ_{min} trouvées dans [Ci56]. La plus grande amélioration est sur le réseau Zib54 : $\lambda_{minRE} = 147$ (pour H_{ILP}-GreenRE) en comparaison avec $\lambda_{minRE} = 168$ (pour H-GreenRE).

Nous montrons ensuite les économies d'énergie pour les dix réseaux. Nous utilisons la valeur de λ_{minRE} qui permet à H-GreenRE de trouver une solution de routage réalisable pour chaque réseau (la deuxième colonne de la Fig. 3.9). Si un réseau a des liens denses, il y a plus de chances de pouvoir rediriger le trafic et de mettre des liens en mode veille, ce qui permet d'économiser plus d'énergie. Comme le montre la figure 3.10, H_{ILP}-GreenRE surpasse de nouveau H-GreenRE pour tous les réseaux. L'efficacité énergétique peut être augmentée de 2% (réseau Atlanta) à 19.8% (réseau Pioro40).

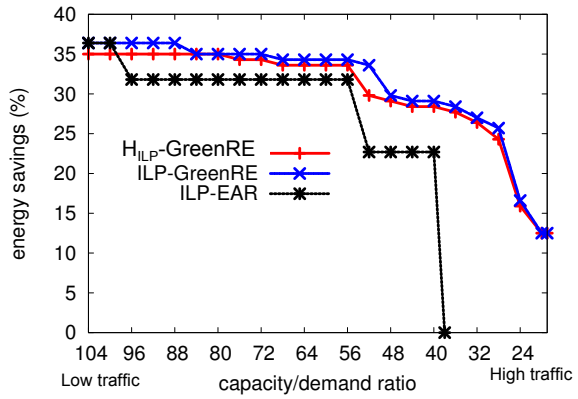


Figure 3.11: Résultats de simulation pour le réseau Atlanta

Economie d'énergie pour le réseau Atlanta Nous présentons dans la figure 3.11 les résultats de simulation pour le réseau Atlanta : solutions optimales *sans* routeurs RE (ILP-EAR) donnés par [Ci56], les résultats ILP avec les routeurs RE (ILP-GreenRE) et ceux de l'heuristique avec les routeurs RE (H_{ILP}-GreenRE). Comme CPLEX prend plusieurs heures pour trouver une solution optimale, nous limitons le temps de résolution à une heure et toutes les meilleures solutions se situent à moins de 10% de l'optimal. L'heuristique est assez rapide, il faut moins de 10 secondes pour trouver une solution. L'axe $x-$ dans la figure 3.11 représente le rapport capacité/demande λ. Comme le montre la figure 3.11, sans RE-router (ILP-EAR), il n'y a pas de solution de routage réalisable et, par conséquent, aucune économie d'énergie n'est réalisée si $\lambda < 38$. Lorsque λ augmente, les liens ont plus de bande passante pour l'ensemble du trafic, les solutions avec et sans routeur RE convergent vers la même quantité d'économies d'énergie. En général, l'heuristique avec les routeurs RE fonctionne bien et se rapproche des résultats d'ILP-GreenRE (l'écart maximum est de $3,8\%$).

Réseau	$\mid V \mid$	$\mid E \mid$	λ_{min}	Volume de trafic (ratio capacité/demande λ)					
				avec routeurs-RE			sans routeurs-RE		
				λ_{min}	$2\lambda_{min}$	$3\lambda_{min}$	λ_{min}	$2\lambda_{min}$	$3\lambda_{min}$
Atlanta	15	22	38	27.7%	34.3%	36.4%	0%	32%	36%
New York	16	49	15	52.2%	62.9%	65.8%	2%	59%	63%
Germany17	17	26	44	30.6%	36.7%	37.3%	0%	35%	39%
France	25	45	67	39.2%	43.4%	46%	0%	42%	44%
Norway	27	51	75	37.7%	45.6%	47.8%	12%	43%	47%
Nobel EU	28	41	131	29.2%	33.1%	34.2%	12%	32%	34%
Cost266	37	57	175	30.6%	35%	36.3%	3.5%	32%	35%
Giul39	39	86	85	42.5%	50.5%	53.3%	0%	45%	50%
Pioro40	40	89	153	50.5%	53.7%	55.2%	0%	53%	54%
Zib54	54	80	294	27.5%	30.8%	32.8%	0%	30%	33%

Table 3.3: Gains énergétiques (en %)

Economie d'énergie pour 10 réseaux classiques Nous présentons dans le tableau 1 les gains énergétiques pour dix topologies de réseaux classiques utilisant H_{ILP}-GreenRE et H-EAR - l'heuristique *sans* routeurs RE trouvée dans [Ci56]. Différent de λ_{minRE} en section 4.2.1, nous

utilisons λ_{min} comme plus petite valeur du rapport capacité/demande qui permet de trouver un itinéraire réalisable pour toutes les demandes (donné dans [Ci56]). Dans les simulations, une plage de $\{\lambda = \lambda_{min}, 2\lambda_{min}, 3\lambda_{min}, 3\lambda_{min}\}$ est utilisée pour représenter la charge de trafic élevée (p. ex. trafic aux heures de pointe), moyenne et faible (p. ex. trafic de nuit) sur les réseaux. Comme le montre le tableau 1, avec les routeurs RE, on commence à économiser une grande quantité d'énergie (en moyenne 37 %) même avec $\lambda = \lambda_{min}$. Rappelez-vous que le routage avec les routeurs RE est possible même avec $\lambda << \lambda_{min}$ alors qu'aucune solution réalisable n'est trouvée sans routeur RE. Quand λ est assez grand, il n'est pas nécessaire d'avoir des routeurs RE sur le réseau, donc les deux solutions (avec et sans RE-routeur) convergent vers presque la même valeur de gains en économies d'énergie.

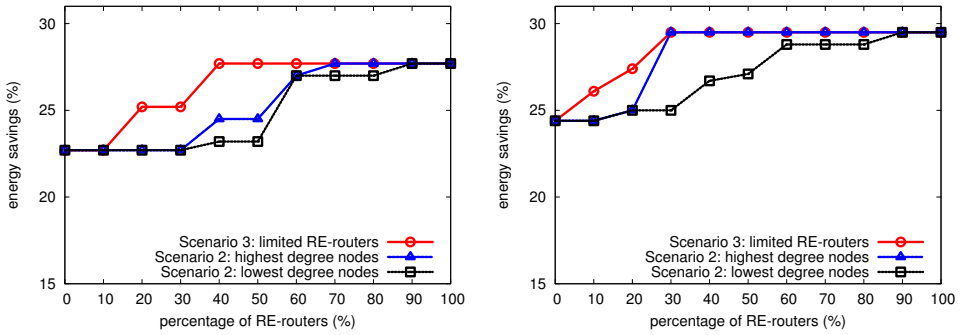


Figure 3.12: Economie d'énergie avec with limited RE-routers vs. a subset of capable RE-routers

Economie d'énergie pour les Scenarios 2 et 3 du problème GreenRE Dans cette section, nous évaluons les économies d'énergie du scénario 2 (un sous-ensemble prédéfini de routeurs compatibles RE) et du scénario 3 (un nombre limité de routeurs compatibles RE). Nous avons fixé la capacité de liens et la demande correspondant à λ_{min} dans la section 4.2.3. L'axe x de la figure 3.12 est le pourcentage de routeurs compatibles RE sur le réseau. Par exemple, avec le scénario 3, nous trouvons la solution de routage qui minimise la consommation d'énergie alors qu'il y a au plus $(x \times |V|)$ routeurs RE sur le réseau. Pour le scénario 2, nous plaçons $(x \times |V|)$ des routeurs compatibles RE sur (1) les nœuds de degré les plus élevés ou (2) les nœuds de degré les plus bas dans le graphique G. Comme le montre la figure 3.12, le scénario 3 surpasse toujours le scénario 2 puisqu'il peut trouver les meilleures positions pour placer des routeurs compatibles RE. Par exemple, dans le réseau d'Atlanta avec un maximum de 6 routeurs RE, l'écart maximum est de 4,5 % et il y a 4 routeurs RE aux nœuds de degré le plus élevé et les deux autres sont aux nœuds de degré moyen et le plus bas. Une autre observation importante que nous avons trouvée dans le scénario 2 est que le fait de placer des routeurs RE sur des nœuds de haut degré donne de meilleurs résultats en termes d'économies d'énergie. C'est parce que placer des routeurs compatibles RE sur des nœuds de haut degré aide à réduire la charge de trafic et donne plus de chances de rediriger le trafic sur quelques liens, permettant à d'autres liens de ces nœuds d'être mis en veille.

3.2.5 Conclusion

À notre connaissance, GreenRE est le premier travail considérant l'élimination de la redondance comme une aide complémentaire pour un problème de routage sensible à l'énergie. Nous for-

mulons le problème sous la forme d'un programme linéaire en nombres entiers et proposons des algorithmes heuristiques gloutons. Les simulations sur plusieurs topologies de réseaux montrent un gain significatif en économies d'énergie avec GreenRE. Pour les travaux futurs, il serait intéressant d'étudier un modèle plus réaliste dans lequel les taux de redondance des données et les volumes de la demande de trafic varient en fonction des traces de trafic réelles.

3.3 Distribution de contenus efficace en énergie

La distribution de contenu, en particulier vidéo, est devenue l'application numéro un des réseaux. Il est estimé que plus de 70% du trafic lui est dû. Dans le même temps, plusieurs propositions ont été faites pour améliorer l'efficacité de cette distribution, en particulier l'utilisation de caches pour stocker les vidéos les plus populaires proches des utilisateurs. Cela correspond à l'utilisation de plusieurs technologies comme les réseaux centrés sur les contenus ou CCN en bref pour Content Centric Networks ou les CDN pour Content Delivery Networks. Dans cette section, j'étudie comment effectuer une distribution de contenus efficaces en énergie en utilisant ces caches. Je me place dans deux contextes différents.

- Le premier est celui des réseaux cœur. Dans ce travail, nous étudions l'impact de la coopération entre l'utilisation de caches en réseau et des CDN sur un routage économe en énergie. Nous formulons ce problème sous la forme d'une distribution de contenu économe en énergie. Nous proposons un programme linéaire en nombres entiers (ILP) et un algorithme heuristique pour le résoudre. L'objectif de ce problème est de trouver un itinéraire réalisable, de sorte que la consommation totale d'énergie du réseau soit minimisée tout en respectant les contraintes imposées par les demandes et les capacité de liens. Nous exposons pour quelle gamme de paramètres (taille des caches, popularité du contenu, intensité de la demande, etc.) il est utile d'utiliser les caches. Les résultats expérimentaux montrent qu'en plaçant un cache sur chaque routeur de backbone pour stocker le contenu le plus populaire, ainsi qu'en choisissant le meilleur serveur de fournisseur de contenu pour chaque demande à un CDN, nous pouvons économiser environ 20 % de l'énergie en moyenne pour tous les réseaux de cœur considérés.

  Ce travail est une collaboration avec J. Araujo, J. Moulierac, Y. Liu, R. Modrzejewski. Il correspond aux publications [Ci49, J15].

- Le second est celui du réseau régional d'un opérateur. Nous étudions le problème de la réduction de la consommation d'énergie dans un réseau d'un fournisseur d'accès Internet (FAI) en concevant l'infrastructure de distribution de contenu gérée par l'opérateur. Nous proposons un algorithme pour décider de manière optimale où mettre en cache le contenu à l'intérieur du réseau du FAI. Nous évaluons notre solution sur la base de deux études de cas pilotées par les retours d'expérience des opérateurs. En particulier, la collaboration avec Orange Labs à Lannion nous a permis d'avoir accès à une vue précise de la topologie des FAI et des matrices de trafic réseau. Les résultats montrent que la conception économe en énergie de l'infrastructure de contenu permet de réaliser des économies substantielles, tant en termes d'énergie qu'en termes de bande passante requise au point de peering de l'opérateur. De plus, nous étudions l'impact des caractéristiques du contenu et des modèles de consommation d'énergie. Enfin, nous en tirons des enseignements pour la conception de réseaux futurs sensibles à l'énergie.

  Ce travail est une collaboration avec E. Bonetto, L. Chiaraviglio, R. Gonzalez, C. Guerrero, E. Le Rouzic, F. Musumeci, Y. Liu, R. Modrzejewski, T. K. Phan, I. Tahiri. Il a été publié dans [Ci45].

3.3.1 Distribution de contenus efficace en énergie avec CCN et CDN

Introduction

Les opérateurs de réseaux coeur étudient le déploiement de solutions de routage économes en énergie. Le principe général est d'agréger le trafic afin de pouvoir désactiver autant de périphériques réseau que possible [278, 252, 254, ch5].

D'autre part, pour réduire la charge du réseau et améliorer la qualité du service, les fournisseurs de contenu et les opérateurs de réseau ont intérêt à désagréger le trafic en répliquant leurs données en plusieurs points du réseau afin de réduire la distance entre les données demandées et leurs utilisateurs. Ces dernières années ont vu, parallèlement à la popularité croissante de la vidéo sur Internet, une augmentation considérable du trafic desservi par les réseaux de diffusion de contenu (CDN). Ces types de réseaux fonctionnent en répliquant le contenu entre leurs serveurs et en le servant aux utilisateurs finaux à partir du serveur le plus proche. Les CDN fournissent aujourd'hui une grande partie du trafic Internet total : l'estimation va de 15 % à 30 % de tout le trafic Web dans le monde entier pour le CDN [258] le plus populaire. Chiaraviglio et al. [292, 271] ont montré comment le choix des serveurs CDN a un impact sur la consommation d'énergie du backbone. Plus précisément, ils visent à désactiver les périphériques réseau en choisissant, pour chaque demande d'un client à un fournisseur de contenu, le meilleur serveur de ce CDN tout en essayant de minimiser la consommation d'énergie.

Ici, nous allons plus loin dans cette idée en considérant également l'utilisation des caches sur chacun des routeurs du backbone, tout en tenant compte du choix des serveurs CDN. Il est important de mentionner qu'il y a eu plusieurs propositions pour développer des systèmes globaux de mise en cache [302]. En particulier, il a été récemment proposé d'utiliser le stockage réseau et le routage orienté contenu pour améliorer l'efficacité de la distribution de contenu par les futures architectures Internet [333, 224, 309, 309, 283]. Parmi ces études, nous mentionnons que dans ce document, nous ne supposons aucune technologie spécifique pour les futures architectures Internet, ni rien d'autre qui nécessiterait une refonte majeure du fonctionnement d'Internet, comme le routage par contenu dans nos caches. Nous supposons qu'un cache dessert une seule ville, en prenant tout son contenu du fournisseur d'origine. Nous considérons que les caches peuvent être activés ou désactivés. Il y a donc un compromis entre les économies d'énergie qu'ils permettent en réduisant la charge du réseau et leur propre consommation d'énergie.

Nous proposons une formulation ILP pour réduire la consommation d'énergie en utilisant des caches et en choisissant correctement les serveurs des fournisseurs de contenus pour chaque demande. Nous avons implémenté cette formulation sur le solveur ILP CPLEX [98] version 12 et fait des expériences sur des topologies de réseaux réels que nous avons obtenues à partir de SNDlib [280]. Nous avons également testé sur des instances aléatoires générées à partir de graphes Erdős-Rős-Rényi [413]. Nous étudions l'impact de différents paramètres : taille des caches, intensité de la demande, taille du réseau, etc. En particulier, nous avons constaté que des gains énergétiques efficaces peuvent être réalisés, dans nos scénarios, par des caches de l'ordre de 1 TB et que les caches de plus grande taille ne conduisent pas à des gains significativement meilleurs.

Les résultats expérimentaux montrent un potentiel d'économie d'énergie d'environ 20 % en mettant en veille les appareils pendant les périodes de faible trafic. Si le CDN est pris en compte mais sans caches, il y a 16 % d'économies, et au contraire, lorsque les caches sont introduits dans le réseau sans CDN, il y a aussi environ 16 % d'économies. De plus, nous avons observé que l'impact des caches est plus important dans les grands réseaux. Pour pouvoir quantifier cet effet, nous proposons une heuristique efficace. Cette heuristique, appelée Spanning Tree Heuristic,

nous permet d'obtenir des solutions réalisables beaucoup plus rapidement que la résolution du modèle ILP que nous proposons en utilisant CPLEX. Un autre avantage de l'heuristique est qu'elle accepte un paramètre qui contrôle un compromis vitesse/qualité.

La principale conséquence de notre travail est qu'en stockant le contenu le plus populaire dans les caches de chaque routeur et en choisissant le meilleur serveur de fournisseur de contenu, nous pouvons économiser environ 20 % d'énergie dans les réseaux coeur. De plus, l'utilisation de caches nous permet de trouver des solutions réalisables là où l'algorithme sans cache échoue car il faudrait plus de capacité de bande passante sur les liens pour satisfaire les mêmes demandes.

Etat de l'art

Il existe plusieurs études sur la littérature proposant différentes stratégies pour réduire la consommation d'énergie. Par exemple, un modèle qui propose de fermer des liens individuels est étudié dans [ch5]. Une façon intéressante de réaliser des économies d'énergie de manière distribuée est montrée dans [254]. Les CDN éco-énergétiques ont également fait l'objet d'études récentes. Les auteurs dans [241] proposent de réduire la consommation d'énergie dans les réseaux CDN en désactivant les serveurs CDN tout en considérant les contraintes de niveaux de services ou SLAs en bref pour *Service Level Agreements* des utilisateurs (en particulier les contraintes de délai). Afin d'optimiser la consommation d'énergie des serveurs de contenu dans les plates-formes de distribution de contenu à grande échelle sur plusieurs domaines ISP, la stratégie proposée dans [247] consiste à mettre les serveurs en mode veille sans impact sur la capacité du service du contenu. Notre travail est différent de tous ces travaux mentionnés précédemment, puisqu'ils ne prennent pas en compte les caches en réseau.

Les caches réseau ont été utilisés dans les systèmes de caches globaux [302]. Ces dernières années, plusieurs architectures de réseaux centrés sur l'information, telles que Cache and Forward Network (CNF) [333], Content Centric Networking (CCN) [309], CacheShield [232] et NetInf [224], ont exploitées les caches réseau. Leurs objectifs sont d'explorer de nouvelles architectures de réseau et de nouveaux protocoles pour soutenir les futurs services axés sur le contenu. Les schémas de mise en cache ont été étudiés dans ces nouvelles architectures Internet [333, 243, 238, 202, 233]. Un survey récent [206] sur la mise en cache dans les réseaux centrés sur l'information présente des idées pour réduire la redondance du cache et améliorer la disponibilité du contenu mis en cache. Comme dans notre travail, ces travaux utilisent aussi des caches en réseau, mais ils ne tiennent pas compte des économies d'énergie.

L'efficacité énergétique dans les architectures orientées contenu avec une mise en cache en réseau a été récemment étudiée [267, 259, 251]. Dans [267], les auteurs analysent les avantages énergétiques de l'utilisation du CCN par rapport aux réseaux CDN. Un autre travail a porté sur l'impact des différentes technologies de mémoire sur la consommation d'énergie [251].

Deux travaux proposent également l'ajout de caches réseau aux routeurs du backbone qui fonctionnent de manière transparente avec l'architecture Internet actuelle et ils proposent un placement optimal pendant les heures de pointe pour de telles caches dans le réseau d'accès [214, 266]. Ces travaux se concentrent sur l'efficacité énergétique en tenant compte de la diffusion et du stockage des données, mais ils ne tiennent pas compte des économies d'énergie réalisées en activant ou désactivant les liaisons réseau. Les auteurs dans [259] étendent GreenTE [278] pour réaliser une ingénierie du trafic efficace en énergie dans un réseau CCN.

Le travail de Chiaraviglio et al. [292, 271] est, à notre connaissance, le plus proche du nôtre. Ils proposent de permettre la coopération entre les opérateurs de réseaux et les fournisseurs de contenus afin d'optimiser la consommation totale d'énergie en utilisant une formulation ILP pour les deux parties. Dans cet article, nous considérons une extension de cette formulation de

problème d'optimisation, en considérant la mise en cache dans le réseau.

Discussion

Coopération entre fournisseur de contenus et opérateur réseau. Rappelons que pour mettre en pratique les solutions proposées ici, il est nécessaire d'établir une collaboration entre les fournisseurs de contenu et les opérateurs de réseaux. En effet, il est nécessaire de connaître, d'une part, les emplacements, les capacités et le contenu des serveurs CDN et, d'autre part, les topologies de réseau et les listes d'équipements réseau qui peuvent être désactivés. De nos jours, ce type d'information est privé et est considéré comme stratégique par la plupart des entreprises.

Un contexte naturel pour la mise en œuvre de notre solution est celui d'un opérateur de réseau qui est aussi un fournisseur de contenu. Il s'agit d'une tendance à la hausse, car les revenus passent de plus en plus de la distribution de trafic à la distribution de contenus. A titre d'exemple, Orange (ex-France Telecom) exploite son propre CDN. Elle distribue notamment le trafic du site de partage de vidéos Dailymotion, acheté en 2011. Plusieurs études explorent les avantages possibles de cette nouvelle entente [308, 263], et notre article peut être considéré comme l'un d'entre eux. Néanmoins, nous pensons que, comme les études montrent l'intérêt de ce type de solutions, il est possible d'envisager au moins un partage partiel de l'information entre deux opérateurs de réseaux indépendants et un fournisseur de contenu.

Implémentation pratique. Les protocoles de routage sensibles à l'énergie sont une solution prometteuse, mais il existe encore des problèmes pratiques à résoudre avant la mise en œuvre. Les opérateurs réseau n'aiment pas éteindre les équipements et cela change leurs configurations de routage. En effet, les protocoles de routage mettent un certain temps à converger, ce qui peut entraîner une instabilité du réseau, des pertes de paquets et, par conséquent, un délai accru pour les utilisateurs finaux. Néanmoins, nous croyons que les protocoles de routage sensibles à l'énergie peuvent être mis en œuvre dans un contexte simple comme celui de notre étude. En fait, nous discutons ici d'un petit nombre de changements de routes pré-planifiés au cours de la journée et non de changements rapides. Le trafic agrégé quotidien peut être bien estimé et un ensemble de configurations en fonction de l'heure de la journée peut être pré-calculé. De plus, ce nombre de configurations est relativement faible : nous avons vu dans notre étude, qu'avec 5 à 12 configurations de ce type par jour, nous pouvons déjà obtenir des économies significatives. Ce fait est également observé par d'autres études [225].

Deuxièmement, pour réduire le temps de convergence, un contrôle centralisé peut être implémenté avec une technologie comme les réseaux définis par logiciel (SDN). Cette technologie est très prometteuse pour mettre en pratique des solutions énergétiques. En effet, cela permet d'effectuer des mesures de trafic, d'effectuer des calculs d'itinéraires, puis de déclencher une installation de nouvelles règles de routage dans les routeurs et mettre en veille des équipements. En effet, le contrôleur centralisé est capable d'activer/désactiver des interfaces réseau ou des caches via des messages de contrôle SDN. Notez que ces messages seront très petits par rapport au trafic global et peu fréquents (seuls quelques changements suffisent pour obtenir la plus grande partie du gain d'énergie, par exemple toutes les 4 heures). L'augmentation de la consommation d'énergie sera donc négligeable. En résumé, le contrôleur centralisé de SDN peut collecter des matrices de trafic et ensuite calculer une solution de routage satisfaisant la QoS tout en étant minimal en consommation d'énergie. Ensuite, le contrôleur mettra à jour les tables de routage des nœuds du réseau considérés et désactivera certaines interfaces réseau et certains caches si nécessaire afin d'économiser de l'énergie. Nous étudions une solution similaire dans le chapitre 8.

Applications très sensibles au délai. Dans le cas d'une application pour laquelle le délai est

critique, il pourrait être intéressant de placer d'abord cette application, de sorte qu'elle ne subisse qu' au pire un léger retard, puis d'appliquer les algorithmes (au mieux) au trafic restant. Ceci peut être facilement ajouté dans notre modèle en ajoutant un critère de priorité avec un champ de qualité de service pour chaque demande par exemple. De plus, notez que l'algorithme utilisera la plupart du temps le serveur/données le plus proche. En fait, ce choix est le meilleur en termes d'utilisation de la bande passante et, par conséquent, sera souvent le choix préféré de l'algorithme d'efficacité énergétique. Notez également que pour ce type d'applications, l'utilisation de caches n'est pas adaptée, car les données ne seront généralement pas redondantes entre les utilisateurs (appels téléphoniques ou vidéo, jeux vidéos). De plus, cela représente un petit pourcentage du trafic global, donc les résultats globaux de nos expériences ne seront pas modifiés de manière significative.

Conclusion et perspectives

Nous avons abordé ici un problème d'économie d'énergie dans les réseaux cœur. C'est le premier travail à considérer l'impact sur un routage économe en énergie de la présence de caches dans les routeurs, ainsi que de l'affectation des serveurs des CDN aux demandes.

Nous avons étudié des exemples basés sur des topologies de réseau réelles tirées de SNDLib. Les économies d'énergie totales que nous avons trouvées oscillent autour de 20 % pour des paramètres réalistes. La partie de l'énergie économisée uniquement grâce à l'introduction des caches peut aller jusqu'à 16 % pour nos instances.

Comme travail futur, on pourrait examiner différentes architectures de réseau. Ce travail ne considérait que le réseau cœur. Une prochaine étape pourrait être l'introduction de réseaux d'accès, conduisant à des instances plus larges. Comme les économies dues aux caches augmentent avec la taille du réseau, elles devraient être considérablement plus élevées dans ce cas. Cela pourrait également motiver l'étude de nouveaux mécanismes, par exemple la mise en cache par couches. Nous étudions ce cas ci-dessous.

3.3.2 Distribution de contenus efficace en énergie dans un réseau FAI

Introduction

Les centres de données et les réseaux cœur connaîtront les taux de croissance de consommation d'énergie les plus élevés dans les années à venir [265], en raison de l'augmentation du trafic, en particulier pour le contenu multimédia. Par exemple, Afin d'atténuer cette tendance, différentes solutions ont été proposées dans la littérature pour la conception et la gestion des réseaux cœur à haut rendement énergétique (voir [273] pour une vue d'ensemble).

Récemment, le problème de la réduction de la consommation d'énergie dans un réseau cœur en déplaçant les contenus accessibles par les utilisateurs a attiré l'attention de la communauté de chercheurs. En particulier, dans [293], le problème de la réduction de la consommation d'énergie d'un Fournisseur d'Accès Internet ou FAI en bref et d'un fournisseur de contenu (CP) conjointement a été étudié, montrant que des économies d'énergie considérables peuvent être obtenues lorsque le CP et l'ISP coopèrent pour minimiser la consommation d'énergie totale. Dans [283, 250] les auteurs proposent une architecture basée sur le Content Centric Networking (CCN) pour réduire la consommation d'énergie. De plus, dans [300], une architecture basée sur des gateways domestiques formant une infrastructure de centre de données distribuées gérée par le FAI est proposée et évaluée. Enfin, les compromis énergétiques d'une architecture basée sur des services vidéo immersifs sont évalués dans [242]. Tous ces travaux prouvent qu'une

énorme quantité d'énergie est économisée lorsque le FAI prend le contrôle du contenu et le met en cache en tenant compte de l'énergie consommée pour déplacer l'information sur le réseau. Les questions algorithmiques générales sur la mise en cache et le placement des données dans les réseaux ont été étudiées dans [297, 347].

Dans ce travail, nous étudions le problème de la réduction de la consommation d'énergie dans un réseau de FAI en considérant la conception d'une infrastructure de distribution de contenus gérée par le FAI. Notre objectif est d'étudier où mettre en cache le contenu à l'intérieur du réseau afin de réduire la consommation d'énergie globale du système composé des éléments réseau et de stockage installés. Dans les réseaux actuels des fournisseurs d'accès Internet, une énorme quantité de trafic est échangée entre les utilisateurs et les centres de données appartenant à de grands fournisseurs de contenus ou CPs en bref pour Content Providers, tels que Google, Yahoo, Amazon et Limelight. Normalement, les centres de données des grands CPs sont situés à proximité des points de peering de l'ISP [341]. Par conséquent, le trafic provenant des centres de données doit traverser un certain nombre de sauts dans le réseau du FAI avant d'atteindre les utilisateurs. Nous étudions donc la mise en cache optimale du contenu à l'intérieur du FAI, plutôt que d'envoyer le contenu des centres de données aux utilisateurs. Dans notre scénario, nous considérons une topologie logique hiérarchique composée de différents niveaux (par exemple, coeur, métro et accès), et nous optimisons la consommation d'énergie en choisissant le meilleur niveau où placer chaque contenu.

Les avantages de la conception éco-énergétique des architectures de distribution de contenus à l'intérieur du FAI sont multiples. Tout d'abord, il est possible de réduire conjointement les coûts d'électricité du stockage et du réseau, puisque leur énergie est explicitement prise en compte lors de la phase de conception. Deuxièmement, le FAI réduit le volume de trafic échangé sur le réseau. Cela peut à son tour diminuer les coûts de maintenance encourus par le FAI, puisque les éléments de réseau sont mis à jour moins fréquemment et qu'il y a moins de nouveaux dispositifs de commutation à installer. Troisièmement, les coûts monétaires pour l'envoi et la réception d'informations provenant de l'extérieur du réseau sont également réduits, puisque moins de bande passante est nécessaire.

Les articles les plus proches de notre travail sont [301, 266, 262]. Dans [301] les auteurs détaillent un modèle analytique pour la mise en cache en tenant compte du coût de transport de l'information et du coût de stockage du contenu. Cependant, le modèle est dérivé d'un scénario simple (un réseau métropolitain), avec au maximum trois niveaux dans la topologie comme emplacements possibles pour la mise en cache du contenu. De plus, l'évaluation des économies d'énergie n'est pas effectuée. Dans [266] les auteurs proposent un modèle de mise en cache qui intègre les coûts énergétiques. L'évaluation est effectuée en tenant compte de cinq niveaux possibles pour la mise en cache. Enfin, dans [262], un modèle ILP et deux heuristiques simples pour la distribution de contenu économe en énergie sont détaillés. Cependant, les auteurs ne tiennent pas compte de l'énergie consommée pour l'envoi du contenu vers les emplacements possibles à l'intérieur du réseau et un nombre limité de niveaux est également supposé.

Contrairement aux travaux précédents, nous allons plus loin dans cet article : a) en définissant un modèle avec un nombre générique de niveaux et pas seulement limité à des valeurs spécifiques ou à des segments spécifiques du réseau, b) en proposant un algorithme optimal pour décider où mettre en cache le contenu et calculer la consommation totale d'énergie, c) en évaluant les résultats sur deux études de cas. De plus, nous considérons l'impact des propriétés topologiques sur la mise en cache du contenu, et nous en déduisons quelques idées pour la conception de futurs réseaux sensibles à l'énergie.

Je présente d'abord le problème dans la section 3.3.2, puis mes résultats dans la section 3.3.2.

Description du problème

Nous supposons que le réseau est organisé selon une structure hiérarchique composée de différents niveaux. En particulier, nous supposons un réseau arborescent pour représenter la collection de chemins entre chaque utilisateur et le nœud de peering Internet. Les nœuds sont regroupés en fonction d'une hiérarchie et chaque niveau de l'arborescence correspond à un niveau différent de la hiérarchie. Les données de contenu sont transmises aux clients en suivant un chemin sur l'arbre à partir de la racine, c'est-à-dire du point de peering Internet. Un cache de stockage peut être situé à chaque nœud du réseau, ce qui offre une possibilité de stockage des données. De plus, les caches sont organisés selon une structure hiérarchique : si un contenu demandé n'est pas disponible dans un cache donné, la demande est transmise au cache parent de la hiérarchie, sans aucune collaboration entre les caches situés au même niveau de l'arborescence. Enfin, nous n'imposons pas une taille de cache donnée, c'est-à-dire que la taille du cache est une sortie de notre approche.[1]

La procédure de distribution du contenu est divisée en plusieurs étapes : a) le contenu est récupéré depuis le point de peering jusqu'aux caches de stockage situées à un niveau donné de l'arbre, b) le contenu est mis en cache pour une durée déterminée, c) pendant cette période, le contenu est récupéré par les utilisateurs, en fonction de sa popularité. Nous associons ensuite un coût énergétique à chacune de ces étapes et nous calculons la consommation totale d'énergie. Notre objectif est alors de trouver la quantité optimale de données à mettre en cache à chaque niveau de l'arbre afin de minimiser la consommation globale d'énergie.

En nous concentrant sur les besoins en énergie, nous considérons le coût de conservation du contenu stocké dans le cache, le coût de lecture/écriture du contenu de/vers le cache et le coût d'envoi du contenu à travers un hop de l'arbre. Nous supposons que le coût de la traversée d'un hop est différent pour chaque niveau, en raison des différents dispositifs de commutation déployés dans chaque segment du réseau [264]. Afin de modéliser la consommation d'énergie de chaque appareil, nous supposons une dépendance linéaire avec le volume de trafic, suivant les hypothèses des travaux précédents [326, 289, 262]. En particulier, le coût de transport de l'information est exprimé en termes d'énergie par bit, c'est-à-dire la consommation totale d'énergie divisée par le débit moyen.

Plus formellement, l'ensemble des niveaux dans le réseau est $\mathcal{L} = \{1, \ldots, L\}$, $L = |\mathcal{L}|$ étant le nombre de niveaux. Le point de peering est situé au niveau 1, tandis que les utilisateurs sont connectés au niveau L (par exemple, les DSLAMs). Nous dénotons le nombre total de dispositifs de commutation situés au niveau $j \in \mathcal{L}$ comme N_D^j. Définissons le coût de stockage pour un seul cache comme C_S. C_R est le coût de lecture/écriture de contenu sur un cache. C_H^j est le coût de traversée d'un nœud situé au niveau j dans le réseau. De plus, nous considérons les caractéristiques suivante du contenu. Nous supposons que le contenu est représenté par des vidéos regardées par les utilisateurs. τ est le débit total des vidéos demandées par les utilisateurs. Dénotons la taille moyenne de la vidéo comme A et la durée de la fenêtre de popularité comme I. Ainsi, le nombre total de vidéos V_W regardé pendant la fenêtre de popularité est :

$$V_W = \frac{\tau I}{A} \tag{3.1}$$

Définissons V_S comme le nombre total de vidéos fournies par le CP. Nous divisons les vidéos en classes en fonction de leur popularité, N_C étant le nombre de classes. L'ensemble des classes est désigné par $\mathcal{K} = \{1, ..., N_C\}$. La classe 1 est la plus populaire tandis que la classe N_C est la

[1] Dans nos scénarios, la taille de cache obtenue est toujours inférieure à la capacité maximale des périphériques de stockage actuels.

moins populaire. Nous supposons qu'en moyenne, chaque classe a le même nombre de vidéos, ce qui correspond à $V_C = \frac{V_S}{N_C}$.

Pour chaque classe $k \in \mathcal{K}$ nous adoptons le modèle de popularité de Zipf de [346] et calculons le nombre de vidéos regardées par classe comme suit

$$V_W^k = V_W \frac{k^{-\beta}}{\sum_{k=1}^{N_C} k^{-\beta}} \tag{3.2}$$

β étant le paramètre de la distribution de Zipf.

Nous calculons ensuite l'énergie consommée pour disséminer la classe k lorsqu'elle est stockée sur les caches situées au niveau j. En particulier, nous calculons d'abord l'énergie consommée pour récupérer le contenu dans les caches, et pour conserver le contenu stocké :

$$\phi(j) = AV_C N_D^j \left(\sum_{z=1}^{j-1} C_H^z + C_R + C_S I \right). \tag{3.3}$$

Le premier terme entre parenthèses est le coût de la traversée de $(j-1)$ hops. Le deuxième terme est le coût de l'écriture du contenu sur le cache. Le troisième terme est le coût de conservation du contenu stocké, qui est multiplié par la durée de la fenêtre de popularité I puisque ce coût doit toujours être comptabilisé pour toute la période de temps. Tous les coûts sont ensuite multipliés par la quantité d'informations stockées dans le niveau j, c'est-à-dire $A \times V_C \times N_D^j$. Notez que $\phi(j)$ ne dépend pas directement de la popularité de la classe mais seulement du niveau j choisi pour la mise en cache.

Nous calculons ensuite l'énergie consommée pour récupérer le contenu :

$$\varphi(j,k) = AV_W^k \left(C_R + \sum_{z=j}^{L} C_H^z \right). \tag{3.4}$$

En particulier, nous considérons le coût de lecture du contenu et le coût d'envoi du contenu à partir des caches au niveau j aux utilisateurs. les informations récupérées correspondent aux vidéos qui sont regardées pendant la durée de la fenêtre de popularité, c'est-à-dire, $A \times V_W^k$. Différemment de $\phi(j)$, $\varphi(j,k)$ dépend à la fois de la popularité de la classe et du niveau où le contenu est mis en cache.

L'énergie totale pour disséminer la classe k au level j est :

$$E_k^j = \begin{cases} \phi(j) + \varphi(j,k), & j > 0 \\ AV_W^k \sum_{z=1}^{L} C_H^z, & j = 0 \end{cases}. \tag{3.5}$$

Notez que le niveau 0 est le cas particulier où les données sont servies à partir de la source originale, c'est-à-dire que la mise en cache n'est pas exploitée dans le réseau considéré. Dans ce cas, la consommation totale d'énergie est le coût de l'envoi des vidéos regardées directement du point de peering aux utilisateurs.

Le meilleur niveau pour stocker la classe k est donc :

$$h_k = \underset{j \in \mathcal{L}}{\operatorname{argmin}} E_k^j. \tag{3.6}$$

Notez que le meilleur niveau pour chaque classe est calculé indépendamment des autres classes. Nous répétons donc cette procédure pour chaque classe k.

Table 3.4: Résumé des résultats pour les deux réseaux.

Métrique		FT	Marocain
Gains énergétiques (S)		8.7%	11.0%
Gains monétaires annuels [k€]		769	122
Gains en bande passante (Ψ)		18.2%	30.2%
Taille du cache [GB]	Λ^1	0	0
	Λ^2	0	0
	Λ^3	32546	0
	Λ^4	0	23510
	Λ^5	35878	5581
	Λ^6	2041	46

La consommation totale d'énergie avec mise en cache est calculée comme suit :

$$T = \sum_{k \in \mathcal{K}} E_k^{h_k} \tag{3.7}$$

que l'on peut comparer à la consommation d'énergie sans mise en cache :

$$T' = \sum_{k \in \mathcal{K}} E_k^0. \tag{3.8}$$

En comparant T avec T', nous pouvons estimer si la mise en cache est efficace ou non pour économiser de l'énergie. Cependant, calculer Eq.(3.6) pour chaque classe n'est pas faisable, puisque l'itération sur les niveaux doit être répétée pour toutes les classes, résultant en une complexité temporelle de $\mathcal{O}(L \times N_C)$. Pour résoudre ce problème, nous avons proposé un nouvel algorithme afin de calculer efficacement T, voir [Ci45].

Résultats

Nous avons ensuite évalué GCT sur deux réseaux réalistes de FAI nationaux, à savoir Orange (FT) et un FAI au Maroc. Les deux réseaux sont composés de six niveaux au total (ncore, core-regional, metro-core, metro, access-metro, access), et avec un nombre différents de commutateurs déployés à chaque niveau.

Le tableau 3.4 rapporte les résultats pour les deux scénarios obtenus avec l'algorithme GCT. Nous considérons d'abord les économies d'énergie par rapport au cas où la mise en cache n'est pas exploitée. Des économies d'énergie de près de 9% et 11% sont possibles pour les scénarios FT et marocain, respectivement. En supposant que les caches sont rafraîchis une fois par semaine pendant une année entière, nous avons estimé une économie monétaire[2] de plus de 700 k€ pour FT, et de plus de 100 k€ pour le réseau marocain. De plus, les économies de bande passante réalisées au point de peering sont encore plus importantes, atteignant 18 % pour le scénario FT et 30 % pour le scénario marocain.

La table indique également la taille du cache Λ^j par périphérique pour chaque niveau j. Il est intéressant de noter que Λ^j est au maximum 36 To, une valeur qui peut être couverte par un array commercial de disques durs. De plus, les besoins en capacité tendent à diminuer en se rapprochant des utilisateurs, avec au maximum 2041 Go de stockage requis au niveau de l'accès pour le réseau FT et seulement 46 Go pour le réseau marocain.

[2]Nous avons supposé un coût de l'électricité de 0,21€/kWh.

Conclusions

Nous avons étudié la conception économe en énergie d'une architecture de contenus dans un réseau de FAI, en exploitant les caches gérés par le FAI. Nous avons pris en compte le coût du routage du contenu à l'intérieur du réseau, le coût de lecture/écriture du contenu depuis/vers les caches, et le coût du stockage. Après avoir défini un modèle pour la distribution de contenu, nous avons proposé un algorithme efficace, appelé GCT, pour décider de l'emplacement des caches dans le réseau du FAI et calculer la consommation d'énergie totale. Nous avons appliqué notre algorithme sur les scénarios FT et marocain. Nos résultats indiquent que la mise en cache apporte des économies substantielles en termes d'énergie et de bande passante. De plus, nous avons montré que les résultats sont influencés par la caractéristique du contenu, en particulier la popularité du contenu. De plus, nous avons montré que l'application de notre approche aux futurs réseaux augmentera les économies.

Comme prochaines étapes, nous envisagerons la gestion conjointe de l'architecture de distribution de contenu. En particulier, notre objectif est d'étudier la variation du trafic dans le temps et de calculer le meilleur ensemble de caches activés pour satisfaire une demande de trafic donnée, tout en laissant les autres caches désactivés. Une autre direction possible est d'introduire une coopération entre les caches voisins pour servir les utilisateurs et réduire la quantité d'informations stockées. Enfin, nous prévoyons d'étudier l'impact de la prise en compte de plus d'un nœud de peering et l'impact de l'introduction de matrices de trafic réalistes à l'intérieur du FAI.

Chapter 4

Question fondamentale de théorie des graphes posée par le routage efficace en énergie

L'étude de la conception et de la gestion de réseaux efficaces en énergie nous a amené à utiliser des outils de théorie des graphes. Nous montrons ici deux exemples de cette utilisation qui correspondent aux publications [Ci40, Ci39, J13, R84]. Ces travaux sont le fruit d'une collaboration avec S. Pérennes et I. Tahiri.

Au cœur de la problématique de réduction de la consommation énergétique d'un réseau ISP se trouve donc le problème suivant : trouver le sous-graphe minimum en nombre d'équipements qui peut supporter tout le trafic à transporter. Le trafic est agrégé sur un nombre minimal d'arêtes dans un réseau efficace en énergie.

Nous avons étudié la complexité du probléme général. Nous avons montré dans le chapitre précédent que le problème est NP-complet et non-approximable [Ci56, ch5]. Il est à noter que même le sous-problème beaucoup plus simple de trouver l'arbre couvrant maximisant la charge est aussi NP-complet [261].

Dans le présent chapitre, je me suis intéressé à un cas particulier où la demande est uniforme et all-to-all et les capacités sur les liens sont uniformes. Dans ce cas, trouver, pour une charge réseau donnée, le sous-graphe permettant de router toutes les demandes et qui a un nombre minimum d'arêtes est équivalent à un problème dual qui est, étant donné un nombre d'arêtes, trouver le sous-graphe permettant de faire passer la plus grande charge.

L'indice de transmission d'un graphe est le minimum, sur tous les routages possibles de toutes les demandes, de la charge maximale d'une arête. Cette métrique est d'un grand intérêt puisqu'elle capture la notion de congestion globale de manière précise : moins l'indice de transmission est élevé, moins la congestion est importante. Je suis donc parti à la recherche de (sous-)graphes couvrants avec un nombre fixé d'arêtes qui ont l'indice de transmission (forwarding index) minimum. J'ai effectué pour cette quête deux études qui correspondent aux deux sections de ce chapitre.

Dans la première, section 4.2, nous avons étudié le problème pour des graphes *planaires* avec degré borné [Ci40, J13]. Nous avons pris l'exemple de la grille carré $n \times n$, nommé G_n, comme c'est un graphe simple qui se retrouve dans beaucoup de modèles réseaux. Notre étude avait trois objectifs principaux.

- Le premier était de déterminer combien de liens pouvaient être enlevés (ou éteints) d'une

grille carrée sans dégrader son indice de transmission. Nous avons obtenu une réponse précise à cette question. Ainsi, nous montrons que l'on doit garder au moins $\frac{4}{3}n^2$ arêtes et nous fournissons une construction utilisant $(\frac{4}{3}+\frac{1}{9})n^2$ arêtes.

- Le deuxième était de comprendre comment l'indice de transmission d'un réseau similaire à une grille avec N sommets évolue quand le nombre d'arêtes augmente de $N-1$ à $2N$. Nous prouvons que si le nombre d'arêtes est $N-1+e$ le forwarding index est $\Theta(\frac{N^2}{\sqrt{e}})$. Ce phénomène est complètement différent de l'évolution dans un graphe général pour lequel le forwarding index decroit de façon linéaire (i.e. le forwarding index se comporte comme $\Theta(\frac{N^2}{e})$). Ce comportement particulier est en effet relié à la planarité de la grille et est en fait dû principalement à l'inégalité isopérimétrique de Tarjan-Lipton [407]. Ainsi, un comportement similaire aurait lieu pour n'importe quel graphe planaire.

- Le troisième était d'utiliser certaines techniques comme le clustering et les inégalités isopérimétriques et d'exhiber comment elles pouvaient être utilisées dans ce contexte. L'utilisation de ces techniques est continuée dans la section suivante dans laquelle nous étudions le problème pour des classes de graphes plus générales, mais de façon moins précise (ordre de grandeur).

Dans la seconde section, section 4.3, nous étudions la question de conception suivante : étant donné un nombre e d'arêtes et un nombre n de sommets, quels sont les graphes avec le plus petit indice de transmission que nous pouvons construire [Ci39, R84, S28] ?

- Nous répondons ici à cette question pour différentes familles de graphes, *graphes généraux, graphes avec degré borné, graphes peu denses avec un petit nombre d'arêtes*, en fournissant des constructions, la plupart d'entre elles asymptotiquement optimales. Par exemple, nous fournissons une construction asymptotiquement optimale pour les graphes cubiques-$(n, e = n+k)$, leurs indices de transmission est $\sim \frac{n^2}{3k}\log_2(k)$. Nos résultats permettent de comprendre comment l'indice de transmission chute lorsque des arêtes sont ajoutées à un graphe et aussi de déterminer quelle est la *meilleure structure avec e arêtes.*

- Nous avons introduit et utilisé la notion de *squelette* de graphes pour trouver des graphes optimaux pour une large gamme de valeurs de n et m. Certains des résultats sont surprenants : en effet, contrairement à ce que nous attendions, certaines structures optimales ne sont pas symétriques.

4.1 Préliminaires

Définition. Étant donné un graphe $G = (V, E)$ avec $n = |V|$ sommets, un routage de R est une collection de chemins reliant toutes les paires ordonnées de sommets de G. Un routage R induit sur chaque arête e une *charge* qui est le nombre de chemins passant par e. L'*indice de transmission arête* (ou simplement l'indice de transmission) $\pi(G, R)$ de G par rapport à R est alors le nombre maximum de chemins de R passant par n'importe quelle arête de G. En d'autres termes, il correspond à la charge maximale d'une arête de G lorsque R est utilisé. Par conséquent, il est important de trouver des routages minimisant cet index. L'indice de transmission $\pi(G)$ de G est le minimum $\pi(G, R)$ sur tous les routages R de G.

Etat de l'art. L'indice de transmission a été introduit par Chung et al. en 1987 [403], en raison de son importance, ce paramètre a été étudié de manière assez approfondie : d'un côté,

des résultats ont été donnés pour différentes classes de graphes (par ex. graphes aléatoires [395], transitifs et de Cayley [401, 388, 375], graphes avec de petits nombres de sommets [391] et graphes bien connectés [396]). D'autre part, des relations profondes avec d'autres invariants de graphes liés à l'expansion ont été établies : Laplacian, constante de Cheeger (voir le survey [385]), Sparsest cut [381] et la "géométrie des graphes" [389]. Cette notion a également été utilisée pour prouver que certaines chaînes de Markov se mélangent rapidement en utilisant soit des chemins canoniques (routages), soit de la "résistance", [393]. Voir le survey récent [231] pour une vue d'ensemble des résultats connus. Quelques variantes comme la charge sur les arcs pour les digraphes ([387]) et la charge sur les sommets ont également été étudiées.

Indice de transmission et débit réseau. Le problème est également connu sous le nom de *maximum concurrent flow problem* et son problème dual a probablement été introduit pour la première fois dans [398] dans lequel les auteurs ont également discuté de la relation avec le débit du réseau. Dans [392], les auteurs donnent un algorithme de routage de paquets simple permettant d'obtenir la stabilité du réseau pour tout débit λ avec $\lambda\pi < 1$. Plus précisément, quand on suppose une demande de trafic uniforme entre toutes les paires de sommets ordonnés à un taux donné λ (i.e. u envoie un paquet à v avec probabilité λdt), on peut prouver que, quelle que soit la politique de routage, la charge maximale sur toutes les arêtes sera d'au moins $\lambda\pi(G = V, E)$. De plus, il existe des politiques de routage (best effort [392]) qui garantissent que, si les arêtes ont un débit d'au moins $\lambda\pi(G)$, le réseau sera stable et le trafic routé. Si nous adoptons un point de vue plus grossier et regardons les demandes comme des chemins, une image du réseau à un moment donné est un ensemble de chemins choisis au hasard et choisis indépendamment avec probabilité λ et ainsi chaque arc reçoit une charge fortement concentrée autour de $\lambda\pi(G)$. Ceci rend *l'indice de transmission arête d'une grande importance pratique, puisque le débit maximum, qu'un réseau peut tolérer, est* $\frac{1}{\pi(G)}$.

4.2 Spanners de la grillle avec petits indices de transmission pour des réseaux efficaces en énergie

4.2.1 Introduction

Motivés par l'efficacité énergétique, nous recherchons, pour différents nombres d'arêtes, les meilleurs graphes couvrants (spanning graphs) d'une grille carrée, c'est-à-dire ceux qui ont un faible indice de transmission.

Nous appelons un sous-graphe connecté couvrant d'un graphe G, un *spanner* de G. Plus précisément, il s'agit d'un sous-graphe connecté qui a le même ensemble de sommets que G. Notre but est de trouver, pour une borne donnée sur le nombre d'arêtes, le meilleur spanner de G, c'est-à-dire celui avec l'indice de transmission minimum. Le problème peut aussi être vu comme suit : pour une borne U donnée sur l'indice de transmission, trouver un spanner F de G avec un nombre minimum d'arêtes tel que $\pi(F) \leq U$.

Savoir comment résoudre ce problème est très intéressant en pratique pour les opérateurs de réseaux désireux de réduire la consommation d'énergie de leurs réseaux. En fait, la plupart des liens réseau consomment une énergie constante indépendamment de la quantité de trafic qu'elles génèrent. [343], [303]. Il a donc été proposé de réduire la consommation d'énergie des liens réseau en éteignant certains d'entre eux ou, plus commodément, en les mettant en mode veille en dehors des heures de pointe. Plusieurs études ont été réalisées. [321], [332], [Ci50], [Ci43], montrent qu'un bon choix des liens à désactiver peut conduire à des économies d'énergie significatives, tout en conservant la même qualité de communication. Dans le cas où les flots de

chaque nœud vers chaque autre nœud sont du même ordre, ainsi que les capacités des liens, un bon choix de ces liens est réduit au problème de trouver des spanners du réseau avec de faibles indices de transmission.

Dans cette étude, nous considérons le cas où le graphe initial est une grille carrée. Les réseaux cœur ne sont généralement pas modélisés sous forme de grille, comme le montrent les modèles typiques trouvés dans SNDLib [280] et étudiés, par exemple, dans [286]. Cependant, un grand nombre de réseaux sont modélisés par une grille dans la littérature. Nous pouvons citer : des réseaux sans fil [354], tels que des des réseaux adhoc de capteurs sans fil [377], ou des réseaux sans fil aléatoires [155], des réseau d'antennes de lecture RFID [260], des réseaux mobiles ad hoc [368], des réseaux d'accès (*urban mesh access networks*) [358], des réseaux avec *femto cells* [323], des réseaux de collecte sans-fil [383], des réseaux cellulaires [277], des réseaux d'interconnexion [384], des grilles optiquement interconnectées (*optically interconnected arrays*) [397], graphes géométriques stochastiques [310]. Plus important encore, nous voulions bien comprendre la difficulté du problème sur des graphes simples. Nous avons donc choisi d'étudier les grilles carrées, car il s'agit d'une famille classique de graphes. Elles sont aussi un cas simple de graphes planaires. Résoudre le problème des grilles carrées donne des indices pour résoudre le cas plus général des graphes planaires avec degrés bornés, car ils peuvent être plongés dans une grille [399]. Le cas de la grille doit donc être considéré comme un paradigme ou un graphe planaire typique plutôt qu'un exemple réel d'un réseau existant.

Nous considérons le cas asymptotique avec n grand. Nous avons deux contributions principales.

D'un côté, il est bien connu que l'indice de transmission de la grille $n \times n$, G_n, est $\frac{n^3}{2}$ (voir Proposition 3). Une remarque importante est que la charge du routage associé sur les $2(n-1)^2 \sim 2n^2$ arêtes est plus faible dans le coin que dans le milieu de la grille. En utilisant ce fait, nous montrons comment construire des spanners de G_n avec beaucoup moins d'arêtes (seulement $13/18 \approx 72\%$ des arêtes) et les mêmes indices de transmission que G_n. Nous démontrons ensuite que nos spanners sont proches de l'optimum, en ce sens que nous prouvons qu'il est impossible de construire des spanners avec moins de $4/3n^2$ arêtes (66% des arêtes).

De l'autre côté, le plus petit spanner possible de G_n est un arbre couvrant. L'indice de transmission du meilleur arbre couvrant est asymptotiquement $\frac{3n^4}{8}$, voir la proposition 4. Lorsque nous considérons les spanners avec un plus grand nombre d'arêtes, la charge sur les arêtes diminue, de même que l'indice de transmission. Dans ce chapitre, nous étudions comment l'indice de transmission diminue lorsque nous augmentons le nombre de d'arêtes. Le tableau suivant résume nos résultats. Un fait intéressant est que, avec $n^2 + a^2$ arêtes (c.-à-d. a^2 arêtes supplémentaires), l'indice de transmission a ordre $\Theta(\frac{n^4}{a})$. Ceci est dû à la planarité de la grille.

	Arbre couvrant	Spanners Pour un entier a, $2 \leq a \leq n$		Grid
indice de transmission	$\frac{3}{8}n^4$	$\frac{1}{2a}n^4$	$\frac{1}{2}n^3$	$\frac{1}{2}n^3$
borne inférieure pour le nombre d'arêtes	n^2-1	$\simeq n^2 + \frac{4}{9}(0.1a)^2$	$\frac{12}{9}n^2$	
nombre d'arêtes des constructions	n^2-1	$n^2 + \frac{4}{9}a^2$	$\frac{13}{9}n^2$	$2n^2$

Nous n'énumérons que les résultats de la première contribution. Nous présentons ensuite les résultats et les preuves pour la deuxième proposition car la méthode est générale et pourrait être utilisée pour d'autres problèmes impliquant des spanners et des graphes planaires.

Proposition 3. *[ch5] L'indice de transmission de G_n est asymptiquement $\frac{n^3}{2}$.*

Proposition 4. *[ch5] Pour $n \geq 3$, l'arbre couvrant de G_n avec l'indice de transmission minimum est un arbre avec centroïde de degré 4 et 4 branches de tailles presque égales. Son indice de transmission est asymptotiquement $\frac{3n^4}{8}$.*

4.2.2 Spanners avec l'indice de transmission de la grille, $\frac{n^3}{2}$, mais beaucoup moins d'arêtes

Dans [Ci40], nous montrons d'abord qu'un spanner avec l'indice de transmission de la grille a au moins $\frac{4n^2}{3} = \frac{12n^2}{9}$ arêtes. Nous fournissons ensuite des spanners avec $\frac{13n^2}{9}$ arêtes. Nous énonçons juste les résultats correspondants.

Proposition 5. *Soit F un spanner de G_n tel que $\pi(F) \leq \frac{n^3}{2}$. F doit avoir, asymptotiquement, au moins $\frac{4n^2}{3}$ arêtes.*

Théorème 6. *Il existe un spanner de G_n, F_n, tel que $\pi(F_n) \sim \frac{n^3}{2}$ et son nombre d'arêtes est asymptotiquement égal à $\frac{13n^2}{9}$.*

4.2.3 Spanners avec indices de transmission dans $[\frac{n^3}{2}, \frac{3n^4}{8}]$ et bornes inférieures

Nous fournissons d'abord des spanners avec des indices de transmission dans l'intervalle $[\frac{n^3}{2}, \frac{3n^4}{8}]$ dans la proposition 7. Nous prouvons ensuite que ces spanners ont un nombre d'arêtes d'ordre optimal, voir la proposition 10.

Construction des spanners

Proposition 7. *Soit a un entier tel que, $2 \leq a \leq n$. Il exite un spanner $F_n(a)$ de G_n avec $n^2 + \frac{4}{9}a^2$ arêtes asymptotiquement et $\pi(F_n(a)) \leq \frac{n^4}{2a}$.*

Démonstration. Nous construisons un spanner de G_n, $F_n(a)$, de la manière suivante. Nous divisons la grille en a^2 secteurs. Un point se trouve dans le secteur (i, j) si ses coordonnées dans la grille (x,y) sont telles que $\frac{n}{a}i \leq x < \frac{n}{a}(i+1)$ et $\frac{n}{a}j \leq y < \frac{n}{a}(j+1)$. Chacun de ces secteurs a des $(n/a)^2$ sommets. Nous appelons *centre* du secteur (i, j) le sommet $((i + 1/2)n/a, (j + 1/2)n/a)$. Nous considérons la sous-grille $a \times a$ reliant les centres de tous les secteurs. Nous relions ensuite tous les sommets restants d'un secteur à son centre à l'aide d'un arbre couvrant. De cette façon, nous obtenons $F_n(a)$. La figure 4.1 fournit un schéma de la construction du spanner.

Nous construisons maintenant un routage R pour $F_n(a)$. La demande entre deux sommets d'un même secteur est acheminée sur l'arbre couvrant de leur secteur en utilisant l'unique chemin entre eux. La demande entre deux sommets de secteurs différents est d'abord acheminée vers le centre de leurs secteurs, puis est acheminée dans la grille $a \times a$.

Calculons la charge du routage R. Nous considérons d'abord les arêtes de la sous-grille $a \times a$. Nous savons qu'une grille $a \times a$ a un routage avec charge $a^3/2$ (Proposition 3). Ainsi, nous savons qu'elle a aussi un routage-w de charge $wa^3/2$ (un routage-w est un routage dans lequel chaque chemin à une charge w et non 1). Chaque sommet de la grille $a \times a$ reçoit la charge des $(n/a)^2$ sommets qui lui sont connectés. Ainsi, nous prenons $w = (n/a)^2$ et nous obtenons un w-routage de la grille $a \times a$ de charge $\frac{a^3}{2}2(\frac{n}{a}a)^4 = \frac{n^4}{2a}$.

Nous considérons maintenant une arête qui n'appartient pas à la grille $a \times a$. Les seuls chemins qui peuvent utiliser cette arête sont les chemins allant de n'importe quel sommet de la grille à un sommet de son secteur. Ainsi, sa charge est plus petite que $(n/a)^2 n^2 n^2 = \frac{n^3}{a^2}$. Cette

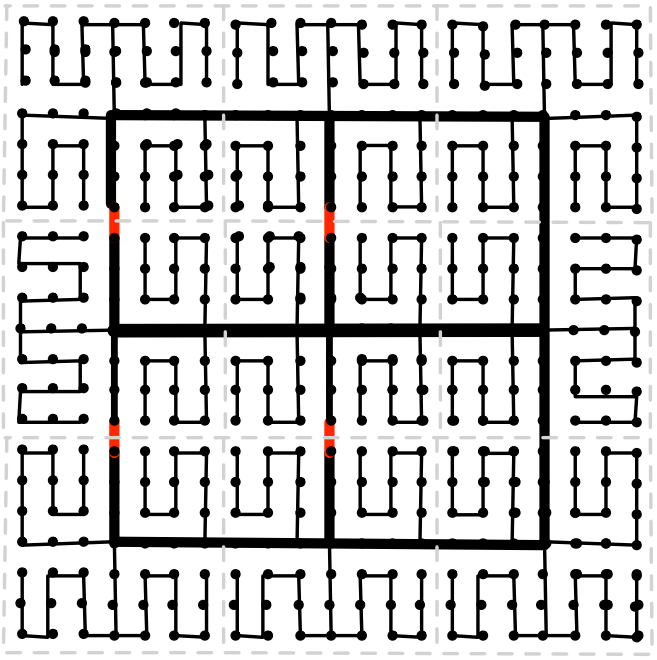


Figure 4.1: Spanner de la proposition 7 for $n = 21$ and $a = 3$. les arêtes de la grille $a \times a$ sont en gras. Arêtes qui ne sont pas celle de l'arbre couvrant de G_n sont en rouge. Les secteurs avec $(n/a)^2 = 7^2$ sommets sont séparés par des lignes grises hachurées.

charge est plus petite que la charge maximale sur la grille $a \times a$ dès que $a^2 \geq 2a$ ce qui signifie dès que $a \geq 2$.

Donc $\pi(F_n(a), R) = \frac{n^4}{2a}$.

Considérons maintenant le nombre d'arêtes du spanner $F_n(a)$. Le nombre d'arêtes nécessaires pour connecter tous les nœuds est $n^2 - 1$. Si nous choisissons bien ces arêtes, il suffit d'ajouter a^2 arêtes pour obtenir la grille $a \times a$ (voir Figure 4.1, les arêtes supplémentaires sont en rouge). $F_n(a)$ a donc $n^2 + a^2$ arêtes. Nous pouvons améliorer le spanner en utilisant les résultats de la section 4.2.2. Dans le théorème 6, nous montrons que nous pouvons trouver un spanner d'une sous-grille $a \times a$ avec $\frac{13}{9}a^2$ arêtes et un routage R' avec la même charge qu'une grille complète avec $2a^2$ arêtes. Ce faisant, nous obtenons un nouveau spanner $F_n(a)$, avec $n^2 + \frac{4}{9}a^2$ arêtes et $\pi(F_n, R') = \frac{n^4}{2a}$. □

On peut réécrire le résultat de la proposition 7 pour mettre en évidence l'impact des arêtes supplémentaires en général (Corollaire 8) et lorsque l'on part d'un arbre couvrant (Corollaire 9).

Corollaire 8. *Il existe :*

- *Un spanner de G_n avec $n^2 + p^2$ arêtes, et un indice de transmission de $\frac{n^4}{3p} \simeq 0.33\frac{n^4}{p}$;*
- *Un spanner de G_n avec $n^2 + p$ arêtes, et un indice de transmission de $\frac{n^4}{3\sqrt{p}} \simeq 0.33\frac{n^4}{\sqrt{p}}$.*

Démonstration. Direct par la proposition 7 en posant $p^2 = \frac{4}{9}a^2$ ou $p = \frac{4}{9}a^2$. □

Corollaire 9. *Il existe un spanner d'indice de transmission $\frac{1}{\alpha}\frac{3n^4}{8}$, ce qui est un facteur α moins que celui d'un arbre couvrant optimal, et qui utilise seulement $\frac{64}{81}\alpha^2 \simeq 0.79\alpha^2$ arêtes de plus qu'un arbre couvrant.*

Démonstration. On rappelle qu'un arbre couvrant optimal a un indice de transmission de $\frac{3n^4}{8}$, voir la proposition 4. En divisant par α, on obtient l'indice de transmission $\frac{3n^4}{8\alpha} = \frac{n^4}{2(4\alpha/3)}$. Cette valeur est atteinte par le spanner $F_n(a)$, avec $a = 4\alpha/3$. ce spanner a un nombre additionnel d'arêtes par rapport à un arbre couvrant égal à $\frac{4}{9}(4\alpha/3)^2 \simeq 0.79\alpha^2$. □

Bornes inférieures

Proposition 10. *Il n'existe pas de spanner de G_n avec n^2+p^2 arêtes et un indice de transmission inférieur à $\frac{1}{9\sqrt{12}}\frac{n^4}{p} \simeq 0.032\frac{n^4}{p}$.*

Démonstration. Considérons un spanner de G_n qui a n^2+p^2 arêtes. Nous construisons un multi-graphe de la manière suivante. Nous commençons par attribuer à chaque nœud un poids de 1. Ensuite, tant qu'il y a encore un sommet avec un degré 1 ou 2, nous supprimons ce sommet et les arêtes le reliant au graphe et nous répartissons son poids également entre ses voisins ; dans le cas où le sommet enlevé était de degré 2, nous connectons aussi ses deux voisins par la suite. A la fin de ce processus, on obtient un multi-graphe H tel que le nombre de ses sommets N' et le nombre de ses arêtes M' sont liés par l'équation suivante : $N' + p^2 = M'$. En effet, chaque fois qu'un sommet est enlevé dans le processus menant à H, le nombre d'arêtes est diminué de 1. Puisque tous les sommets en H ont un degré strictement supérieur à 2, nous avons $\frac{3}{2}N' \leq M'$. Cela implique avec l'équation précédente que $\frac{3}{2}N' \leq N' + p^2$. Par conséquent, nous avons $N' \leq 2p^2$. Notez que le poids total est égal à n^2. Nous appliquons maintenant la version pondérée du théorème du séparateur planaire [405] à H : il existe une partition des sommets de H en trois sous-ensembles A, S et B, telle que A et B ont chacun un poids d'au plus $2n^2/3$ et S a moins de $\sqrt{6}\sqrt{2p^2}$ sommets (Le graphe original est de degré borné 4.) et il n'y a pas d'arête liant un sommet dans A à un sommet dans B. Cela donne directement une coupe-arête du graphe original qui a moins de $2\sqrt{6}\sqrt{2p^2}$ arêtes et qui divise les sommets du graphe original en deux sous-ensembles de taille au plus $2n^2/3$. Par conséquent, tout routage de ce spanner induira, au moins sur une arête de la coupe, une charge supérieure à :

$$\frac{1}{3}n^2 \cdot \frac{2}{3}n^2 \cdot \frac{1}{2\sqrt{6}\sqrt{2p^2}} = \frac{1}{9\sqrt{12}}\frac{n^4}{p} \simeq 0.032\frac{n^4}{p}.$$

□

4.2.4 Simulations et efficacité des constructions

Pour montrer que notre méthode peut être appliquée dans la pratique, nous comparons, pour une série d'indices de transmission, le nombre d'arêtes des constructions proposées dans la Proposition 7. avec celles obtenues à l'aide des méthodes classiques de la littérature pour trouver des spanners économes en énergie, à savoir un Programme Linéaire en Nombres Entiers (ILP) et un algorithme heuristique (référencé comme *Algo*) et que l'on peut trouver par exemple dans [ch5]. L'ILP prend en entrée un réseau avec des capacités et renvoie le spanner avec le nombre minimum d'arêtes. *Algo* prend la même entrée et enlève de façon gloutonne les arêtes les moins chargées aussi longtemps qu'il est possible.

n	Indice de transmission	Construction	*Algo*	ILP
$a = 1$				
4	128	15	15	15
6	648	35	39	*
12	10368	143	168	*
18	52488	323	*	*
$a = 2$				
4	64	16	17	16
8	1024	64	64	*
12	5184	144	169	*
16	16384	256	*	*
$a = 3$				
9	1094	84	92	*
12	3456	147	150	*
15	8438	228	230	*
18	17496	327	*	*
$a = 4$				
8	512	72	74	*
12	2592	152	155	*
16	8192	264	*	*
$a = 5$				
5	1000	115	116	*
15	5063	240	242	*
20	16000	415	*	*

Table 4.1: Comparaison du nombre d'arêtes dans les constructions proposées dans le chapitre, avec celles données par l'algorithme heuristique (*Algo) et un ILP optimal. Une absence de valeurs (*) signifie que le calcul prend plus d'une heure.*

Dans le tableau 4.1, nous donnons le nombre d'arêtes des spanners pour différentes valeurs de a et n. Nous les comparons avec les meilleures valeurs trouvées par l'ILP et , *Algo*. Nous avons utilisé une grille de même taille $n \times n$ et nous avons fixé la capacité à l'indice de transmission du spanner correspondant, calculée dans la proposition 7 et également donnée dans la table 4.1. Nous montrons que nos constructions donnent de très bons résultats. Leur nombre d'arêtes est proche de l'optimal (lorsqu'il est possible de calculer cette valeur) et toujours meilleur ou égal à celui donné par *Algo*. De plus, nos constructions sont génériques, structurées et fournissent ainsi des solutions pour les grands réseaux, tandis que l'ILP et l'algorithme heuristique ne fournissent que des solutions particulières et ont un temps d'exécution important (sur un Quad-Core Intel Xeon 2.4 GHz avec 12 Go de RAM, l'ILP ne peut pas résoudre une instance de taille 4×4 en une heure et *Algo* ne peut plus être exécuté pour des tailles supérieures à 16×16).

4.2.5 Conclusion

Dans le présent document, nous nous sommes penchés sur le problème suivant : étant donnée une valeur cible, construire des spanners de la grille $n \times n$ avec un indice de transmission inférieur à la cible et le plus petit nombre d'arêtes possible. Nous avons proposé des spanners avec un certain nombre de arêtes d'ordre optimal et dans certains cas très proches de l'optimal. Plus

précisément,

i) Nous avons fourni les spanners de la grille avec $n^2 + \frac{4}{9}a^2$ arêtes et d'indices de transmission $\frac{n^4}{2a}$ (pour $2 \leq a < n$). Du côté des bornes inférieures, nous avons prouvé que les spanners à indice de transmission $\frac{n^4}{2a}$ et moins de $\simeq n^2 + \frac{4}{9}(0.1a)^2$ arêtes n'existent pas.

ii) De même, nous avons construit des spanners avec $\frac{13}{9}n^2$ arêtes et avec un indice de transmission égal à celui de la grille complète G_n. Pour la borne inférieure, nous avons prouvé que les spanners avec un tel indice de transmission doivent avoir au moins $\frac{12}{9}n^2$ arêtes.

Même si nos constructions sont assez serrées, elles ne sont pas optimales et cela laisse deux problèmes ouverts sur le plan théorique : Premièrement, réduire l'écart entre notre borne inférieure et notre borne supérieure. Notons que nous croyons qu'il peut être très difficile de combler complètement cet écart, car cela implique peut-être de déterminer une inégalité isopérimétrique serrée pour les graphes planaires. Deuxièmement, déterminer s'il existe ou non des spanners avec $\frac{12}{9}n^2$ arêtes et indice de transmission $\frac{n^3}{2}$.

De plus, dans ce travail, nous nous sommes concentrés sur la grille carrée et visions à fournir des résultats proches des résultats optimaux. Nous avons donc étudié en détail un cas particulier d'un problème général de *théorie des graphes extrêmes*, dans lequel le but est de trouver le meilleur graphe (celui avec le moins d'arêtes) étant donné une borne sur son indice de transmission et quelques contraintes supplémentaires (comme être planaire, être un sous-graphe de la grille, ...). Nous croyons que ce problème de graphe extrême est intéressant. Cependant, il n'a pas été beaucoup abordé et la plupart des questions sont largement ouvertes (voir [Ci39]).

Du point de vue pratique, ces spanners sont importants pour les réseaux économes en énergie, dans lesquels le trafic doit être acheminé sur le réseau, tout en utilisant un nombre minimum d'équipements. Les équipements non utilisés sont ensuite éteints pour économiser de l'énergie. Il convient de noter que le cas considéré dans le présent document est celui d'un trafic uniforme de tout le monde vers tout le monde (all-to-all) avec des capacités de liens homogènes. Toutefois, les résultats de l'étude établissent des limites utiles pour des contextes plus généraux : i) Si le trafic n'est pas all-to-all, les résultats fournis dans ce chapitre (pour un trafic all-to-all) donnent une borne supérieure pour le nombre de liens nécessaires dans le spanner. ii) Si les capacités ne sont pas homogènes, nos résultats fournissent également une borne supérieure pour le nombre d'arêtes, si nous fixons la capacité à la capacité minimale d'un lien dans le cas hétérogène (et une borne inférieure si nous fixons la capacité à la capacité maximale d'un lien). Il serait intéressant d'étudier ces cadres plus généraux à l'avenir.

4.3 Comment construire des graphes avec un indice de transmission bas et un nombre limité d'arêtes

4.3.1 Introduction

Problème. Dans cette section, notre objectif est de fournir, pour un nombre donné de sommets n et un nombre donné d'arêtes k, des graphes avec l'indice de transmission minimum, ou au moins des graphes avec des indices de transmission faibles. Pour un n donné, nous étudierons comment le nombre d'arêtes d'un graphe affecte son indice de transmission. Formellement, nous définissons le problème de conception suivant :

GRAPHE-(n, e) MINIMALEMENT CONGESTIONNÉ *:Etant donné $n, e \in \mathcal{N}$ tels que $e \geq n-1$, le problèmes est double :*
- Déterminer quel est l'indice minimum de transmission, noté par $\pi^*(n, e)$, qui peut être obtenu par un graphe G=(V,E) avec $|V| = n$ vertices et $|E| = e$ edges.
- Exhiber des graphes avec avec index de transmission $\pi^*(n, e)$.

Au meilleur de notre connaissance, le problème de la conception d'un graphe avec un indice minimum de transmission n'a pas été étudié, quand la contrainte principale est le nombre d'arêtes. En effet, la plupart des résultats existants sont limités à déterminer π pour des classes de graphes classiques, ou prennent en compte d'autres contraintes, comme un degré borné. Le nombre d'arêtes est une contrainte naturelle, par exemple si vous voulez minimiser le coût d'installation du réseau ou la consommation d'énergie du réseau [343, ch5].

Indice de transmission et distances. L'indice de transmission est fortement lié aux propriétés de distance d'un graphe. En effet, une borne inférieure naïve habituelle sur π (Average Distance Bound) est :

$$\pi(G = V, E) \geq \frac{\sum_{(u,v)\in V^2} D(u, v)}{|E|} = \frac{\bar{D}_G |V|^2}{|E|} = 2|V| \frac{\bar{D}_G}{\bar{d}_G},$$

où $D(u, v), d(v), \bar{D}_G$ et $\bar{d}_G$ désignent respectivement la fonction de distance, la fonction de degré, la distance moyenne en G et le degré moyen en G. De plus, cette borne est atteinte si et seulement s'il existe un routage par plus courts chemins qui est équilibré sur les arêtes. C'est le cas des graphes tels que les cycles, les tores et tout autre graphe arêtes-transitif (voir [388]). Cela indique que la résolution de notre problème de conception est fortement liée à la recherche de graphes avec une faible distance moyenne. Le problème Degré-Diamètre ou (Δ, D)-DESIGN PROBLEM consiste à trouver le graphe avec degré Δ et diamètre D avec le nombre maximum de sommets (ou réciproquement il s'agit de minimiser le diamètre d'un graphe régulier Δ). Il s'agit d'un problème assez complexe qui a fait l'objet d'études approfondies (voir [213] pour un survey récent). Même après 30 ans d'efforts constants, les constructions génériques sont encore très loin d'être optimales. Ainsi, puisque les bons graphes-(n, e) devraient ressembler aux graphes (Δ, D), nous pouvons nous attendre à ce que notre problème soit complexe. Mais nous pouvons aussi espérer pouvoir utiliser les résultats concernant le problème (Δ, D) dans notre contexte.

Sur l'indice de transmission entier. Nous allons nous concentrer dans la suite sur l'indice fractionnaire de transmission (c'est-à-dire que les routages peuvent être fractionnaires). Le problème de déterminer l'indice de transmission entier, noté $\pi_i(G)$, est $\mathcal{NP}$-complet [307], et,

utiliser l'index de transmission entier comme paramètre réseau principal exigerait trop d'efforts. Alternativement, nous pouvons utiliser l'indice de transmission fractionnaire, qui est facile à calculer, parce que l'écart d'intégralité pour l'indice de transmission est suffisamment petit, à savoir

$$\pi_i(G) \leq \pi(G) + O(\sqrt{\log_2(|V|)|E|}),$$

comme montré dans [402]. Puisque, pour la plupart des cas que nous allons étudier, nous avons $\sqrt{\log_2(|V|)|E|} = o(\pi(G))$, nous serons presque toujours dans le cas pour lequel $\pi_i(G) = (1 + o(1))\pi(G)$. Ainsi, nous étudierons principalement l'indice de transmission fractionnaire.

Contributions et plan

- Dans la section 4.3.2, nous considérons notre problème de conception pour les graphes généraux, c'est-à-dire lorsque la seule contrainte de conception est le nombre d'arêtes. Lorsque le nombre d'arêtes est $k(n-k), k \in \mathcal{N}$, nous caractérisons les graphes avec l'indice de transmission minimum et les graphes bipartis complets $K_{k,n-k}$ fournissent, entre autres, des solutions optimales génériques simples et symétriques. Entre ces valeurs, la fonction $\pi^*(n, e)$ suit, de façon assez surprenante, une fonction avec sauts (voir les propositions 3.4 et 3.5 [R84]. Une conséquence pratique est que l'ajout de quelques arêtes (moins de $\sim n$) n'augmente pas significativement le débit.

- Dans la section 4.3.3, motivé par les réseaux de télécommunication, nous étudions le cas des graphes de degrés bornés. Nous fournissons des graphes presque optimaux pour différentes valeurs du degré maximum Δ. Nous nous concentrons ensuite sur les graphes avec un petit nombre d'arêtes ($\Delta = 3$) car ils correspondent à la plage de valeurs pour laquelle l'indice de transmission change considérablement. Nous déterminons très précisément comment se comporte l'indice de transmission minimum et évolue de $\Theta(n^2)$ à $\Theta(n \log n)$ lorsque le nombre d'arêtes passe de $n - 1$ à $n + \frac{n}{2}$. Nous développons également une méthode qui nous permet de simplifier le problème de conception en considérant le squelette du graphe.

- Puis, dans la section 4.3.4, nous examinons le cas $e = n + k$ avec un petit $k \in \{1, 2, 3\}$. Nous déterminons l'indice minimum de transmission exactement pour n'importe quel n. Ceci est possible parce que la structure principale du graphe, que nous appelons squelette, est finie, de sorte que nous pouvons tous les explorer et utiliser des arguments de poids afin de réduire le problème à un problème fini. Certains des résultats, comme par exemple la proposition 16, sont étonnamment contre-intuitifs.

- Enfin, dans la section 4.3.5, nous fournissons des graphes cubiques avec *un petit nombre de sommets*, c'est-à-dire pour $n \in [4, 22]$. Ces graphes sont intéressants, parce que, comme nous le verrons, leur structure peut être utilisée comme un squelette pour construire de bons graphes avec un petit nombre d'arêtes et des tailles arbitraires.

4.3.2 Graphes généraux avec indice de transmission minimum

Dans cette section, nous discutons rapidement des résultats prouvés dans [R84]. Nous étudions la conception de graphes à congestion minimale pour des nombres donnés de sommets n et d'arêtes e. Nous donnons d'abord une borne inférieure triviale de $\pi^*(n, e)$, l'indice minimum d'un graphe-(n, e) (nous noterons ainsi un graphe avec n sommets et e arêtes). Nous fournissons ensuite des familles de graphes à congestion minimale atteignant cette limite pour certains couples de valeurs (n, e), par exemple des graphes bipartis complets $K_{i,n-i}$, des graphes complets k-parties,

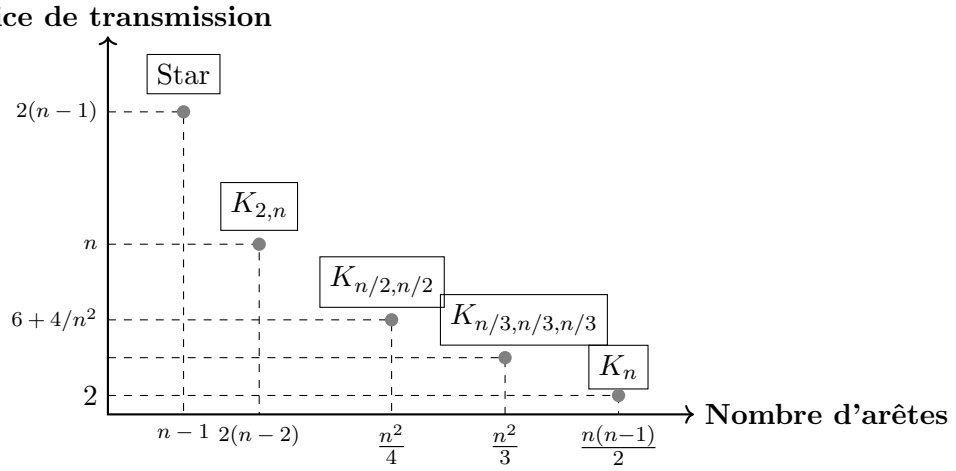


Figure 4.2: Indices de transmission des graphes à congestion minimum avec n sommets en fonction de leurs nombres d'arêtes.

ou des graphes de Kneser, voir la figure 4.2. Ces graphes sont des graphes arêtes-transitifs de diamètre 2. En particulier, nous montrons que les graphes $K_{i,n-i}$ ($i \in \mathcal{N}, i \leq \lfloor n/2 \rfloor$) sont des graphes-$(n, i(n-i))$ optimaux avec indice de transmission $\pi^*(n,e) = 2(\frac{n(n-1)}{e} - 1)$. Enfin, nous étudions le comportement de $\pi^*(n,e)$ lorsque e varie entre deux cas "parfaits", soit de $i(n-i)$ à $(i+1)(n-(i+1))$. Étonnamment, π^* suit une fonction avec sauts et saute soudainement de $\pi^*(n, i(n-i))$ à $\pi^*(n,(i+1)(n-(i+1))$.

4.3.3 Graphes avec degré borné avec un indice de transmission bas

Dans la section précédente, nous avons fourni des familles de graphes presque optimales. Cela résout la question de la congestion minimale des graphes dans le cas général. Nous étudions maintenant des graphes avec une contrainte sur le degré maximum (Δ désignera le degré maximum). La motivation vient des réseaux de télécommunication et des réseaux d'interconnexion pour lesquels le degré des nœuds est souvent faible, voir par exemple [280, Ci40]. Dans [R84], nous considérons d'abord le cas général $\Delta \geq 3$ ($\Delta = 2$ est trivial), voir la figure 4.3. Nous réussissons à déterminer comment l'indice de transmission chute de $\pi(n,e) = n^2/4$ à $\frac{2}{3}n\log_2 n$ lorsque le degré moyen passe de 2 à 3. Donc, nous nous concentrons sur des graphes avec un petit nombre d'arêtes, à savoir des graphes avec un degré moyen $\overline{\Delta} \in [2,3[$, c'est-à-dire avec $e \in [n, \frac{3}{2}n]$, et nous étudions la transition de $\pi(n,e)$ de $\frac{n^2}{4}$ (cas de l'étoile) à $\Theta(n\log n)$ (graphe cubique aléatoire) lorsque le nombre d'arêtes e passe de $n-1$ à $\frac{3}{2}n$.

Graphes avec degré borné Δ : quelques remarques.

Pour $\Delta = 3$, quand $e = \frac{3n}{2}$, les graphes tels que le shuffle exchange (voir par exemple [406] pour une définition) fournissent des constructions génériques déterministes pour lesquelles $\pi(G) \leq n\log_2 n$ (c'est un résultat classique pour les personnes qui étudient le débit du réseau, on peut voir [231]). Comme l'utilisation de la borne de Moore (cette borne indique par comptage direct que la distance moyenne dans un graphe de degré borné Δ est d'ordre $\log_{\Delta-1}(n)$, voir par exemple [213]), on peut prouver que $\pi^*(n, \frac{3n}{2}) \geq \frac{2}{3}n\log_2 n(1+o(1))$. Les bornes inférieure et supérieure correspondent à un facteur $\frac{2}{3}$. Nous comblons cet écart en prouvant que les graphes

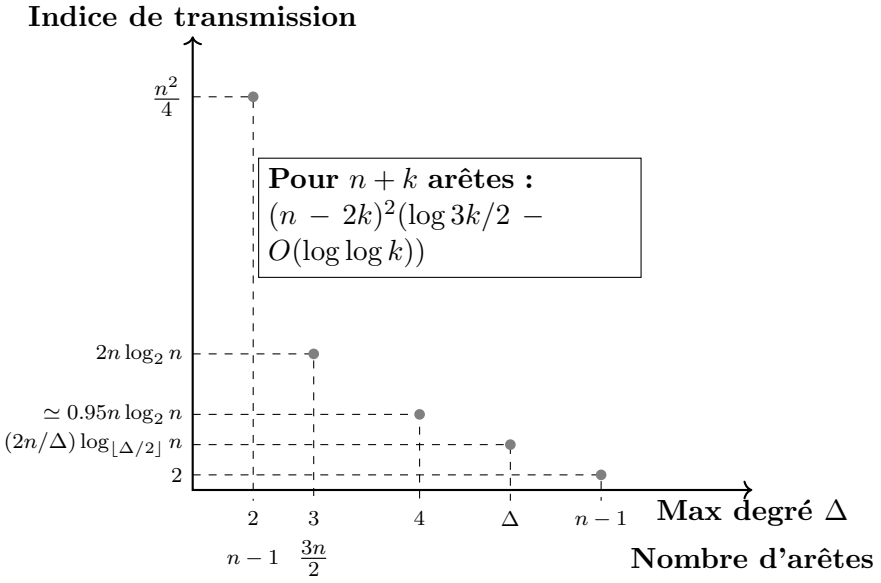


Figure 4.3: Indices de transmission de graphes à degrés bornés avec n sommets en fonction de leur nombre d'arêtes.

cubiques aléatoires sont presque optimaux puisqu'avec forte probabilité, ils sont tels que $\pi(G) = \frac{2}{3}n\log_2 n(1+o(1))$. Pour des valeurs plus grandes de Δ, les graphes de de Bruijn et leurs variantes fournissent des graphes Δ-réguliers dont l'indice de transmission est du bon ordre (voir la figure 4.3). Ainsi, lorsque le degré est délimité par Δ, la valeur de $\pi(n, \frac{\Delta}{2}n)$ est relativement bien comprise (voir [403, 244]), et les structures proches de l'optimal sont obtenues en utilisant des graphes de Bruijn ou de légères variantes de celui-ci. En effet, d'une part, la borne de Moore implique que :

$$\pi^*(n, \frac{\Delta}{2}n) \geq \frac{2}{\Delta}n\log_{\Delta-1} n(1-o(1)).$$

D'autre part, pour les graphes de de Bruijn, on a (voir [403, 244])

$$\pi(n, \frac{\Delta}{2}n) \leq \frac{2}{\Delta}n\log_{\lfloor\frac{\Delta}{2}\rfloor} n.$$

L'argument qui fournit la borne ci-dessus pour les graphes de de Bruijn avec degré $\Delta = 2d$ et d^n sommets, est qu'il existe dans ce graphe un routage intégral qui est uniforme sur les arêtes et qui relie chaque couple de sommets avec un chemin de longueur exactement n. Cette longueur n'est qu'un facteur constant supérieur à la distance moyenne minimale prévue par la borne de Moore, de sorte que le rapport entre la borne supérieure et la borne inférieure est d'au plus 3 et diminue avec Δ.

Donc notre but est de comprendre ce qui se passe quand $\overline{\Delta}$ passe de 2 à 3. Nous étudions donc les graphes avec $\Delta = 3, \overline{\Delta} \leq 3$ qui sont en effet des graphes cubiques dans lesquels les arêtes sont subdivisées et sur lesquels on peut attacher des arbres de degré ≤ 3.

Bornes inférieure et supérieures pour le cas $e \in [n, \frac{3}{2}n]$ pour $\Delta \leq 3$

Nous avons prouvé les deux résultats suivants.

Proposition 11. *Si G est un graphe-$(n, n+k)$ avec $\Delta = 3$ alors $\pi(G) \geq \frac{(n-2k)^2}{3k}(\log(3k/2) - O(\log\log(k))$.*

Proposition 12. *Il existe des graphes-$(n, e = n+k)$ cubiques tels que $\pi(G) \leq \frac{n^2}{3k}\log_2(k)(1 + o(1))$.*

Je ne mets pas les preuves formelles ici qui sont assez longue et calculatoires, mais j'explique leurs principes qui est fondée sur la *notion de squelette.*

L'approche par squelette L'idée principale est de remplacer le graphe G par son squelette qui est défini dans la suite comme un graphe cubique régulier avec le même indice de transmission que G, ce squelette est obtenu par la réduction des arbres ou des chemins aux noeuds auxquels ils sont attachés. Ces nœuds devenant pondérés par la taille des arbres ou chemins qui ont été réduits (la définition formelle est donnée ci-dessous).

Tout d'abord, définissons deux quantités importantes $\mathbf{W}_1$ (resp. $\mathbf{W}_2$), qui sont intuitivement les poids maximum (c'est-à-dire le nombre de sommets) qui peuvent se trouver à l'intérieur d'un arbre attaché (resp. à l'intérieur d'une arête sous-divisée) du graphe.

$$\pi^*(n, n+k) = 2\mathbf{W}_1(n - \mathbf{W}_1)$$
$$\pi^*(n, n+k) = 2\mathbf{W}_2(n - \mathbf{W}_2)/2 = \mathbf{W}_2(n - \mathbf{W}_2)$$

Définition 13 (Squelette)**.** Considérons un graphe $G = (V, E)$ avec n sommets et $n+k$ arêtes. Le *squelette de G, désigné par $S[G]$*, est le graphe $S[G] = (V_S, E_S, w_S)$ avec $w_S(u)$ le poids du sommet u, obtenu à partir de G par la procédure inductive suivante en deux étapes.

- **Éliminer les sommets de degré 1.** Nous définissons $G' = (V, E, w)$, avec $w(u) = 1, \forall u \in V$. Au début de la procédure, tous les nœuds ont un poids 1. Tant que G' a une feuille u avec père v, supprimez u et mettez à jour le poids de v à $w(u) + w(v)$.
- **Éliminer certains sommets de degré 2.** Les sommets de degré 3 de G' sont maintenant reliés par des chemins formés par des nœuds de degré 2 qui sont potentiellement pondérés (si un arbre était initialement attaché en G à ces nœuds). Nous remplaçons alors chacun de ces chemins avec le poids total w par un chemin avec seulement deux arêtes et un nœud de poids w.

Proposition 14 (Squelette)**.** *Etant donné un graphe-$n, n+k$, son squelette a les propriétés suivantes :*

- *Son degré maximum est $\Delta = 3$.*
- *Il a au plus $2k$ sommets (Notez que cela ne dépend pas de n).*
- *La somme des poids de tous les sommets est notée* **ns** *et est telle que* $\mathbf{ns} \stackrel{def}{=} n - 2k$.
- *Le poids de chaque sommet est moins que $\mathbf{W}_2$.*
- *Son indice de transmission (pondéré) est au plus $\pi(G)$.*

Le lemme suivant indique comment on peut construire des graphes avec indice de transmission $\pi^*(n, n+k)$.

Lemme 15 (squelette)**.** *Tout graphe G avec n sommets et $n+k$ arêtes et avec un indice de transmission minimum $\pi^*(n, n+k)$ peut être construit comme suit :*

- *Prenez un graphe cubique (avec potentiellement des arêtes multiples) avec au maximum $x = 2k$ sommets, mettez au maximum 5 nouveaux* sommets-arêtes *sur les arêtes et affectez un poids total $n - x$ aux nœuds-arêtes de telle sorte que i) chaque nœud-arête a un poids inférieur à $\mathbf{W}_1$, ii) chaque arête contient un poids inférieur à $\mathbf{W}_2$.*
- *Si au lieu de cela nous ne subdivisons les arêtes qu'une fois, et utilisons des poids inférieurs à $\mathbf{W}_2$, le plus petit indice de transmission que nous pouvons atteindre est une borne inférieure sur $\pi^*(n, k)$.*

Muni de la notion de squelette et de ses propriétés ci-dessus, nous pouvons maintenant expliquer la preuve de nos deux propositions.

- Pour prouver la proposition 11 et trouver une borne inférieure sur $\pi(G)$, il est suffisant d'en trouver une pour $\pi(S[G])$ (en raison de la proposition 14 et du lemme 15). Pour ce faire, nous avons compté la distance totale entre les sommets situés à l'intérieur des arêtes (représentés par des poids).
- Pour prouver la borne supérieure de la proposition 12, nous exhibons un graphe avec cet indice de transmission. On montre d'abord que les graphes cubiques aléatoires ont un indice de transmission minimum. Ensuite, on en choisit un de bonne taille ($v = 2k$ sommets and $e = 3k = v + k$ arêtes) comme squelette. Enfin, on subdivise chaque arête en deux arêtes reliées par un sommet et on attache sur ce sommet un arbre binaire de taille $\mathbf{W}$ tel que $(\frac{\Delta \mathbf{W}}{2} + 1)2k = n$, c.-à-d., $\mathbf{W} = \frac{n}{\Delta k} - \frac{2}{\Delta}$.

4.3.4 Indice de transmission de graphes cubiques ($\Delta = 3$) avec peu d'arêtes : $e = n + k$

Quand k est grand, nous avons fourni dans la section 4.3.3 des bornes inférieures et supérieures asymptotiques qui coïncident pour la congestion minimale. Cela implique que $\pi^*(n, n+k)$ se comporte comme $\Theta(\frac{n^2}{k} \log \frac{n}{k})$ quand k et n sont grands. Donc, pour avoir une image complète de la situation, nous devons toujours comprendre le cas des graphes $(n, n+k)$ lorsque k est fixé. Dans cette section, nous répondons à cette question, c'est-à-dire que nous résolvons le MIN-CONGESTION DESIGN PROBLEM, pour des graphes avec des n arbitraires, mais de petites valeurs de k. Nous avons dérivé des constructions optimales pour $k = 0, 1, 2, 3$ in [Ci40]. Certaines solutions sont surprenantes. Par exemple, pour les graphes avec n arêtes, les meilleurs sont des cycles avec 7 nœuds (pourquoi 7 ?) auxquels sont attachés des arbres avec environ $n/7$ nœuds. Je ne présente ici que la construction pour $k = 3$, car elle est très surprenante (voir Figure 4.4).

Graphes-$(n, n+2)$: Arbres + $(k = 3)$ arêtes supplémentaires. Subdivision non-uniforme du K_4 Le résultat suivant est plutôt surprenant puisque intuitivement une subdivision uniforme (ou au moins symétrique) du K_4 devrait fournir une solution optimale. Mais un phénomène similaire à celui que nous avons déjà rencontré dans le cas $k = 1$ (le C_7) se reproduit d'une manière un peu plus complexe.

Proposition 16.

$$\pi^*(n, n+2) = \frac{20}{11^2} n^2$$

Démonstration. Nous prouvons d'abord la borne inférieure dans le lemme 17, puis nous fournissons une borne coincidante (construction). dans le lemme 19. □

Nous prouvons d'abord la borne inférieure :

Lemme 17.

$$\pi^*(n, n+2) \geq \frac{20}{11^2} n^2$$

Démonstration. Nous examinons les squelettes possibles. Rappelons que le degré maximum est de 3 et que, lors de la formation des squelettes, nous enlevons les nœuds avec degrés 1 et 2. Cela signifie que nous avons soit un cycle + 2 cordes, ce qui signifie soit (a) une subdivision du K_4 ou (b) 2 doubles arêtes reliées par un pont double, soit 2 cycles reliés par deux arêtes ce qui correspond au même cas que (b).

- **Cas (b):** Dans ce cas, la charge totale du pont double est $2 \cdot \frac{n}{2} \cdot \frac{n}{2}$ pour 2 arêtes. Cela implique que $\pi(G) \geq \frac{n^2}{4} > \frac{20}{11^2} n^2$.

- **Cas a:** D'abord, remarquez que la charge maximale d'un arbre attaché est au plus $\frac{n}{11}$. Sinon, $\pi(G) \geq 2 \times \frac{1}{11} n \times \frac{10}{11} n = \frac{20}{11^2} n^2$ et nous avons prouvé la borne inférieure.

  Maintenant, appelons coupe-$[a, b]$ une coupe telle que ces deux côtes ont une charge totale dans $[a, b]$. Nous voulons montrer qu'il existe une coupe-$[\frac{5}{11} n, \frac{6}{11} n]$ avec bord 3. Cela implique que

  $$\pi(G) \geq 2 \frac{\frac{5}{11} \times \frac{6}{11}}{3} = \frac{20}{11^2} n^2.$$

  Dans ce cas, nous considérons, pour chaque sommet $v_i, i \in [3]$du squelette, les coupes formées par un sommet v_i et ses trois arêtes adjacentes avec leurs poids. Chacune de ces 4 coupes a une frontière de bord 3. Comme chaque apparait dans exactement deux coupes, quand on somme les poids de ces coupes, on compte deux fois le poids total. Ainsi, au moins une de ces coupes a un poids au moins $\frac{n}{2}$. Maintenant, tant que le poids de la coupe est plus grand que $\frac{6n}{11}$, nous diminuons le poids de la coupe en enlevant pas à pas le dernier arbre attaché à l'une des trois arêtes divisées. Le poids maximum d'un arbre est $\frac{n}{11}$. Ainsi à chaque étape, on diminue le poids d'un montant plus petit que $\frac{1}{11} n$. Il s'ensuit qu'on obtient à un moment une coupe-$[\frac{5}{11} n, \frac{6}{11} n]$.

□

Pour finaliser la preuve, il est suffisant de fournir une construction. L'argument de la borne inférieure donne presque directement la construction suivante.

Définition 18 ($K_{4,sub}(n)$)**.** Le graphe $K_{4,sub}(n)$ est obtenu en divisant 5 arêtes du K_4 deux fois et une arête une fois. Nous ajoutons ainsi 11 nouveaux sommets. Ensuite, nous attachons un arbre avec poids $\frac{n}{11}$ à chaque sommet, voir la figure 4.4.

Lemme 19.

$$\pi^*(n, n+2) \leq \pi(K_{4,sub}(n)) = \frac{20}{11^2} n^2$$

Notez que l'indice de transmission de la division uniforme du K_4 qui intuitivement aurait due être optimale est $\frac{n^2}{6}$.

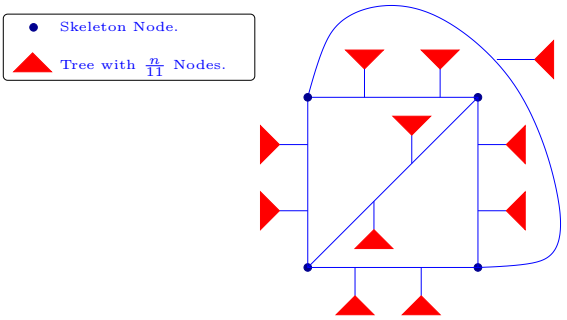


Figure 4.4: Division optimale du K_4 avec $n+3$ arêtes.

4.3.5 Graphes avec un petit nombre de sommets ($\Delta = 3$)

Nous avons vu dans les sections 4.3.3 et 4.3.4 l'importance d'avoir de bons squelettes pour construire des graphes avec des indices de transmission faibles. Dans le tableau 4.2, nous présentons des graphes avec un petit nombre de sommets qui ont un indice de transmission minimum. Ces graphes peuvent servir de squelettes pour construire des familles de graphes avec un nombre arbitraire de sommets. Dans certains cas, l'optimalité est facile à prouver en utilisant :

- la borne de Moore. Dans un graphe cubique, et pour un sommet donné, le nombre de sommets qui sont à distance $0, 1, 2, 3, \ldots$, sont respectivement, au maximum $1, 3, 6, 12, \ldots$. Lorsque ces bornes sont atteintes pour tous les sommets d'un graphe cubique, ce dernier minimise $\mathcal{L} = 2|V|\frac{\overline{D(G)}}{\overline{d}(G)}$ parmi tous les graphes de même taille et avec degré 3. Lorsque le graphe est optimal pour la borne de Moore et qu'il est arête-transitif, son indice de transmission est minimal. C'est le cas pour $n = 6, 14$;

- des arguments de coupes, pour $n = 4, 8, 10$.

Dans d'autres cas ($n = 12, 16, 18$), les arguments génériques ne fournissent pas les bornes supérieures et inférieures correspondantes. Nous avons vérifié tous les graphes cubiques possibles ([167]). Pour le dernier cas ($n = 20$), nous n'avons pas pu vérifier tous les graphes possibles. Néanmoins, nous fournissons un graphe avec un indice de transmission très proche de la borne de Moore. Nous avons considéré ce graphe, comme il est optimal pour le cas $(3, 3)$ du problème (Δ, D).

4.3.6 Conclusion

Dans ce chapitre, nous avons fourni une compréhension de base de l'interaction entre l'indice de transmission d'un graphe et son nombre d'arêtes. Nos bornes sont pour la plupart asymptotiquement serrées et expliquent comment la transition se produit entre les graphes fortement congestionnés (arbres, chemins, ...) et les graphes cubiques réguliers qui ont beaucoup moins de congestion.

Certains résultats, comme le comportement à saut de la section 4.3.2 ou des structures optimales irrégulières, sont également amusants, puisqu'ils sont inattendus. Enfin, nous croyons que notre travail ouvre de nombreuses questions :

- **Petits cas** : Dans le cas de quelques arêtes supplémentaires, nous nous sommes arrêtés à 3 arêtes supplémentaires (et même dans ces cas, les bornes inférieures ne sont pas totalement triviales alors que la construction découle directement d'argument de bornes inférieures)

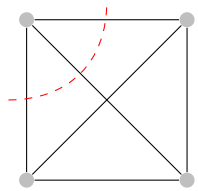
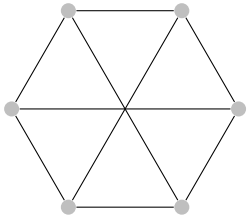
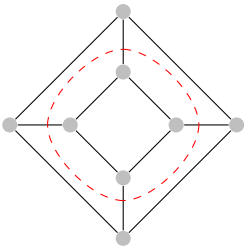
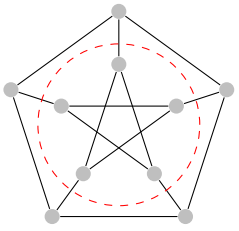
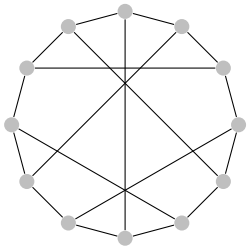
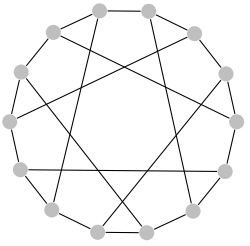
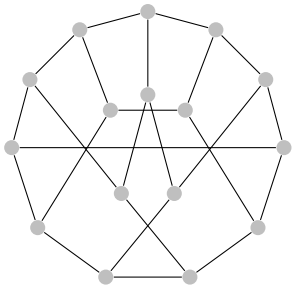
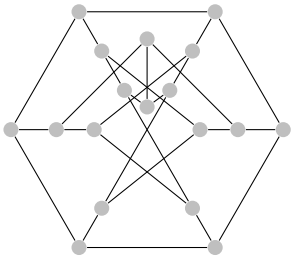
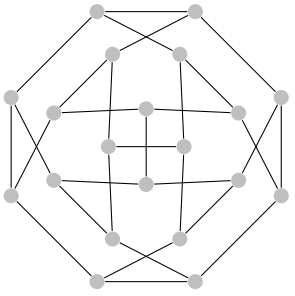
$n = 4, \pi = \mathcal{L} = 2$ $\mathcal{L}$ donné par une coupe	$n = 6, \pi = \mathcal{L} = 4.66...$ $\mathcal{L}$ donné par la borne de Moore	$n = 8, \pi = \mathcal{L} = 8$ $\mathcal{L}$ donné par une coupe
		
$n = 10, \pi = \mathcal{L} = 10$ $\mathcal{L}$ donné par une coupe $\pi(n, n+5) \leq 0.163n^2$	$n = 12, \pi = \mathcal{L} = 14.26...$ $\mathcal{L}$ trouvé par brute force $\pi(n, n+6) \leq 0.152n^2$	$n = 14, \pi = \mathcal{L} = 18$ $\mathcal{L}$ donné par la borne de Moore $\pi(n, n+7) \leq 0.138n^2$
		
$n = 16, \pi = \mathcal{L} = 22$ $\mathcal{L}$ trouvé par brute force $\pi(n, n+8) \leq 0.126n^2$	$n = 18, \pi = \mathcal{L} = 26.66...$ $\mathcal{L}$ trouvé par brute force $\pi(n, n+9) \leq 0.118n^2$	$n = 20, \pi = 30.84..., \mathcal{L} = 30$ (optimalité non prouvée) $\pi(n, n+10) \leq 0.110n^2$
		

Table 4.2: Petits graphes cubiques avec un indices de transmission minimum.

et nous avons indiqué une feuille de route pour automatiser le processus pour les petites valeurs de k. Il peut être intéressant d'aller plus loin et de comprendre si des graphes optimaux avec arêtes k supplémentaires sont construits en utilisant un graphe cubique optimal avec $\frac{k}{2}$ sommets (nous avons déterminé de tels graphes jusqu'à $k = 22$). Par exemple : *est-ce que la famille de graphes optimaux avec* 5 *arêtes supplémentaires se construit en utilisant le Petersen et en le subdivisant correctement ? Et, si c'est le cas, comment trouver la meilleure subdivision (c'est-à-dire résoudre le problème de l'allocation de poids sur les petits graphes) ?*

- **Construction à partir de squelettes** : Étant donné un squelette, nous ne savons pas comment affecter les poids afin de minimiser l'indice de transmission du graphe résultant. Ce problème peut être exprimé comme un problème quadratique non convexe et nous supposons qu'il est NP-Complet.

Part II

Réseaux logiciels virtualisés (SDN, NFV, SFC)

Chapter 5

Introduction

La gestion distribuée des réseaux existants et la difficulté pour configurer le matériel freine le déploiement de politiques nouvelles par les opérateurs. La virtualisation et la "logiciérisation" du réseau, apportées par les paradigmes SDN et NFV, portent la promesse de simplifier cette gestion du réseau en apportant l'abstraction et la programmabilité du réseau sous-jacent. Ces deux technologies apporteraient aux opérateurs de réseaux plus de liberté dans le déploiement de nouvelles politiques à l'échelle du réseau, tout en réduisant de manière significative leurs coûts.

SDN. Le paradigme des réseaux logiciels ou SDN est un paradigme émergent qui propose une alternative aux réseaux existants, en découplant le plan de données et le plan de contrôle. Dans les réseaux existants, les routeurs transmettent des paquets et échangent également des messages de contrôle avec d'autres routeurs de la topologie pour prendre des décisions de routage local. Dans les réseaux SDN, les routeurs sont relégués à de simples dispositifs de commutation de paquets, et un ou plusieurs contrôleurs gèrent le plan de contrôle. Cette approche simple permet aux administrateurs réseau d'obtenir un meilleur contrôle sur le trafic de leur réseau. En effet, SDN rend possible

(i) un relevé des données de métrologie des nœuds et des liens ;

(ii) une gestion centralisée de ces nœuds, permettant d'optimiser et de faciliter l'optimisation du réseau par rapport à un contrôle distribué.

Grâce à leur contrôle centralisé, les réseaux SDN permettent une meilleure gestion des ressources du réseau, l'introduction de la qualité de service et l'introduction de politiques dynamiques, comme l'efficacité énergétique.

Le protocole OpenFlow est l'instanciation principale du concept SDN pour le moment et est supporté par les principaux fabricants, par exemple HP, Juniper, IBM ainsi que des commutateurs virtuels open-source comme Open vSwitch [422], qui est au cœur de solutions de gestion du cloud comme, par exemple, OpenStack [419]. Ce nouveau paradigme est promu par l'Open Network Foundation (ONF), créée en 2011 et composée d'entreprises telles que Google, Verizon ou Facebook. Google a présenté son premier réseau SDN entièrement opérationnel, nommé B4 [218], et a montré qu'il pouvait utiliser un réseau liant des centres de données à pleine capacité en tirant parti de la gestion centralisée SDN. Cette réalisation a démontré que le paradigme SDN n'est pas seulement une mode et peut sans aucun doute révolutionner la façon dont les réseaux sont construits et gérés.

Virtualisation réseau. Les centres de données offrent une infrastructure rentable pour héberger des services ou applications à grande échelle (par exemple, le streaming vidéo). Aujourd'hui,

de grandes entreprises comme Amazon, Google, Facebook et Yahoo! utilisent régulièrement des centres de données pour le stockage, la recherche sur le Web et les calculs à grande échelle. Malgré leur importance, les architectures des centres de données sont encore loin d'être idéales. Il y a une dizaine d'années, les centres de données utilisaient des serveurs dédiés pour exécuter des applications, ce qui entraînait une mauvaise utilisation des serveurs, un faible support de QoS, un manque de flexibilité pour leur gestion et un coût opérationnel élevé.

La situation s'est améliorée avec l'émergence de la virtualisation des serveurs, qui permet à plusieurs machines virtuelles (ou VM en bref pour Virtual Machines) d'être co-localisées sur une seule machine physique. La virtualisation des serveurs permet aux fournisseurs de services en nuage de mutualiser les installations informatiques entre leurs clients tout en offrant la séparation nécessaire entre les serveurs gérés pour des utilisateurs différents. Les infrastructures de réseaux actuelles offrent moins de flexibilité et les solutions actuelles pour offrir une isolation entre les clients reposent sur des solutions lourdes de *tunneling*. La virtualisation réseau constitue une solution prometteuse pour offrir plus de souplesse aux fournisseurs de services cloud.

NFV. NFV est un concept d'architecture réseau prometteur pour réduire les coûts d'exploitation. Le déploiement de services réseau nécessite l'utilisation de différentes fonctions réseau telles que pare-feux (firewall), inspection de paquets (ou DPI en bref pour Deep Packet Inspection) ou optimisation vidéo. Dans les réseaux existants, ces fonctions sont exécutées par des dispositifs matériels spécialisés appelés *middleboxes*. Ces dispositifs sont coûteux et difficiles à migrer sur l'ensemble du réseau, ce qui limite la capacité des opérateurs de réseau à s'adapter au trafic et à concevoir de nouveaux services.

La virtualisation des fonctions réseau, poussée par l'initiative NFV [148], apporte une flexibilité dans leur gestion. Les fonctions peuvent maintenant être exécutées dans des machines virtuelles (VMs), sur du matériel générique. Elles peuvent donc facilement être déplacées sur le réseau. Un serveur peut exécuter une fonction pare-feu à un moment donné, et le même serveur peut exécuter une fonction d'optimisation vidéo l'heure suivante.

Comme les fonctions nécessitent beaucoup de ressources virtuelles différentes (bande passante, cœurs CPU, mémoire), la gestion et l'orchestration des fonctions virtuelles (NFV Management & Orchestration ou MANO) est au cœur du paradigme. Il comprend le provisionnement des fonctions de réseau virtuel (VNF), ainsi que la gestion de la structure sous-jacente (réseau et serveurs). Un orchestrateur est chargé d'assigner des emplacements à la fonction et d'allouer les ressources nécessaires pour satisfaire les demandes dans le réseau.

La virtualisation réseau permet une baisse significative des coûts matériels, une meilleure utilisation des ressources réseaux et l'introduction de nouveaux services.

Contexte actuel. Ces technologies sont en train de pénétrer rapidement le marché, en raison de l'importante réduction potentielle de coûts qui sera réalisé avec ces nouvelles technologies. Ces réductions de coûts sont multiples. Tout d'abord, en matériel, grâce à la possibilité d'utiliser des équipements génériques en lieu et place d'équipements spécialisés qui sont nettement plus chers. Il s'agit ici de la même évolution qui s'est produite dans les centres de données avec l'utilisation de machines de base moins puissantes, mais en plus grand nombre. Ensuite, ces technologies, en permettant d'utiliser des équipements génériques, vont réduire la dépendance des opérateurs vis-à vis des constructeurs de matériel réseaux. Enfin, on assistera à une réduction des frais opérationnels. En effet, comme déjà discuté, le contrôle des réseaux sera beaucoup plus facile, car ils pourront être programmés à distance. Cela évitera de longues et difficiles configurations manuelles. De plus, cela permettra une introduction rapide de nouvelles technologies qui pourront ensuite être monétisées.

Défis. Ces nouveaux paradigmes apportent des solutions à des problèmes existants, mais aussi de nouveaux défis à relever. En effet, l'utilisation d'un plan de contrôle centralisé pose des problèmes de passage à l'échelle, de protection contre les pannes et de sécurité. Plusieurs surveys sont maintenant disponibles sur ces paradigmes, voir, par exemple, [170, 144, 139] où leurs différents défis sont discutés.

- **Gestion dynamique du réseau.** L'utilisation d'un plan de contrôle centralisé utilisant des données de métrologie mises à jour régulièrement permet de prendre des décisions dynamiques de routage, comme discuté pour la mise en place du routage efficace en énergie dans le chapitre 8. Cependant, cette gestion dynamique pourrait entrainer des conséquences négatives sur les réseaux : pertes de paquets, phénomènes de boucles lors de changements de routes, coexistence difficile avec les protocoles existants...

- **Complexité supérieure de certains protocoles et matériels SDN.** Dans OpenFlow 1.3, le routage peut être fait en utilisant jusqu'à 40 champs d'un paquet IP, comme source IP ou protocole. Cela permet de mettre en place des politiques de qualités de services ou de sécurité. Mais, la contrepartie est de devoir utiliser une mémoire plus complexe, la mémoire TCAM (pour Ternary Content-Addressable Memory) qui est plus coûteuse et de taille très limitée. Les tables de routage SDN comportent donc un nombre très limité de règles, de l'ordre de quelques milliers. Il devient donc nécessaire de re-penser le routage dans ce contexte, voir par example [163, 217].

- **Point de contrôle unique.** Le contrôleur devient à la fois un point de panne unique et un goulot d'étranglement potentiel que ce soit à cause de son CPU ou de sa bande passante limitée. En effet, le contrôleur doit recevoir des mises à jours récurrentes des routeurs pour maintenir une vue cohérente du réseau. Le contrôleur peut donc se retrouver en sur-charge ou tout simplement avoir un délai trop long pour communiquer avec certains routeurs. Il est donc important de s'intéresser à la performance du contrôleur [236, 207]. Dans de grands réseaux, il peut devenir nécessaire d'avoir recours à plusieurs contrôleurs [193], ce qui pose des questions de cohérence de vue entre ceux-ci, de gestions des pannes de contrôleurs et de placement.

- **Choix de l'architecture.** L'architecture des réseaux logiciels est primordiale pour assurer leur bonne efficacité. Il est par exemple crucial que le ou les contrôleurs puissent communiquer rapidement avec les équipements qu'ils contrôlent. Le placement des contrôleurs a été par exemple étudié dans des travaux tels que [245, 171]. Ce placement doit prendre en compte la charge des contrôleurs, la distance aux routeurs et la tolérance aux pannes.

- **Placement des ressources virtuelles.** Avec l'utilisation des fonctions réseau virtuelles, le choix d'une route comprend simultanément le choix d'où ces fonctions seront exécutées. Cela pose un nouveau type de problème à résoudre. De plus, certains services réseaux sont composés de plusieurs fonctions réseaux et peuvent avoir une structure complexes.

  – Les chaînes de fonctions de services se décomposent en une séquence ordonnée de fonctions réseaux que les flots réseaux doivent traverser dans le bon ordre. Cela rajoute une dimension supplémentaire à ces problèmes de placement [168, 169].

  – Les slices (5G par exemple) sont de véritables réseaux virtuels dans lesquels chaque noeud est une fonction réseau. Le problème de placement revient alors à plonger ces réseaux virtuels dans le réseau physique. Dans la plupart des scénarios pratiques, des contraintes fortes de délai et de priorité des flots doivent en plus être gérées [119, 118].

Contribution et plan. Au cours de ces dernières années, j'ai apporté des solutions à certains de ces problèmes.

- Je me suis d'abord intéressé aux problèmes de routage dans des réseaux logiciels utilisant de la TCAM dans le chapitre 6. Dans ces réseaux, le routage peut se faire sur plusieurs champs, mais le nombre d'entrées dans les tables de routage est très limité. J'ai donc introduit des algorithmes de compression des tables de routage multi-dimensionnelles, en particulier des algorithmes d'approximation et des algorithmes FPT en section 6.3. J'ai ensuite testé en pratique ces algorithmes en utilisant une petite plateforme expérimentale qui nous a permis d'émuler le routage de demandes sur une petite architecture de centre de données en section 6.6.

- J'ai ensuite examiné des problèmes de placement de ressources virtuelles et en particulier de chaînes de fonctions de services dans le chapitre 7. Il est courant qu'un flot de données doive être traité par une ou plusieurs fonctions réseaux, et ce dans un ordre précis. Ces fonctions réseaux peuvent maintenant être virtualisées et instanciées à la demande à différents endroits du réseau. Le problème classique de routage devient alors un problème joint de routage et de placement de ressources avec des contraintes d'ordre. J'ai proposé dans la section 7.1 des modèles de décomposition pour résoudre le problème et utilisé la génération de colonnes pour les résoudre tout en passant à l'échelle. J'ai aussi regardé un scénario dans lequel un opérateur réseau a déjà fixé ses routes, souvent selon des plus courts chemins. Dans la section 7.2, j'ai proposé des algorithmes d'approximation avec facteur logarithmique pour résoudre le problème de placement de chaines de fonctions de services tout en minimisant leur coût d'installation.

Ces travaux ont été fait en collaboration avec deux doctorants de COATI, N. Huin et A. Tomassilli, et les chercheurs suivants : F. Havet, *B. Jaumard*, *D. Lopez-Pacheco*, J. Moulierac, S. Pérennes, *M. Rifai*, *G. Urvoy-Keller*.

Ils ont donné lieu à 4 publications dans des journaux internationaux [J9, J11, J8, Ci30], 6 dans des conférences internationales [Ci42, Ci38, Ci36, Ci33, Ci30, Ci32] et 3 dans des conférences nationales [Cn72, Cn70, Cn69].

Chapter 6

Réseaux logiciels (SDN)

6.1 Introduction

Depuis quelques années, le paradigme SDN attire l'attention des chercheurs et des industriels. De plus en plus d'équipements supportant ce type de réseau programmable voient le jour. Contrairement aux réseaux classiques où le routage est calculé grâce à des protocoles distribués, SDN se base sur une séparation du plan de contrôle et du plan de données : un ou plusieurs contrôleurs sont placés sur le réseau et décident de l'acheminement des paquets. Ils transmettent alors aux commutateurs (qui sont de simples équipements de transmission de paquets, sans intelligence) les règles de "forwarding".

De plus, les règles de routage utilisées dans les réseaux SDN sont plus complexes que celles des réseaux classiques. Dans OpenFlow 1.3, la reconnaissance de paquet peut s'effectuer sur un maximum de 40 champs. La complexité de ces règles nécessite l'utilisation de Ternary Content Adressable Memory (TCAM) qui est malheureusement plus chère et plus gourmande que le type de mémoire utilisé sur les réseaux classiques. La taille des tables de routage pouvant être parcourues s'en retrouve alors grandement réduite (de l'ordre de 750 à 4000 règles [423]). Cette limitation est une contrainte importante pour le déploiement des réseaux logiciels.

Ce problème a été attaqué dans la littérature, comme discuté en section 6.2, en utilisant différentes stratégies, comme la compression de tables de routages [196, 173] ou la distribution des règles de routage [200].

Dans ce chapitre, nous examinons une méthodologie de compression dans laquelle n'importe quel champ de l'en-tête d'un paquet peut être compressé. C'est une avancée importante car cela permet une compression plus efficace des tables de routage et permet de mettre en œuvre des politiques de routage avancées, comme l'équilibrage de charge (load balancing) et/ou des politiques de qualité de service. Dans la suite, nous considérons la compression de règles pour deux champs, les sources et destinations. Mais, notre solution peut cependant être aussi appliquée sur d'autres champs tels que le ToS (Type de Service), le protocole de transport, ... et peut se généraliser à un nombre plus élevé de champs.

Nous étudions d'abord le problème théorique de compresser les tables de routage SDN. La nouveauté est que ces tables sont multi-dimensionelles et la difficulté vient du fait que l'ordre dans lequel les règles sont écrites devient crucial dans ce contexte. Nous avons d'abord déterminé la complexité du problème, répondant ainsi à une question ouverte de [367]. Nous avons ensuite proposé des algorithmes d'approximation, puis des algorithmes à complexité paramétrée fixe (FPT) pour résoudre le problème.

Nous présentons ensuite Minnie, une solution de routage SDN qui utilise la compression

de règles. Puisque le problème de compression de table est NP-Complet, MINNIE implémente une heuristique de compression qui crée trois tables compressées différentes, utilisant seulement l'agrégation par source pour la première, par destination pour la deuxième, ou la règle par défaut pour la dernière, et choisit la plus petite des trois tables. Cette heuristique est une 3-approximation [183]. Puis MINNIE utilise une heuristique de routage où la métrique des poids dépend de l'utilisation des liens et de la taille des tables.

En Section 6.6.2, nous présentons les résultats de MINNIE obtenus par simulation sur plusieurs *topologies de centres de données* parmi les plus communes. Nous montrons que notre solution passe à l'échelle et peut *gérer plus d'un million de flots différents avec moins de 1000 entrées* dans sa table de routage et avec un temps de compression négligeable.

Enfin, nous avons utilisé une plateforme matérielle pour tester à la fois nos algorithmes de compression et la possibilité pour les résaux SDN d'implémenter efficacement des algorithmes d'optimisation. Cette plateforme nous a permis d'émuler un 4-fat tree, l'une des architectures de centre de données les plus courantes et de tester la mise en pratique des technologies logicielles. En effet, le fait de centraliser les décisions de routage en plaçant un algorithme d'optimisation dans le contrôleur pourrait apporter divers problèmes : augmentation du délai en raison de potentiels contacts avec le contrôleur (en particulier pour le premier paquet d'un flot), saturation du CPU ou des liens avec le contrôleur, augmentation du taux de perte, ... Nos résultats montrent que notre solution est capable de minimiser le nombre d'entrées dans les commutateurs, *tout en gérant avec succès la dynamique des requêtes et en maintenant la stabilité des réseaux.*

6.2 État de l'art

Pour prendre en charge une vaste gamme d'applications réseau, SDN a été conçu pour appliquer des règles basées sur les flots, qui sont plus complexes que les règles basées sur les destinations utilisées dans les routeurs IP traditionnels. La complexité des nouvelles règles de routage est bien supportée par les mémoires TCAM, mais celles-ci sont de taille très limitées.

De nombreuses études existantes dans la littérature ont abordé ce problème d'espace de règles limité. Par exemple, les auteurs de [215] et [205] essaient de compacter les règles en réduisant le nombre de bits décrivant un flot dans le commutateur en insérant une petite étiquette dans l'en-tête du paquet. Cette solution est complémentaire de la nôtre, mais elle nécessite une modification (*i*) des en-têtes de paquets et (*ii*) de la façon dont les tables SDN sont remplies. De même, il est difficile d'ajouter un identificateur à chaque paquet entrant dans les ASIC (Application-Specific Integrated Circuits), car il ne s'agit pas d'une opération standard, ce qui entraîne le traitement des paquets par le processeur central du routeur, pénalisant fortement la performance et le débit. Une autre approche consiste à compresser les règles sur un seul commutateur. Par exemple, les auteurs de [356, 282, 240] ont proposé des algorithmes pour réduire le nombre de règles requises pour réaliser des politiques sur un seul commutateur.

Plusieurs travaux ont proposé des solutions pour distribuer des politiques de routage tout en gérant les contraintes d'espace de règles à chaque commutateur [200, 216, 217, 163, 142]. Cependant, aucun mécanisme de compression n'est ajouté à ces solutions. Par exemple, dans [163], les auteurs proposent OFFICER. Cette solution crée un chemin par défaut pour toutes les communications, et ensuite, certaines déviations sont introduites à partir de ce chemin en utilisant différentes politiques pour atteindre la destination. Selon les auteurs, la stratégie Edge First (EF), où cette déviation est effectuée pour minimiser le nombre de sauts (hops) entre le chemin par défaut et le chemin cible, offre le meilleur compromis entre la qualité de service requise (QoS) et la taille de la table de routage. Notez cependant que l'application de cet algorithme pourrait

pénaliser inutilement la QoS des flots pour les scénarios dans lesquels les tables de routage des routeurs sont rarement pleines. Dans [142], les auteurs proposent CacheFlow qui introduit un module CacheMaster et une section partagée de commutateurs logiciels par TCAM (disponible uniquement dans les commutateurs matériels). CacheMaster construit l'arbre de dépendances des règles à installer et distribue ensuite les règles entre la TCAM et les commutateurs logiciels, en plaçant les règles les plus populaires dans le commutateur matériel, permettant ainsi un transfert rapide pour le plus grand volume possible de trafic. Lorsqu'un paquet a besoin d'une règle de transfert non disponible dans la TCAM, un tel paquet est transmis aux commutateurs logiciels, qui renvoient le paquet au commutateur matériel dans un port d'entrée prédéterminé, pour être enfin renvoyé à un port de sortie spécifique. Si les commutateurs logiciels n'ont pas de règles de correspondance, le contrôleur SDN est appelé. Les faiblesses de CacheFlow reposent sur son architecture, car cette solution nécessite l'installation d'un commutateur logiciel pour chaque commutateur matériel, ce qui peut nécessiter une réorganisation du câblage réseau et des ressources supplémentaires pour héberger les commutateurs logiciels. Deuxièmement, le nombre optimal de commutateurs logiciels nécessaires peut être difficile à déterminer, du fait que pour des raisons de performance, les commutateurs logiciels ne doivent conserver que des règles de routage (dont le nombre dépend des caractéristiques de trafic) dans l'espace mémoire du noyau. Enfin, l'architecture à deux couches de CacheFlow (c'est-à-dire commutateur logiciel au-dessus d'un commutateur matériel) augmente le délai pour contacter le contrôleur et installer les règles manquantes.

À notre connaissance, les travaux les plus proches du notre sont [173, 203, 196]. Dans [173], les auteurs introduisent XPath qui identifie les chemins de bout en bout en utilisant l'ID de chemin, puis compresse toutes les règles et pré-installe les règles nécessaires dans la TCAM. Nous comparons nos résultats avec ceux de XPath dans [J11]. Minnie utilise moins de règles même dans le cas d'un trafic global car XPath code les routes pour tous les plus courts chemins entre les sources et les destinations. Cela se fait au détriment de la redondance de chemins, qui est utile pour l'équilibrage de la charge et la tolérance aux pannes. Les opérateurs de réseau devraient envisager ce compromis lors du choix de la méthode à utiliser. Dans [203], les auteurs suggèrent la compression des règles SDN en suivant le concept de correspondance des préfixes les plus longs avec les priorités en utilisant l'heuristique Espresso [386] et montrent que leur algorithme conduit à 17 % d'économies seulement. Nous réussissons à atteindre de meilleurs ratios de compression en utilisant Minnie. Enfin, [196] résout le problème de la compression des tables de routage en utilisant la règle par défaut uniquement dans le cas du routage Energy-Aware. Nous étendons cette solution en considérant d'autres types de compression.

6.3 Compresser des tables de routages bi-dimensionnelles avec ordre

Ce travail est un le fruit d'une collaboration avec Frédéric Havet et Joanna Moulierac et a donné lieu aux publications [Ci42, J9, S29].

Dans ce chapitre, nous étudions une méthode pour compresser les tables de routage en utilisant des règles agrégées des formes suivantes : "(*,destination)→ port" ou "(source,*)→ port" ou aussi une entrée par défaut "(*,*)→ port".

Nous considérons alors le problème de trouver, en utilisant des règles d'agrégation, la liste de routage la plus courte qui émule une table de routage donnée. Nous étudions plusieurs variantes du problème formellement définies dans la section 6.3.1 : Liste de routage (étant donné un

entier, k, trouver une liste de routage de taille au plus k), Réduction de liste (réduire la taille de la liste de routage d'au moins k règles), avec un nombre fixe de ports k (k-Ports problèmes), utilisant ou non le triplet global (Problèmes sans règle globale).

Contributions. Après quelques préliminaires, nous montrons d'abord dans la section 6.4 que le problème Liste de routage 2-Ports est fortement lié au fameux problème Feedback Arc Set dans un digraphe biparti associé.

Cette relation nous permet de montrer que le problème Liste de routage 2-Ports est NP-complet via une réduction de Feedback Arc Set. Comme conséquence triviale, Liste de routage et Réduction de liste, ainsi que leurs variantes avec k-ports pour tous $k \geq 2$, sont NP-complets. Dans ce manuscrit, nous énonçons seulement les résultats en section 6.5. Toutes les preuves peuvent être trouvées dans [J9].

Cela répond à une question ouverte de [367]. Comme pour nous, les auteurs ont examiné le problème de la détermination d'une table de routage compacte utilisant des triplets d'agrégation qui a le même comportement que la table de routage d'origine. La différence avec notre problème est que leur but était de trouver ce qu'ils ont appelé une table de routage *sans conflit (conflict-free)* dans laquelle les triplets peuvent être pris dans n'importe quel ordre. Au contraire, comme indiqué ci-dessus, l'ordre est crucial dans nos problèmes. Nous citons: "Le problème de compression avec des filtres incohérents [triplets], où l'on utilise la priorité pour définir le meilleur filtre [triplets] correspondant, est ouvert, et nous conjecturons qu'il est NP-complet. "

En fait, nous prouvons un résultat plus fort en montrant que Liste de routage avec 2-ports et Liste de routage avec 2-ports sans règle globale sont NP-complets, même si restreint à un ensemble complet de communications. Un ensemble de communication est *complet* s'il est de la forme $S \times T$, et un ensemble de triplets de communications est *complet* s'il s'agit d'un ensemble de triplets de communication sur un ensemble complet de communications.

Dans la section 6.5.1, nous fournissons des algorithmes d'approximation. Nous décrivons d'abord quelques heuristiques et montrons que l'une d'entre elles, appelée heuristique-basée-sur-la-direction, est une $2-$approximation pour Réduction de liste sans règle globale et une autre, appelée heuristique globale, est une $3-$approximation pour Réduction de liste. Ces algorithmes peuvent être généralisés pour n'importe quel nombre de champs $f \geq 2$, conduisant respectivement à des f- et $(2^f - 1)$-approximations. Ensuite, nous nous concentrons sur le problème avec deux ports. Nous fournissons un algorithme d'approximation pour Liste de routage lorsque l'ensemble des triplets de communication est presque plein. Cet algorithme utilise un algorithme polynomial 4-approché pour Feedback Arc Set dans les tournois bipartis complets, publié par Van Zuylen [299].

Nous avons montré dans [Ci42, S29] que tous ces problèmes sont FPT avec le paramètre naturel. Toutes nos preuves consistent à montrer que le problème considéré a un noyau. Nous présentons dans la section 6.5.2 les résultats et preuves pour le problème Réduction de liste sans règle globale.

Travaux connexes. D'autres types de compression de tables de routage ont été considérés, par exemple de la compression avec perte [159]. L'idée est d'atteindre un taux de compression élevé au prix d'une perte d'information. Notez que, dans ce travail, nous considérons des règles d'agrégation qui correspondent à un joker sur un champ entier. La mémoire TCAM permet d'utiliser des caractères génériques plus expressifs à la granularité d'un seul bit, tels que les caractères génériques classiques utilisant des préfixes et ou des plages (ranges) d'adresses. Il existe une littérature abondante sur la classification des paquets, qui compare un en-tête de

paquet à un ensemble de règles, voir par exemple le survey [361]. La mémoire TCAM peut représenter efficacement les règles de préfixes, mais ne gère les règles de plages d'adresse qu'avec difficulté. Des travaux récents proposent de nouvelles méthodes pour gérer efficacement ces règles de plages [135, 141]. Notez que nos résultats de complexité (NP-complétude) s'étendent à des règles d'agrégation plus expressives (par exemple avec des préfixes ou des intervalles).

6.3.1 Modélisation du problème et préliminaires

Modélisation du problème

Modélisation. Une *communication* dans un réseau est une paire de nœuds (s, t). Le noeud s s'appelle la *source* et t la *destination*. Nous utilisons les champs source et destination dans nos exemples, bien que nos idées s'appliquent à n'importe quels paires de champs de préfixes IP. Un *routage* d'une communication (s, t) est un chemin dans le réseau de s à t. Un *routage* d'un ensemble de communications est l'union des routages de ses communications.

A chaque nœud, on associe un ensemble $\mathcal{X}$ de communications dont le chemin de routage passe par ce nœud. Le nœud doit connaitre pour chaque communication (s, t) dans $\mathcal{X}$, le port $\pi(s, t)$ de sortie que le chemin de routage de (s, t) utilise. Une façon simple de le faire est de stocker l'ensemble $\mathcal{C} = \mathcal{C}(\mathcal{X}, \pi)$ de tous les triplets $(s, t, \pi(s, t))$ pour $(s, t) \in \mathcal{X}$. De tels triplets sont appelés

- *triplets de communications*. On n'utilisera *cette terminologie* dans la suite pour *désigner des triplets sans règle* $*$. Ils sont souvent notés (s, t, p).

Cependant, une telle liste pourrait être très grande. Nous voulons donc la réduire autant que possible (surtout si l'on considère la limitation de mémoire TCAM dans SDN) en utilisant le symbole $*$. Par conséquent, nous pouvons également utiliser des triplets supplémentaires, appelés *triplets-*$*$. Il y a trois sortes de triplets-$*$:

- *t-triplet de destination* $(*, t, p)$, qui signifie que chaque communication qui a t comme destination a p comme port de sortie.
- *s-triplet de source* $(s, *, p)$, qui signifie que chaque communication qui a s comme source a p comme port de sortie.
- *triplet global* $(*, *, p)$, qui signifie que toutes les communications ont p comme port de sortie.

Une *liste de routage* $\mathcal{R}$ est une liste ordonnée $T_1, \ldots, T_r$ de triplets (triplets de communication ou triplets-$*$). Elle *route* la communication (s, t) au port $r(s, t) = p$ qui apparaît sur le premier triplet de la liste $\mathcal{R}$ qui est de la forme (s, t, p), $(*, t, p)$, $(s, *, p)$ ou $(*, *, p)$. Il est crucial de remarquer qu'*utiliser les triplets-$*$ rend l'ordre dans la liste de routage crucial*. Par exemple, dans les deux tables compactées de la table 6.1, si $(*, 4, \text{Port-4})$ apparaît en première position, alors $(1, 4)$ ne va pas être correctement routé par le Port-6 comme dans la liste de routage initiale, mais via le Port-4. Notez que le triplet global $(*, *, p)$ doit apparaître dans la dernière position, sinon toutes les communications sont routées vers p.

Si $r(s, t) = \pi(s, t)$, alors nous disons que (s, t) est *correctement routé* par $\mathcal{R}$. Si toutes les communications de $\mathcal{X}$ sont correctement routées, on dit que $\mathcal{R}$ *émule* $(\mathcal{X}, \pi)$.

$(\mathcal{X}, \pi) \mapsto \mathcal{C}(\mathcal{X}, \pi)$ est une bijection évidente entre toutes les paires d'un ensemble de communications et un port fonction et tous les ensembles de triplets de communications. Par conséquent,

(s,t)	$\pi(s,t)$
(0, 4)	Port-4
(0, 5)	Port-5
(0, 6)	Port-5
(1, 4)	Port-6
(1, 5)	Port-4
(1, 6)	Port-6
(2, 4)	Port-4
(2, 5)	Port-5
(2, 6)	Port-6

(s,t)	$\pi(s,t)$
(1, 4)	Port-6
(1, 5)	Port-4
(2, 5)	Port-5
$(*, 4)$	Port-4
$(0, *)$	Port-5
$(*, 6)$	Port-6

(s,t)	$\pi(s,t)$
(1, 5)	Port-4
(2, 6)	Port-6
$(1, *)$	Port-6
$(*, 4)$	Port-4
$(*, *)$	Port-5

Table 6.1: Un exemple avec une liste de routage (sur la gauche) et deux listes de routage de tailles minimum qui l'émulent, une (au milieu) sans le triplet global, et l'autre (à droite) avec celui-ci.

pour des raisons de commodité, nous considérons toujours l'ensemble de triplets de communications. En particulier, nous disons que $\mathcal{R}$ *émule* $\mathcal{C}$ s'il émule le $(\mathcal{X}, \pi)$ correspondant, c'est-à-dire si à chaque communication de $\mathcal{C}$ est assigné le même port par $\mathcal{C}$ et $\mathcal{R}$. Observez que $\mathcal{R}$ peut router plus de communications que $\mathcal{C}$. Par exemple, si le port de tous les triplets de $\mathcal{C}$ a la source s et le port p, alors la liste singleton faite du triplet global $(s, *, p)$ émule $\mathcal{C}$, même s'il n'y a pas de triplet dans $\mathcal{C}$ pour toutes les communications.

Problèmes. Le problème principal est alors de trouver la liste de routage la plus courte qui émule un ensemble donné de triplets de communications $\mathcal{C}$. Nous considérons d'abord les problèmes où nous ne sommes pas autorisés à utiliser le triplet global. La table 6.1 présente un exemple de liste de routage, avec des listes de routage de taille minimum l'émulant avec et sans le triplet global.

Nous notons $\text{rmin}(\mathcal{C})$ le nombre minimum de triplets dans une liste de routage qui émule $\mathcal{C}$ sans le triplet global.

Liste de routage sans règle globale :
Input: Un ensemble $\mathcal{C}$ de triplets de communications et un entier r.
Question: $\text{rmin}(\mathcal{C}) \leq r$?

Le *nombre de triplets sauvegardés* est $\text{sav}(\mathcal{C}) = |\mathcal{C}| - \text{rmin}(\mathcal{C})$. Le problème complémentaire de Liste de routage sans règle globale est le suivant.

Réduction de liste sans règle globale :
Input: Un ensemble $\mathcal{C}$ de triplets de communications et un entier z.
Question: $\text{sav}(\mathcal{C}) \geq z$?

Notons $\text{rmin}^*(\mathcal{C})$ le nombre minimum de triplets dans une liste de routage avec triplet global émulant $\mathcal{C}$ et $\text{sav}^*(\mathcal{C}) = |\mathcal{C}| - \text{rmin}^*(\mathcal{C})$. Nous étudions les variantes suivantes de Liste de routage sans règle globale et de Réduction de liste sans règle globale.

Liste de routage :
Input: Un ensemble $\mathcal{C}$ de triplets de communications et un entier r.
Question: $\text{rmin}^*(\mathcal{C}) \leq r$?

Réduction de liste :
Input: Un ensemble $\mathcal{C}$ de triplets de communications et un entier z.

Question: $\text{sav}^*(\mathcal{C}) \geq z$?

Notez que rmin et rmin* (resp. sav et sav*) sont des paramètres étroitement liés, dans le sens des deux lemmes suivants.

Lemme 20. *Soit $\mathcal{C}$ un ensemble de triplets de communications dans lequel n sources apparaissent.*

$$\text{rmin}^*(\mathcal{C}) \leq \text{rmin}(\mathcal{C}) \leq \text{rmin}^*(\mathcal{C}) + n - 1.$$

Démonstration. Clairement, $\text{rmin}^*(\mathcal{C}) \leq \text{rmin}(\mathcal{C})$.

Soit $\mathcal{R}$ le plus petit liste de routage avec des triplets globaux. Trivialement, $\mathcal{R}$ contient au plus un triplet global, parce que tous les triplets de communications sont routés par un triplet global. Soit $s_1, \ldots, s_n$ les sources apparaissant dans $\mathcal{C}$. Si $\mathcal{R}$ a un triplet global $(*, *, p)$, alors il peut être remplacé par n triplet de source $(s_i, *, p)$, $1 \leq i \leq n$, pour obtenir liste de routage émulant $\mathcal{C}$ sans triplet global s. Par conséquent, $\text{rmin}(\mathcal{C}) \leq \text{rmin}^*(\mathcal{C}) + n - 1$. □

Soit $M(\mathcal{C})$ le nombre maximum de triplets de $\mathcal{C}$ avec le même port.

Lemme 21. *Soit $\mathcal{C}$ un ensemble de triplets de communications.*

$$\text{sav}(\mathcal{C}) \leq \text{sav}^*(\mathcal{C}) \leq \text{sav}(\mathcal{C}) + M(\mathcal{C}) - 1.$$

Démonstration. Clairement, $\text{sav}(\mathcal{C}) \leq \text{sav}^*(\mathcal{C})$.

Prouvons maintenant que $\text{sav}^*(\mathcal{C}) \leq \text{sav}(\mathcal{C}) + M(\mathcal{C}) - 1$. Soit $\mathcal{R}$ le plus petit liste de routage avec des triplets globaux. Nous avons $|\mathcal{R}| = |\mathcal{C}| - \text{sav}^*(\mathcal{C})$. Trivialement, $\mathcal{R}$ contient au plus un triplet global.

Si $\mathcal{R}$ ne contient pas de triplet global, alors $\text{sav}(\mathcal{C}) \geq |\mathcal{C}| - |\mathcal{R}|$, et donc $\text{sav}^*(\mathcal{C}) \leq \text{sav}(\mathcal{C})$.

Supposons maintenant que $\mathcal{R}$ contienne un triplet global, disons τ. Soit $\mathcal{R}_\tau$ l'ensemble des triplets de $\mathcal{C}$ qui sont routés par τ. Soit $\mathcal{R}'$ la liste obtenue de $\mathcal{R}$ en remplaçant τ par $\mathcal{R}_\tau$ (dans n'importe quel ordre). Clairement, $\mathcal{R}'$ émule $\mathcal{C}$ et n'a pas de triplet global. Par conséquent

$$\text{sav}(\mathcal{C}) \geq |\mathcal{C}| - |\mathcal{R}'| = |\mathcal{C}| - (|\mathcal{R}| + |\mathcal{R}_\tau| - 1) = \text{sav}^*(\mathcal{C}) - (|\mathcal{R}_\tau| - 1).$$

Mais, par définition, $|\mathcal{R}_\tau| \leq M(\mathcal{C})$. Ainsi $\text{sav}^*(\mathcal{C}) \leq \text{sav}(\mathcal{C}) + M(\mathcal{C}) - 1$. □

En pratique, le nombre de ports d'un sommet est fixé. Il est donc naturel de se poser des questions sur la complexité de Liste de routage et de Réduction de liste lorsque le nombre de ports est limité par une constante. Nous appelons Liste de routage avec k-ports, (resp. Liste de routage avec k-ports sans règle globale, Réduction de liste avec k-ports, Réduction de liste avec k-ports sans règle globale), la restriction de Liste de routage, (resp. Liste de routage sans règle globale, Réduction de liste, Réduction de liste sans règle globale) pour les ensembles de triplets de communications avec au plus k ports.

Listes de routage standard et canoniques

Soit $\mathcal{R} = T_1, \ldots, T_r$ une liste de routage, éventuellement avec un triplet global. $\mathcal{R}$ est dit *standard* s'il existe i tel que T_j est un triplet-$*$ si et seulement si $j > i$. Le lemme suivant est laissé au lecteur.

Lemme 22. *Soit $\mathcal{C}$ un ensemble de triplets de communications et soit $\mathcal{R}$ une liste de routage émulant $\mathcal{C}$. Alors, la liste de routage $\mathcal{R}'$ obtenue à partir de $\mathcal{R}$ en*

- *supprimant les triplets de communications inutiles (ceux qui n'acheminent pas de triplets),*
- *mettant tous les triplets de communications de $\mathcal{R}$ au début et tous les triplets-$*$ de $\mathcal{R}$ à la fin, en gardant le même ordre que dans $\mathcal{R}$ pour les triplets-$*$,*

émule aussi $\mathcal{C}$ et est standard.

Une liste de routage est *canonique* si c'est la concaténation de sous-listes $\mathcal{B}_1, \ldots, \mathcal{B}_q$, appelées *blocks*, ayant les propriétés suivantes pour chaque $1 \leq \ell \leq q$:

(i) dans $\mathcal{B}_\ell$, il y a un unique triplet-$*$ et c'est le dernier;

(ii) si le triplet-$*$ de $\mathcal{B}_\ell$ est s-triplet de source (respectivement t-triplet de destination), alors tous les triplets de $\mathcal{B}_\ell$ ont la source s (respectivement destination t);

(iii) si $\ell \neq q$, alors $\mathcal{B}_\ell$ n'a pas de triplet global.

Lemme 23. *Soient $\mathcal{C}$ un ensemble de triplets de communications et $\mathcal{R}$ une liste de routage émulant $\mathcal{C}$. On peut obtenir à partir de $\mathcal{R}$ une unique liste de routage canonique, $\mathcal{R}'$, émulant $\mathcal{C}$ et pas plus longue que $\mathcal{R}$ par des applications successives des opérations suivantes:*

- *supprimer un triplet,*
- *remplacer triplet de communications (s,t,p) par le triplet $(*,t,p)$,*
- *réordonner les triplets.*

Démonstration. Nous allons utiliser trois opérations décrites dans l'instruction selon les règles suivantes sur un liste de routage $\mathcal{L} = T_1, \ldots, T_r$.

(R1) S'il y a des triplets après un triplet global, alors ils sont inutiles, donc nous les supprimons.

(R2) Si le dernier triplet est un triplet de communications (s,t,p), alors nous le remplaçons par $(*,t,p)$.

(R3) Soit $i_1, i_2, \ldots, i_q$ les indices dans l'ordre croissant des triplets-$*$ de $\mathcal{L}$. Si pour ℓ, T_{i_ℓ} est un s - triplet de source (respectivement t - triplet de destination) et il y a un triplet de communications T_i avec $i_{\ell-1} < i < i_\ell$ avec une source distincte de s (resp. destination distincte de t), alors on déplace T_i après T_{i_ℓ} .

Il est simple de vérifier que si l'une des règles ci-dessus s'applique à une liste de routage émulant $\mathcal{C}$, alors nous obtenons une nouvelle liste de routage émulant $\mathcal{C}$ qui n'est pas plus longue que l'originale.

Donc, à partir de $\mathcal{R}$, tant que l'une des règles (R1), (R2) ou (R3) s'applique, nous l'utilisons. Ce processus doit prendre fin, car chaque règle ne peut être appliquée qu'un nombre fini de fois. En effet (R1) diminue la taille de la liste, (R2) augmente le nombre de triplets-$*$, et (R3) diminue le vecteur $(i_1, i_2, \ldots, i_q)$ des indices de triplets-$*$ dans l'ordre lexicographique.

Soit $\mathcal{R}' = T'_1, \ldots, T'_r$ la liste de routage obtenue à la fin du processus. Notez qu'à chaque étape du processus, nous avons une liste de routage émulant $\mathcal{C}$ pas plus longue que $\mathcal{R}$ par la remarque ci-dessus. En particulier $\mathcal{R}'$ émule $\mathcal{C}$ et $|\mathcal{R}'| \leq |\mathcal{R}|$. De plus, aucune des règles (R1),

(R2) et (R3) ne s'applique. Soient $i_1, i_2, \ldots, i_q$ les indices dans l'ordre croissant des triplets-$*$ de $\mathcal{R}'$, et pour $1 \leq \ell \leq q$, soit $\mathcal{B}_\ell$ la sous-liste $T_{i_{\ell-1}+1}, \ldots, T_{i_\ell}$ (avec $i_0 = 0$). Par définition, l'unique triplet-$*$ de $\mathcal{B}_\ell$ est son dernier triplet. La concaténation $\mathcal{B}_1, \ldots, \mathcal{B}_q$ est $\mathcal{R}'$ car sinon (R2) s'appliquerait. Si $\ell \neq q$, alors $\mathcal{B}_\ell$ n'a pas de triplet global, sinon (R1) s'appliquerait. Enfin, si le triplet-$*$ de $\mathcal{B}_\ell$ est s-triplet de source (respectivement t-triplet de destination), alors tous les triplets de $\mathcal{B}_\ell$ ont pour source s (respectivement pour destination t), sinon (R3) s'appliquerait. Par conséquent, $\mathcal{R}'$ est canonique. □

Feedback arc set

Un *feedback arc set* dans un digraphe D est un ensemble d'arcs F tel que $D \setminus F$ est acyclique. Étant donné un ordre $\sigma = v_1, \ldots, v_n$ des sommets de D, un arc $v_i v_j$ de D est *un arc de retour pour* σ si $i > j$. Un ensemble standard d'arcs de retour dans D est l'ensemble de tous les arcs de retour pour un σ donné. Chaque ensemble d'arcs de retour contient un ensemble d'arcs de retour standard. La taille minimale d'un ensemble d'arcs de retour de D est notée $\mathrm{fas}(D)$.

FEEDBACK ARC SET est le problème suivant :
Input: Un digraphe D et un entier k.
Question: D a-t-il un ensemble d'arcs de retour de cardinalité au plus k ?

Ce problème est connu pour être NP-complet : c'est l'un des 21 problèmes NP-complets de Karp [96]. Il est également connu pour être NP-complet pour de nombreuses classes articulières de digraphes, comme les digraphes bipartis. Le problème a aussi été prouvé APX-complet par Kann [394].

Un *tournoi biparti* est une orientation d'un graphe biparti complet. Un digraphe biparti est *équilibré* si les deux parties de sa bipartition sont de tailles égales. Guo et al. [349] ont prouvé que FEEDBACK ARC SET est NP-complet pour les tournois bipartis. Ce résultat peut facilement être étendu à un tournoi biparti équilibré.

Lemme 24. FEEDBACK ARC SET *sur les tournois bipartis équilibrés est NP-complet.*

Démonstration. Réduction de FEEDBACK ARC SET sur les tournois bipartis. Soit D un tournoi biparti avec bipartition (A, B). Sans perte de généralité, on peut supposer que $|A| \leq |B|$. Soit D' le tournoi biparti équilibré obtenu à partir de D en ajoutant $|B| - |A|$ sommets dominant B. Les sommets ajoutés sont des sources dans D' et ne sont donc pas dans des cycles dirigés. Par conséquent, $\mathrm{fas}(D) = \mathrm{fas}(D')$. □

6.4 Relation entre LISTE DE ROUTAGE 2-PORT et FEEDBACK ARC SET problem

Soit $\mathcal{C}$ un ensemble de triplets de communications, avec ensemble de sources $S = \{s_1, \ldots, s_n\}$, ensemble de destinations $T = \{t_1, \ldots, t_m\}$ et ensemble de ports $\{p_1, p_2\}$. Nous associons à $\mathcal{C}$ la matrice $n \times m$ $A = A_\mathcal{C}$ définie par $a_{i,j} = 1$ si $(s_i, t_j, p_1) \in \mathcal{C}$, $a_{i,j} = -1$ si $(s_i, t_j, p_2) \in \mathcal{C}$, et $a_{i,j} = 0$ sinon. Nous associons aussi à $\mathcal{C}$ le graphe biparti $D_\mathcal{C}$ avec ensemble de sommets $S \cup T$ dans lequel pour tous $s \in S$ et tous $t \in T$, st est un arc si et seulement si $(s, t, p_1) \in \mathcal{C}$, et ts est un arc si et seulement si $(s, t, p_2) \in \mathcal{C}$. Ainsi, $A_\mathcal{C}$ est la matrice de bi-adjacence du graphe biparti $D_\mathcal{C}$.

Observez que $\mathcal{C} \to A_\mathcal{C}$ est une correspondance un-à-un entre les ensembles de triplets de communications, avec ensemble de sources $S = \{s_1, \ldots, s_n\}$, ensemble de destinations $T =$

$\{t_1, \ldots, t_m\}$ et ensemble de ports $\{p_1, p_2\}$, et les matrices-$n \times m$ avec entrées $\{-1, 0, 1\}$. Similairement, $\mathcal{C} \to D_{\mathcal{C}}$ est une correspondance un-à-un entre les ensembles de triplets de communications, avec ensemble de sources $S = \{s_1, \ldots, s_n\}$, ensemble de destinations $T = \{t_1, \ldots, t_m\}$ et ensemble de ports $\{p_1, p_2\}$ et les graphes orientés bipartis avec labels (un graphe orienté est un digraphe sans cycle de longueur 2) avec ensemble de sommets $S \cup T$.

Faisons des observations simples sur $\mathcal{D}_{\mathcal{C}}$. Dans un digraphe D, pour chaque sommet v, on note $A_D^+(v)$, ou simplement $A^+(v)$ quand D est clairement donné par le contexte, l'ensemble des arcs partants de v. Similairement, nous notons $A_D^-(v)$, ou simplement $A^-(v)$, l'ensemble des arcs entrants de v.

Fait 25. *1. Les triplets de communications de $\mathcal{C}$ sont en correspondance un-à-un avec les arcs de $\mathcal{D}_{\mathcal{C}}$.*

2. Si $s \in S$, alors $A^+(s)$ correspond à l'ensemble de triplets de communications avec source s et port p_1 et $A^-(s)$ correspond à l'ensemble de triplets de communications avec source s et port p_2.

3. Si $t \in T$, alors $A^-(t)$ correspond à l'ensemble triplets de communications avec destination t et port p_1 et $A^+(s)$ correspond l'ensemble de triplets de communications avec destination t et port p_2.

À la lumière du fait 25.1, pour des raisons de clarté, nous identifions souvent les arcs de $\mathcal{D}_{\mathcal{C}}$ avec leurs triplets de communications correspondants.

Lemme 26. *Soit $\mathcal{C}$ un ensemble de triplets de communications avec deux ports et soit $\mathcal{R}$ une liste de routage émulant $\mathcal{C}$, possiblement avec un triplet global. L'ensemble des triplets de communications dans $\mathcal{R}$ correspond à un feedback arc set de $D_{\mathcal{C}}$.*

Le lemme suivant est une sorte de réciproque du lemme 26.

Lemme 27. *Soit $\mathcal{C}$ un ensemble de triplets de communications avec n sources et m destinations. Si $D_{\mathcal{C}}$ est acyclique, alors il existe une liste de routage émulant $\mathcal{C}$ contenant au plus $n + m - 1$ triplets de source ou triplets de destination et aucun autre triplet.*

Les deux lemmes suivants impliquent que $\mathrm{rmin}^*(\mathcal{C})$ et $\mathrm{rmin}(\mathcal{C})$ sont fortement liés à $\mathrm{fas}(D_{\mathcal{C}})$.

Corollaire 28. *Si $\mathcal{C}$ est un ensemble de triplets de communications avec n sources et m destinations, alors* $\mathrm{fas}(D_{\mathcal{C}}) + 1 \leq \mathrm{rmin}^*(\mathcal{C}) \leq \mathrm{rmin}(\mathcal{C}) \leq \mathrm{fas}(D_{\mathcal{C}}) + n + m - 1$.

6.5 Résultats de Complexité

En utilisant la relation de Liste de routage avec Feedback Arc Set Problem, les théorèmes suivants sont démontrés dans [J9].

Théorème 29. Liste de routage avec 2-ports sans règle globale *est NP-complet.*

Théorème 30. Liste de routageavec 2-ports *est NP-complet.*

6.5.1 Algorithmes d'approximation

Heuristiques

Soit $\mathcal{C}$ un ensemble de triplets de communications avec l'ensemble source S et l'ensemble de destination T. Soient $n = |S|$ et $m = |T|$. Pour tout port p, soit $\mathcal{C}(p)$ l'ensemble des triplets de $\mathcal{C}$ avec le port p. Pour une source s (resp. destination t) et un port p, soit $\mathcal{C}(s,p)$ (resp. $\mathcal{C}(t,p)$) l'ensemble des triplets de $\mathcal{C}$ avec la source s (respectivement la destination t) et le port p. Pour toute source s, soit $M(s) := \max_p |\mathcal{C}(s,p)|$ le nombre maximum de triplets dans $\mathcal{C}$ avec la source s et même port, et pour chaque destination t, soit $M(t) := \max_p |\mathcal{C}(t,p)|$ le nombre maximum de triplets dans $\mathcal{C}$ avec destination t et même port. On définit :

$$\begin{aligned} Z^-(\mathcal{C}) &= \textstyle\sum_{s\in S}(M(s)-1) = \sum_{s\in S} M(s) - n \text{ et} \\ Z^|(\mathcal{C}) &= \textstyle\sum_{t\in T}(M(t)-1) = \sum_{t\in T} M(t) - m. \end{aligned}$$

Tout liste de routage sans triplet global émulant $\mathcal{C}$ donne une limite supérieure sur $\mathrm{rmin}(\mathcal{C})$. Une telle liste peut être obtenue en compressant source par source. Une source s après l'autre émule tous les triplets de $\mathcal{C}$ avec la source s. Cela peut être fait en utilisant le triplet $(s,*,p)$ pour p un port tel qu'il y a $M(s)$ triplets avec la source s et le port p après tous les triplets avec la source s et port distinct de p. Ce faisant, nous économisons $M(s)-1$ triplets en émulant les triplets avec la source s. Par conséquent, nous obtenons un liste de routage de taille $|\mathcal{C}| - Z^-(\mathcal{C})$. Une telle liste s'appelle *liste de routage basée sur la source*.

En procédant de même en fonction des destinations, on obtient une liste de routage, appelée *basée sur la destination* de taille $|\mathcal{C}| - Z^|(\mathcal{C})$.

En notant $Z(\mathcal{C}) = \max\{Z^-(\mathcal{C}), Z^|(\mathcal{C})\}$, nous avons

$$\mathrm{sav}(\mathcal{C}) \geq Z(\mathcal{C}) \text{ et } \mathrm{rmin}(\mathcal{C}) \leq |\mathcal{C}| - Z(\mathcal{C}). \tag{6.1}$$

L'algorithme consistant à calculer une liste de routage basée sur la source et une liste de routage basée sur la destination et à prendre la plus courte des deux, est appelé l'heuristique *basée sur la direction*. Elle fournit une liste de routage émulant $\mathcal{C}$ de taille $|\mathcal{C}| - Z(\mathcal{C})$. Comme nous le verrons dans le corollaire 34, l'heuristique basée sur la direction est une 2-approximation pour Réduction de liste sans règle globale.

De même, toute liste de routage émulant $\mathcal{C}$, éventuellement avec un triplet global, donne une borne supérieure sur $\mathrm{rmin}^*(\mathcal{C})$. On peut facilement obtenir une telle liste comme suit. Nous trouvons un port p tel que $|\mathcal{C}(p)|$ est maximum, et nous remplaçons les éléments de $C(p)$ par le triplet global $(*,*,p)$. Une telle liste est appelée *liste de routage avec le port par défaut*.

L'algorithme consistant à calculer une liste de routage basée sur la source, une liste de routage basée sur la destination, et une liste de routage avec le port par défaut et en prenant le plus court des trois, est appelé l'*heuristique globale*. Comme nous le verrons dans le corollaire 36, l'heuristique globale est une 3-approximation pour Réduction de liste.

Relation entre les problèmes avec et sans triplet global

Nous avons les résultats directs suivants :

Lemme 31. *Si* rmin^* *peut être approché avec un ratio* α *en temps polynomial, alors* rmin *peut être approché avec un ratio* $\alpha + 1$ *en temps polynomial.*

Lemme 32. *Si* sav *peut être approché avec un ratio* α *en temps polynomial, alors* sav^* *peut être approché avec un ratio* $\alpha + 1$ *en temps polynomial.*

Bornes supérieures à facteur constant d'approximation pour sav **et** sav*

Théorème 33. *Soit $\mathcal{C}$ un ensemble de triplets de communications. Alors,*

$$Z(\mathcal{C}) \leq \text{sav}(\mathcal{C}) \leq 2Z(\mathcal{C}).$$

Démonstration. En raison de l'équation (6.1), nous avons $Z(\mathcal{C}) \leq \text{sav}(\mathcal{C})$.

Prouvons maintenant que $\text{sav}(\mathcal{C}) \leq 2Z(\mathcal{C})$. Soit $\mathcal{R}$ la plus courte liste de routage sans triplet global émulant $\mathcal{C}$. Soit S' (resp. T') l'ensemble des sources (resp. destinations) x tel que $\mathcal{R}$ contient un $*$-triplet avec source (resp. destination) x. Pour chaque $x \in S' \cup T'$, soit $P(x)$ le nombre de triplets de $\mathcal{C}$ qui sont routés par τ^*_x, le $*$-triplet avec source ou destination x. Clairement, $\text{sav}(\mathcal{C}) = |\mathcal{C}| - |\mathcal{R}| = \sum_{x \in S' \cup T'}(P(x) - 1)$. De plus $P(x) \leq M(x)$ pour chaque x car tous les triplets de communications routés par τ^*_x ont le même port (celui de τ^*_x). Ainsi

$$\begin{aligned} \text{sav}(\mathcal{C}) &\leq \sum_{x \in S' \cup T'} (P(x) - 1) \leq \sum_{x \in S' \cup T'} (M(x) - 1) \\ &\leq \sum_{s \in S} (M(s) - 1) + \sum_{t \in T} (M(t) - 1) \\ &\leq Z^-(\mathcal{C}) + Z^|(\mathcal{C}) \leq 2Z(\mathcal{C}) \end{aligned} \tag{6.2}$$

□

Les listes de routage basées sur la source et sur la destination peuvent évidemment être calculées en temps polynomial.

Corollaire 34. *L'heuristique-basée-sur-la-direction est une 2-approximation pour le problème* Réduction de liste sans règle globale.

Le ratio 2 est le meilleur possible pour l'heuristique-basée-sur-la-direction. En effet,

Proposition 35. *Il existe des triplets de communications $\mathcal{C}$ pour lesquels la liste renvoyée par l'heuristique-basée-sur-la-direction a taille $|\mathcal{C}| - \frac{1}{2}\text{sav}(\mathcal{C})$.*

Démonstration. Considérons l'ensemble de triplets de communications suivant :

$$\mathcal{C} = \{(s_1, t_j, p_1) \mid 2 \leq j \leq \ell + 2\} \cup \{(s_i, t_1, p_2) \mid 2 \leq i \leq \ell + 2\}.$$

On peut voir facilement que $Z^-(\mathcal{C}) = Z^|(\mathcal{C}) = Z(\mathcal{C}) = \ell$ alors que $\text{sav}(\mathcal{C}) = 2\ell$. □

Dans l'équation (6.2), nous utilisons l'inégalité $Z^-(\mathcal{C}) + Z^|(\mathcal{C}) \leq 2Z(\mathcal{C})$ pour prouver que l'heuristique-basée-sur-la-direction est une 2-approximation. Mais, si nous n'utilisons pas cette inégalité, nous obtenons que l'heuristique est une $\frac{Z^-(\mathcal{C})+Z^|(\mathcal{C})}{Z(\mathcal{C})}$-approximation. Observez que le ratio $\frac{Z^-(\mathcal{C})+Z^|(\mathcal{C})}{Z(\mathcal{C})}$ peut être calculé pendant l'exécution de l'heuristique-basée-sur-la-direction et qu'il est souvent plus petit que 2.

Avec le lemme 32, le corollaire 34 donne directement que

Corollaire 36. *L'heuristique globale est une 3-approximation pour* Réduction de liste.

Le ratio 3 est aussi le meilleur possible pour l'heuristique globale. En effet

Proposition 37. *Il existe des ensembles de triplets de communications $\mathcal{C}$ tels que la liste renvoyée par l'heuristique globale a taille $|\mathcal{C}| - \frac{1}{3}\text{sav}(\mathcal{C})$.*

Généralisation aux plus grandes dimensions. Les algorithmes d'approximation peuvent être généralisés pour des matrices de dimensions supérieures. Elles correspondraient à des routages basés sur plus que 2 champs (par exemple sur 3 champs, adresse source, adresse destination et protocole).

6.5.2 Algorithmes FPT

Nous avons montré dans [Ci42, S29] que tous ces problèmes sont FPT avec le paramètre naturel, à savoir la taille de la liste pour le problème LISTE DE ROUTAGE et le nombre de règles économisées, noté z, pour RÉDUCTION DE LISTE. Toutes nos preuves consistent à montrer que le problème considéré a un noyau. Nous présentons ici les résultats et preuves pour le problème RÉDUCTION DE LISTE SANS RÈGLE GLOBALE.

Théorème 38. *Pour tout $k \geq 1$,* RÉDUCTION DE LISTE AVEC k-PORTS SANS RÈGLE GLOBALE *paramétré par z admet un noyau linéaire, et donc est FPT.*

Démonstration. Nous décrivons un algorithme de nucléarisation (*kernelisation*) qui, étant donné une instance $(\mathcal{C}, z)$ de RÉDUCTION DE LISTE AVEC k-PORTS, renvoie soit 'Oui '(ou une petite instance oui, par exemple $(\emptyset, 0)$) seulement si $\text{sav}(\mathcal{C}) \geq z$, ou une instance $(\mathcal{C}', z)$ équivalent à $\mathcal{C}$ (c'est-à-dire telle que $\text{sav}(\mathcal{C}) \geq z$ si et seulement si $\text{sav}(\mathcal{C}') \geq z$) de taille au plus $(4z-4)k$.

Un triplet $\tau = (s, t, p)$ de $\mathcal{C}$ est *lié à une source* (resp. *lié à une destination*) si $\mathcal{C}$ contient un autre triplet avec le port p et la source s (resp. la destination t). Il est dit *isolé*, s'il n'est ni lié à une source, ni lié à une destination. Soit τ un triplet isolé. Observez que quel que soit le triplet utilisé pour router τ, c'est le seul qui soit routé par ce triplet. D'où $\text{sav}(\mathcal{C}) = \text{sav}(\mathcal{C} \setminus \{\tau\})$.

Par conséquent, notre algorithme de nucléarisation supprime tous les triplets isolés de $\mathcal{C}$. Soit $\mathcal{C}'$ l'ensemble résultant de triplets de communications. Nous exécutons ensuite l'heuristique basée sur la direction sur $\mathcal{C}'$ pour calculer $Z(\mathcal{C}')$. Si $Z(\mathcal{C}') \geq z$, alors nous retournons 'Oui', parce que $\text{sav}(\mathcal{C}) \geq \text{sav}(\mathcal{C}') \geq Z(\mathcal{C}')$. Si non, alors nous retournons $\mathcal{C}'$.

Clairement, l'instance renvoyée est équivalente à $\mathcal{C}$. Montrons maintenant qu'elle a une taille au plus de $(4z-4)k$. Nous savons que $Z^-(\mathcal{C}') \leq z-1$. Soit $\mathcal{C}^-$ (resp. $\mathcal{C}^|$) l'ensemble des triplets de communications liés à la source (resp. liés à la destination) dans $\mathcal{C}'$. Puisque $\mathcal{C}'$ n'a pas de sommets isolés, nous avons $\mathcal{C}' = \mathcal{C}^- \cup \mathcal{C}^|$. Soit S' l'ensemble des sources s telles que $M(s) \geq 2$. Observez que

$$|S'| \leq \sum_{s \in S'} (M(s) - 1) = Z^-(\mathcal{C}') \leq z - 1 \tag{6.3}$$

et tous les triplets de communications liés à la source ont une source dans S'. Mais pour une source s, il y a au plus $kM(s)$ triplets avec la source s. D'où $|\mathcal{C}^-| \leq k \sum_{s \in S'} M(s)$. Maintenant $\sum_{s \in S'} M(s) = \sum_{s \in S'} (M(s) - 1) + |S'| \leq 2z - 2$ par l'équation (6.3). Donc $|\mathcal{C}^-| \leq (2z-2)k$. De même, $|\mathcal{C}^|| \leq (2z-2)k$, donc $|\mathcal{C}'| \leq (4z-4)k$. □

Théorème 39. RÉDUCTION DE LISTE SANS RÈGLE GLOBALE *paramétré par z admet un noyau quadratique, et donc est FPT.*

Démonstration. Nous décrivons un algorithme de nucléarisation qui, étant donné une instance $(\mathcal{C}, z)$ de RÉDUCTION DE LISTE, renvoie soit 'Oui' seulement si $\text{sav}(\mathcal{C}) \geq z$, ou une instance $(\mathcal{C}', z)$ équivalent à $\mathcal{C}$ de taille au plus $z(4z-4)$.

Nous supprimons d'abord tous les triplets isolés de $\mathcal{C}$, et dénotons l'ensemble résultant de triplets de communications $\mathcal{C}'$. Soit S' l'ensemble des sources s telles que $M(s) \geq 2$, et T' soit l'ensemble des destinations t telles que $M(t) \geq 2$. Si $|S'| \geq z$ ou $|T'| \geq z$, alors l'heuristique basée sur la direction sauve au moins z ports, et donc nous pouvons retourner 'Oui'.

On suppose donc maintenant que $|S'| \leq z-1$ et $|T'| \leq z-1$. Pour toute source s, soit $P(s)$ l'ensemble des ports qui n'apparaissent pas sur $\{s\} \times T'$. Observez que pour tout $p \in P(s)$, un triplet (s, t, p) est le seul avec le port p et la destination t. Par conséquent, nous sommes libre

de remplacer $(*, t, p)$ par (s, t, p), et nous pouvons supposer qu'une liste de routage plus courte émulant $\mathcal{C}'$ contient $(s, *, p)$ ou (s, t, p).

Supposons que $P(s) \neq \emptyset$. Soit $M_P(s)$ le nombre maximum de triplets avec la source s et le même port dans $P(s)$, et soit $p_1(s)$ un port dans $P(s)$ apparaissant dans $M_P(s)$ triplets avec la source s. Pour tout $p \in P(s)$, soit $\mathcal{C}'(p, s)$ l'ensemble des triplets de $\mathcal{C}'$ avec la source s et le port p . Soit $\mathcal{R}$ la plus courte liste de routage sans triplet global émulant $\mathcal{C}'$. Supposons que $(s, *, p) \in \mathcal{R}$ pour certains $p \in P(s) \setminus \{p_1(s)\}$. Alors $(s, *, p_1(s))$ n'est pas dans $\mathcal{R}$. Donc, nécessairement, les triplets de $\mathcal{C}'(p_1(s), s)$ précèdent $(s, *, p)$ dans $\mathcal{R}$. Par conséquent, en remplaçant $(s, *, p)$ par $(s, *, p_1(s))$ et les $M_P(s)$ triples de $\mathcal{C}'(p_1(s), s)$ par ceux de $\mathcal{C}'(p, s)$, on obtient une liste de routage $\mathcal{R}'$ qui émule $\mathcal{C}'$ de longueur pas plus longue que $\mathcal{R}$. Par minimalité de $\mathcal{R}$, $\mathcal{R}'$ est aussi une liste de routage plus courte émulant $\mathcal{C}'$. De plus, tous les triplets de $\mathcal{C}'(p, s)$ apparaissent dans $\mathcal{R}$. Par conséquent

$$\text{sav}(\mathcal{C}') = \text{sav}(\mathcal{C}' \setminus \mathcal{C}'(p, s)) \quad \text{pour tout } p \in P(s) \setminus \{p_1(s)\}. \tag{6.4}$$

De même, pour toute destination t, nous définissons $P(t)$ comme l'ensemble des ports qui n'apparaissent pas dans $S' \times \{t\}$. Soit $M_P(t)$ le nombre maximum de triplets avec la destination t et le même port dans $P(t)$, et soit $p_1(t)$ un port dans $P(t)$ apparaissant sur $M_P(t)$ triple avec la destination t. Enfin, pour tout $p \in P(t)$, soit $\mathcal{C}'(t, s)$ soit l'ensemble des triplets de $\mathcal{C}'$ avec destination t et port p. Un argument symétrique à celui ci-dessus montre

$$\text{sav}(\mathcal{C}') = \text{sav}(\mathcal{C}' \setminus \mathcal{C}'(p, t)) \text{ pour tout } p \in P(t) \setminus \{p_1(t)\}. \tag{6.5}$$

Soit $\mathcal{C}''$ l'ensemble des triplets de communications obtenus à partir de $\mathcal{C}'$ en supprimant $\mathcal{C}'(p, s)$ pour chaque source $s \in S'$ et chaque port $p \in P(s) \setminus \{p_1(s)\}$, et en supprimant $\mathcal{C}'(p, t)$ pour toute destination $t \in T'$ et tous les ports $p \in P(t) \setminus \{p_1(r)\}$. En appliquant plusieurs fois les équations 6.4 et 6.5, on obtient $\text{sav}(\mathcal{C}') = \text{sav}(\mathcal{C}'')$, c.-à-d. $\mathcal{C}''$ est équivalent à $\mathcal{C}'$ et donc à $\mathcal{C}$.

Par construction de $\mathcal{C}''$, pour tout $s \in S'$, il y a au plus z ports apparaissant dans les triplets avec la source s et au plus $z - 1$ dans les triplets avec destination dans T' et $p_1(s)$. Donc, il y a au plus $z \sum_{s \in S'} M(s)$ triplets avec la source dans S'. Comme $\sum_{s \in S'} M(s) \leq |S'| + \sum_{s \in S'} (M(s) - 1) \leq 2z - 2$, il y a au plus $z(2z - 2)$ triplets de $\mathcal{C}''$ avec source dans S'. De même, il y a au plus $z(2z - 2)$ triplets de $\mathcal{C}''$ avec destination dans T'. Ainsi, $|\mathcal{C}''| \leq z(4z - 4)$. Par conséquent, renvoyer $\mathcal{C}''$ donne le noyau désiré. □

6.5.3 Conclusion et problèmes ouverts

Complexité et algorithmes d'approximation. Dans ce travail, nous fournissons d'abord une étude de la complexité du problème de compression de tables de routage bidimensionnelles. Nous montrons que les problèmes de décision associés sont NP-complets dès que le nombre de ports est supérieur à 2.

Nous proposons ensuite des algorithmes d'approximation. En particulier, nous fournissons une heuristique simple, appelée heuristique-basée-sur-la-direction, qui est une 2-approximation pour Réduction de liste sans règle globale. Cependant, nous pensons que des heuristiques plus élaborées pourraient atteindre un meilleur ratio d'approximation que 2. Nous laissons donc ouvert le problème suivant.

Problem 40. Quel est le meilleur ratio d'approximation pour le problème Réduction de liste sans règle globale ?

Nous fournissons également une deuxième heuristique, l'heuristique globale, qui est une 3-approximation pour RÉDUCTION DE LISTE. Notez que cette heuristique a été testée pour des scénarios centres de données dans [J11]. De même, nous croyons que une 3-approximation n'est pas la meilleure possible pour RÉDUCTION DE LISTE.

Problem 41. Quel est le meilleur ratio d'approximation pour le problème RÉDUCTION DE LISTE ?

Nous avons établi un lien fort entre les problèmes considérés dans ce chapitre et le problème FEEDBACK ARC SET. Cela nous amène à la question suivante que nous laissons ouverte.

Problem 42. Peut-on déduire du fait que FEEDBACK ARC SET est APX-complet, que LISTE DE ROUTAGE 2-PORTS SANS RÈGLE GLOBALE est aussi APX-complet ? Pour quel ratio d'approximation ?

Enfin, nous avons réussi à généraliser la 3-approximation pour RÉDUCTION DE LISTE à un plus grand nombre de dimensions pour obtenir une $(2^f - 1)$-approximation pour les f champs. Pouvons-nous aussi généraliser la 4-approximation pour LISTE DE ROUTAGE [J9] à des dimensions plus élevées ?

Problem 43. Existe-t-il un algorithme d'approximation à ratio constant pour le problème LISTE DE ROUTAGE AVEC f-CHAMPS ?

Algorithmes FPT. Nous avons ensuite étudié le caractère FPT du problème de compression des tables de routage bidimensionnelles avec l'ordre. Nous montrons que les problèmes associés, RÉDUCTION DE LISTE et LISTE DE ROUTAGE, sont FPT pour leur paramètre naturel. Nous avons les résultats suivants [Ci42, S29]

Étant donné un ensemble de communications $\mathcal{X}$, le problème RÉDUCTION DE LISTE consiste à déterminer s'il y a une liste de routage émulant $\mathcal{X}$ de taille au plus $|\mathcal{X}| - z$. Considérant z comme paramètre, nous fournissons des noyaux linéaires pour RÉDUCTION DE LISTE AVEC k-PORTS SANS RÈGLE GLOBALE et RÉDUCTION DE LISTE AVEC k-PORTS, un noyau quadratique pour RÉDUCTION DE LISTE SANS RÈGLE GLOBALE, et un noyau de taille $z^3 + 2z^2 + z2^{2z}$ pour RÉDUCTION DE LISTE. Nous laissons ainsi ouverts les problèmes suivants.

Problem 44. Est-ce que RÉDUCTION DE LISTE SANS RÈGLE GLOBALE paramétré par z admet un noyau linéaire ?

Problem 45. Est-ce que RÉDUCTION DE LISTE paramétré par z admet un noyau polynomial ?

Le problème LISTE DE ROUTAGE SANS RÈGLE GLOBALE est de déterminer s'il existe une liste de routage émulant $\mathcal{X}$ de taille au plus r. Considérant r comme paramètre, nous montrons que LISTE DE ROUTAGE admet un noyau cubique et que LISTE DE ROUTAGE admet un $(2r^3 - 3r^2 + 6r - 1)$-noyau de Turing généralisé. Nous avons donc les problèmes ouverts suivants :

Problem 46. Est-ce que LISTE DE ROUTAGE SANS RÈGLE GLOBALE paramétré par r admet un noyau linéaire ou quadratique ?

Problem 47. Est-ce que LISTE DE ROUTAGE paramétré par r admet un noyau de Turing linéaire ou quadratique ?

6.6 MINNIE : enfin un monde SDN sans (trop de) règles

Nous proposons ici MINNIE, un algorithme de routage SDN se fondant sur des techniques de compression de règles pour réduire la taille des tables. Nous le validons par simulation pour différentes topologies de réseaux de centres de données, et par expérimentation sur une plateforme de type fat tree composée de 20 commutateurs. Côté simulation, nous montrons que MINNIE peut supporter aux alentours d'un million de flots lorsque la limite est de seulement 1000 règles par table, et avec des temps de calcul (routage et compression) négligeables. Côté expérimentation, nous montrons que, sans MINNIE, la limite de règles peut être rapidement atteinte avec un faible nombre de clients, ce qui accroît le délai sur le réseau. Avec MINNIE, le nombre de règles est réduit de manière importante sans introduire de perte de paquets ni de délai supplémentaire visible. Dans les deux cas, MINNIE affiche des taux de compression de tables entre 70 et plus de 95%.

Ce travail n'aurait pas pu voir le jour sans une collaboration avec *M. Rifai*, N. Huin, C. Caillouet, J. Moulierac, *D. Lopez Pacheco, G. Urvoy-Keller.* Il a donné lieu aux publications [Ci38, J11].

6.6.1 Modèle

Je résume ici la modélisation de la section précédente. Un réseau est représenté par un graphe dirigé $G = (V, A)$. Un sommet représente un commutateur dans le réseau et un arc représente un lien entre deux commutateurs. Chaque lien possède une capacité maximum et chaque commutateur possède une limite de nombre de règles. Pour un ensemble de demandes $\mathcal{D}$, un routage valide consiste à affecter un chemin à chaque demande en respectant les contraintes de capacité de liens et de tables.

Dans les tables de routage, une règle est représentée par un triplet (s, t, p) qui signifie que le flot entre s et t doit être dirigé sur le port p. Les règles d'agrégation permettent de regrouper les règles par destination (i.e. $(*, t, p)$), par source (i.e. $(s, *, p)$) ou les deux (i.e. $(*, *, p)$). Ces règles possèdent des priorités les unes par rapport aux autres. Si plusieurs règles correspondent à un flot, la règle avec la plus haute priorité est utilisée.

Nous présentons MINNIE qui trouve un routage pour chaque demande qui respecte les contraintes de capacités de liens et de tailles de tables de routage en utilisant la compression de table grâce aux règles d'agrégation. MINNIE comporte deux modules décrits ci-dessous. Pour une description plus détaillée de ces modules, nous invitons le lecteur à se référer à [J11].

Module de compression Puisque le problème de compression de règles est NP-Complet [183] et que le temps de compression est crucial pour ne pas impacter le réseau, nous utilisons l'heuristique de compression proposée dans la section 3.1.5, *heuristique globale*. Rappelons que cette heuristique est une 3-approximation du problème de compression et que son déroulé est le suivant. Étant donnée une table de routage, l'heuristique crée trois tables compressées différentes. La première table n'utilise que les règles d'agrégation par destination. Pour chaque destination t, le port p_t^* le plus présent est défini comme port par défaut de cette destination. La règle $(*, t, p_t^*)$ remplace alors toute les règles de la forme (s, t, p_t^*). Toutes les autres règles pour les flots à destination de t sont gardées, avec une priorité plus haute que la règle $(*, t, p_t^*)$. Une fois la compression effectuée pour toutes les destinations, le port p^* le plus présent parmi les règles d'agrégation devient le port par défaut de la table. Toutes les règles d'agrégation correspondantes sont remplacés par la règle par défaut $(*, *, p^*)$, qui a la plus faible priorité de

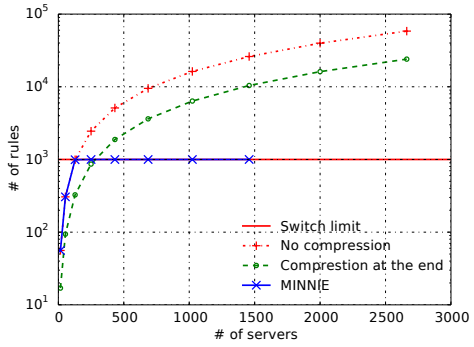


Figure 6.1: Simulation fat tree : nombre maximum de règles en fonction du nombre de serveurs.

toutes les règles. La deuxième table ne fait usage que des règles par source, de façon similaire au cas par destination. La dernière table n'utilise que la règle par défaut, dont le port correspond au port le plus présent dans la table, et qui remplace alors toutes les règles correspondantes. La plus petite des trois tables est la table utilisée.

Module de routage Pour assigner un chemin à chaque demande, nous calculons un plus court chemin où le poids du lien entre u et v dépends de l'utilisation du lien (u, v) ainsi que de l'utilisation de la table de routage de u. Cela permet de distribuer les flots sur la totalité de réseau et d'éviter les congestions de lien et de table. De plus, le poids dépendant des tables de routage est nul si aucun ajout de règle n'est nécessaire pour transmettre le flot par ce chemin. C'est le cas lorsqu'une règle d'agrégation correspondant au flot est déjà présente. En effet, un flot entre s et t passant sur le lien (u, v) ne nécessitera pas d'ajout dans la table si la règle d'agrégation $(s, *, v)$ est présente. Une fois les règles installées, les capacités des tables sont vérifiées et le module de compression est lancé sur celles qui ont atteint la limite.

6.6.2 Simulations

Nous nous intéressons au comportement de MINNIE sur quatre des architectures les plus fréquentes des centres de données, voir [311] : Fat tree, VL2, BCube et DCell. Ces topologies peuvent être séparées en 2 groupes. Dans le premier groupe, constitué de fat tree et de VL2, les serveurs agissent en tant que simples hôtes. Dans le second groupe (Dcell et BCube), les serveurs occupent aussi le rôle de commutateurs.

Dans la Figure 6.1, nous comparons le nombre maximal de règles installées sur un commutateur dans une topologie de fat tree entre trois scénarios. Dans le premier scénario, aucune compression n'est effectuée, seul le module de routage est utilisé. Dans le deuxième scénario, le module de compression est appelé lorsque tous les chemins ont été installés par chaque table. Le troisième scénario correspond à l'utilisation complète de MINNIE. Avec une limite de 1000 règles sur les commutateurs, jusqu'à 128 serveurs peuvent être déployés sans compression. En compressant à la fin, jusqu'à 256 serveurs peuvent être déployés. Enfin, MINNIE permet de déployer jusqu'à 1458 serveurs. Bien que la seule utilisation de la compression permette de déployer plus de 2 fois plus de serveurs, l'utilisation d'un algorithme de routage considérant aussi la capacité des tables permet le déploiement de 6 fois plus de serveurs.

Dans la Table 6.2, nous observons l'impact de l'utilisation de MINNIE sur plusieurs topologies. Bien que possédant un nombre équivalent de serveurs (environ 1000), ces topologies présentent

Topologie	# de serveurs	# de commutateurs	# de liens	Degré moyen	# de flots Total	# de flots par commutateurs Max	# de flots par commutateurs Moyen	# de regles avec comp. Total	# de regles avec comp. Max	# de regles avec comp. Moyen	Moyen Comp. Ratio	Temps de calcul em moyenne (ms) Chemin	Temps de calcul em moyenne (ms) Comp.
Group 1													
4-Fat tree (64)	1024	20	1056	54.4	917 504	454 244	216 268	8923	999	446	∼ 99.60	0.17	13
8-Fat tree (8)	1024	80	1280	19.2	1 015 808	649 044	61 030	25 853	999	323	∼ 99.61	0.21	7
16-Fat tree (1)	1024	320	3072	16	1 040 384	630 998	15 897	97 173	999	303	∼ 98.42	0.30	5
VL2(16, 16, 14)	896	88	384	16	790 272	261 266	42 906	59 237	1000	673	∼ 97.90	0.15	4
VL2(8, 8, 64)	1024	28	612	∼ 41.1	983 040	423 752	161 499	22 394	1000	799	∼ 99.45	0.19	11
VL2(16, 16, 16)	1024	88	1152	∼ 17.5	1 032 192	276 575	56 040	57 078	1000	648	∼ 98.39	0.18	4
Group 2													
Dcell(32, 1)	1056	33	1584	∼ 2.91	1 114 080	63 787	4893	123 655	1000	113	∼ 97.23	0.09	2
Dcell(5, 2)	930	186	1860	∼ 3.33	863 970	11 995	5716	717 018	994	642	∼ 87.84	0.19	2
BCube(32, 1)	1024	64	2048	∼ 3.77	1 047 552	37 738	3734	358 204	999	329	∼ 86.04	0.19	2
BCube(10, 2)	1000	300	3000	∼ 4.62	999 000	10 683	4153	849 316	998	653	∼ 80.85	0.25	2
BCube(6, 3)	1296	864	5184	4.8	1 678 320	7852	5184	1 795 400	991	831	∼ 83.18	0.49	4

Table 6.2: Résultats de MINNIE sur différentes topologies avec environ 1000 serveurs.

des différences importantes, que ce soit au niveau du nombre de commutateurs ou du nombre de liens. Cela entraîne des différences en terme de nombres moyen et maximum de flots par commutateur présentés dans la deuxième partie du tableau. Les topologies du deuxième groupe montrent un nombre de flots maximum et moyen plus faible que celles du groupe 1. Cette différence vient de la capacité des serveurs à effectuer de la transmission de paquets. Dans la troisième partie du tableau, nous voyons que MINNIE permet de router tous les flots du réseau sans dépasser la limite de 1000 règles sur aucun commutateur. Le taux de compression est d'au minimum 80% et monte jusqu'à plus de 99%. Finalement, le temps de calcul des chemins est en dessous de la milliseconde, et le temps de compression ne dépasse pas les 13 ms. L'impact de MINNIE est négligeable sur le délai du réseau, ce que nous confirmons dans la section suivante.

6.7 Résultats expérimentaux utilisant une plateforme SDN

Dans cette section, nous démontrons l'efficacité de MINNIE en utilisant une plateforme SDN. Les caractéristiques de notre réseau expérimental sont décrites dans la sous-section 6.7.1. Plus précisément, nous expliquons comment avec un commutateur matériel unique et des commutateurs OVS, nous déployons une topologie $k = 4$-Fat-Tree avec suffisamment de clients pour dépasser la taille de la table de routage du commutateur matériel, ainsi que le modèle de trafic qui répond aux besoins nos cas d'étude. Quelques détails sur l'implémentation de MINNIE dans un contrôleur Beacon sont également fournis. Les résultats obtenus sont présentés dans la section 6.7.2, où nous discutons de l'impact de MINNIE sur les délais des paquets réseau, le taux de perte et l'impact de l'utilisation de règles logicielles plutôt que de règles matérielles.

6.7.1 Paramètres d'expérimentation

La plateforme expérimentale est décrite ici. Pour que le contrôleur soit mis au courant de l'arrivée de nouveaux flots (et qu'ils ne soient pas routés automatiquement par une règle par défaut préexistante qui correspondrait), nous utilisons des commutateurs SDN virtuels (OVS) à l'entrée du réseau (appelés commutateurs de niveau 0). Ce point technique est discuté dans [J11].

Description de la plateforme

Notre plateforme de test est constituée par un commutateur HP 5400zl de technologie SDN (HP 5400zl SDN capable switch) possédant 4 modules, chacun avec 24 ports GigaEthernet et 4 serveurs DELL. Chaque serveur a 6 processeurs quad-core, 32 GB de RAM et 12 ports GigaEthernet. Sur chaque serveur, nous avons déployé 4 machines virtuelles (VM) avec chacune

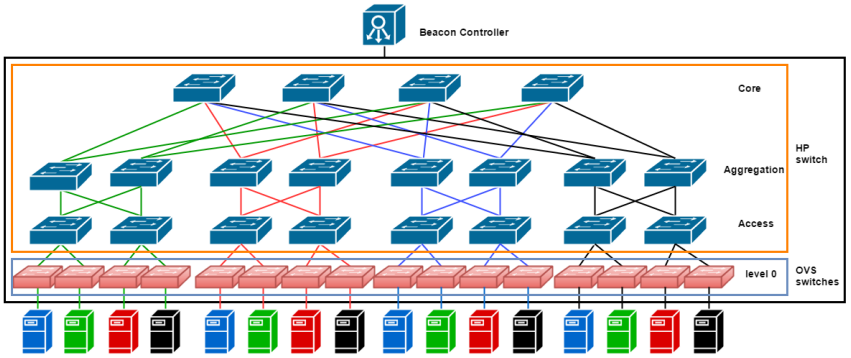


Figure 6.2: Notre architecture k=4 Fat-Tree avec 16 commutateurs OVS, 8 au niveau 1, 8 au niveau 2 et 4 au niveau 3.

8 interfaces réseau. Chaque VM est connectée à un commutateur Open vSwitch (OVS) dédié. Chaque commutateur OVS est ensuite connecté en utilisant un port physique (un des 12 ports du serveur) au commutateur HP.

La topologie de notre data center est un 4-Fat-Tree complet (voir la figure 6.2), qui est constitué de 20 commutateurs matériels SDN. Pour émuler ces 20 switches SDN, nous avons configuré 20 VLANs sur le switch physique (que nous désignons par Vswitches). Comme chaque VLAN possède une instance OpenFlow indépendante, chaque VLAN se comporte comme un commutateur SDN indépendant avec son propre ensemble isolé de ports et adresses MAC. La configuration en VLAN et l'isolation des ports qui en résulte empêchent le commutateur physique de router du trafic entre les VLANs à travers le backplane. Les Vswitches sont enfin connectés au commutateur HP en utilisant des cables Ethernet.

Chaque commutateur d'accès (Figure 6.2) connecte un seul sous-réseau IP avec 16 clients, ce dernier étant émulé par deux machines virtuelles, chacune pouvant comporter jusqu'à 8 ports Ethernet.

Le commutateur HP SDN peut prendre en charge un maximum de 65 536 règles (logicielles + matérielles) à partager parmi les 20 commutateurs SDN émulés. Les règles logicielles sont gérées dans la RAM et traitées par le CPU général (chemin lent) tandis que les règles matérielles sont stockées dans le TCAM (chemin rapide) du commutateur. Le nombre de règles matérielles pouvant être stockées par module dans notre commutateur étant égal à 750, la capacité totale du commutateur est égale à 3000 règles matérielles maximum. Ces entrées disponibles 65 536 (logiciel + matériel) ne sont pas distribuées équitablement entre les 20 commutateurs, car la politique utilisée est flot arrivé le premier, flot premier servi. Les règles SDN sont ainsi installées sur le commutateur HP dans l'ordre d'arrivée.

Dans l'un des serveurs physiques, nous avons également déployé une machine virtuelle supplémentaire hébergeant un contrôleur Beacon [223] pour gérer tous les commutateurs (HP Vswitches ou commutateurs OVS) dans le centre de données. Selon [209], Beacon offre de hautes performances en termes de débit et garantit un haut niveau de fiabilité et de sécurité. Pour éviter que le contrôleur ne devienne le goulot d'étranglement au cours de nos expériences, nous l'avons configuré avec 15 vCPU (c'est-à-dire 15 cœurs) et 16 Go de RAM.

6.7.2 Résultats expérimentaux

Nous évaluons maintenant MINNIE sur la plateforme SDN. Chaque serveur héberge 8 machines virtuelles et chaque machine communique avec toutes les autres machines qui ne sont pas

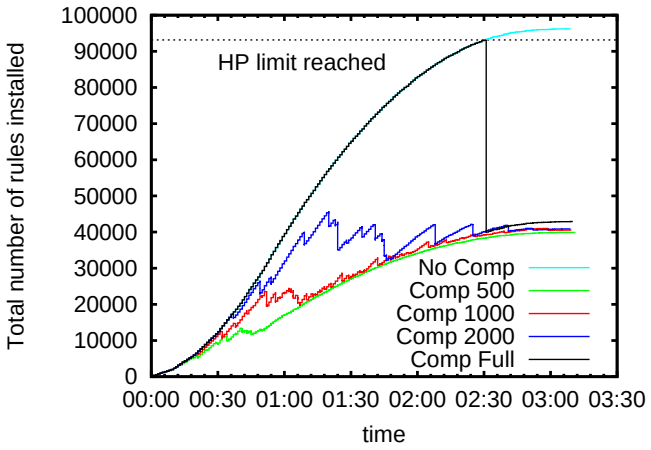


Figure 6.3: Plateforme fat tree : nombre total de règles installées au cours de l'expérimentation.

hébergées dans le même châssis. Sous ces hypothèses, 67 584 règles sont nécessaires au total afin de pouvoir router tous les flots, ce qui dépasse la limite totale.

Nous considérons trois scénarios distincts : (i) les tables ne sont jamais compressées, (ii) les tables sont individuellement compressées lorsqu'elles atteignent un certain seuil (500, 1000 ou 2000), et (iii) l'ensemble des tables est compressé lorsque la limite totale du nombre de règles est atteinte.

Nombre de compression et taux de perte. La Figure 6.3 décrit l'évolution du nombre total de règles installées au cours du temps. Notez que le nombre total de règles comprend aussi des règles installées sur des *Open VSwitch* nécessaire à la détection de nouveaux flots dans le réseau. Pendant la première demi-heure, le nombre de règles est similaire pour tous les scénarios. Chaque baisse du nombre de règles correspond à la compression d'une table. Les seuils à 500 et 1000 règles présentent la progression du nombre de règles la plus faible. En effet, en compressant les tables nous y introduisons des règles d'agrégation, ce qui diminue le nombre de règles à installer lors de l'apparition de nouveaux flots car ceux-ci peuvent appliquer une règle agrégée pré-existante. MINNIE permet ainsi de router tous les flots sans dépasser la limite des 65 536 règles, en offrant des taux de compression compris entre 76% et 97%.

En utilisant un seuil de règles trop bas, le nombre de compressions peut être important, ce qui dégrade le taux de perte. En effet, avec un seuil de 500 règles, plus de 16 000 compressions sont effectuées sur la totalité de l'expérience (voir Table 6.3). Comme il est impossible de redescendre en dessous du seuil de 500 règles dans la table, à chaque nouvelle règle le module de compression est lancé, ce qui fait monter le taux de perte des paquets à 0.003%. Pour les autres scénarios, le taux de perte est peu affecté par la compression.

Temps de compression. La figure 6.4 montre le temps de compression vu par le contrôleur, qui comprend le temps de calcul des règles compressées, la suppression de la table de transfert en cours, le formatage des règles compressées aux standards OpenFlow, et l'injection des nouvelles règles dans le switch.

Nous remarquons que le temps de compression par commutateur reste de l'ordre de quelques millisecondes. En effet, la compression prend environ 5 ms (resp. 7 ms) pour la compression à 500 et 1000 entrées (resp. à 2000 entrées). Même le pire cas - la compression lorsque la table

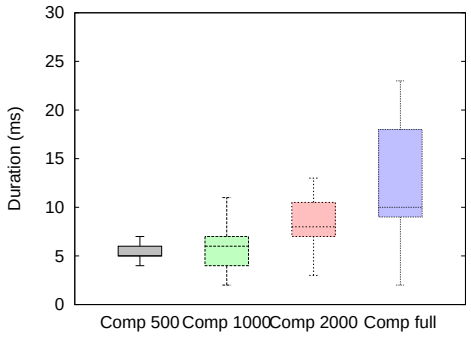


Figure 6.4: Durée moyenne d'une période de compression.

est pleine - représente moins de 18 ms pour la plupart des commutateurs avec une médiane à 9 ms. De plus, dans ce dernier cas, la compression séquentielle de tous les commutateurs ne nécessite pas plus de 152 ms. Cette période de compression est principalement due au temps nécessaire pour supprimer toute la table de routage en utilisant une demande de suppression et pour installer toutes les nouvelles règles dans le commutateur. En effet, le temps nécessaire pour calculer la table de routage compressée est négligeable comme indiqué dans [J11]. Il est important de noter ici que le code utilisé pour calculer les tables compressées est le même dans nos simulations et expériences. Ces résultats sont en ligne avec les résultats présentés dans [194] (figure 3). La raison pour laquelle nous avons des délais plus petits est dû au fait que, comme indiqué précédemment, nous n'attendons pas la fin de la compression avant d'envoyer la règle d'insertion de flots suivante ; de plus, nous supprimons toutes les règles en utilisant une seule action au lieu de supprimer chaque règle une par une.

Chemin de contrôle SDN. Dans le paradigme SDN, le lien entre le contrôleur et les routeurs SDN est un composant sensible, car le commutateur est limité par la vitesse de son CPU et il ne peut pas gérer les événements à un taux trop élevé. La figure 6.5 présente le trafic réseau entre les routeurs et le contrôleur dans les différents scénarios. Nous pouvons constater que la charge augmente fortement lorsque la limite du switch en terme de nombre de règles logicielles + matérielles est atteinte et que nous ne compressons pas les tables de routage. Après le temps t=2:30, la limite est atteinte et pour chaque paquet de chaque nouveau flot, chaque commutateur le long du chemin doit demander au contrôleur le port de sortie. Ces pics de trafic disparaissent lorsque nous compressons les tables de routage pour les limites 1000 et 2000 ou pour le cas de compression lorsqu'elles sont pleines. En ce qui concerne le scénario de compression à 500, nous constatons l'apparition de pics élevés après la première heure. Ils résultent d'événements de compression successifs (plus de 16 000 dans nos expériences comme on peut le voir dans le tableau 6.3) qui sont déclenchés par une nouvelle arrivée de paquets. En effet, dans ce scénario, la plupart des switchs effectueront une compression pour chaque nouveau flot, puisque le nombre total de règles après compression reste supérieur au seuil.

Pour comprendre l'impact du plan de contrôle sur le plan de données, nous devons examiner trois paramètres clés que nous détaillons dans les sections suivantes: (i) le taux de perte pour tous les scénarios; (ii) le retard du premier paquet de nouveaux flots qui devrait être plus élevé lorsqu'il n'y a pas de compression (au moins après t=2:30) ou pour compression à 500 et (iii) le délai des paquets suivants (paquets 2 à 5) qui devrait être plus grand dans le cas pour lequel il

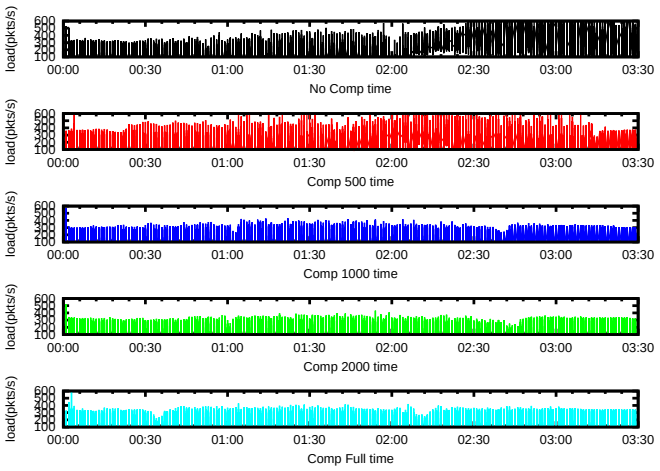


Figure 6.5: Trafic réseau entre les commutateurs et le contrôleur.

Seuil	No Comp	Comp 500	Comp 1000	Comp 2000	Comp full
# compressions	NA	16 594	95	28	20
% perte paquets	6.25×10^{-6}	0.003	5.65×10^{-4}	2.83×10^{-5}	3.7×10^{-4}

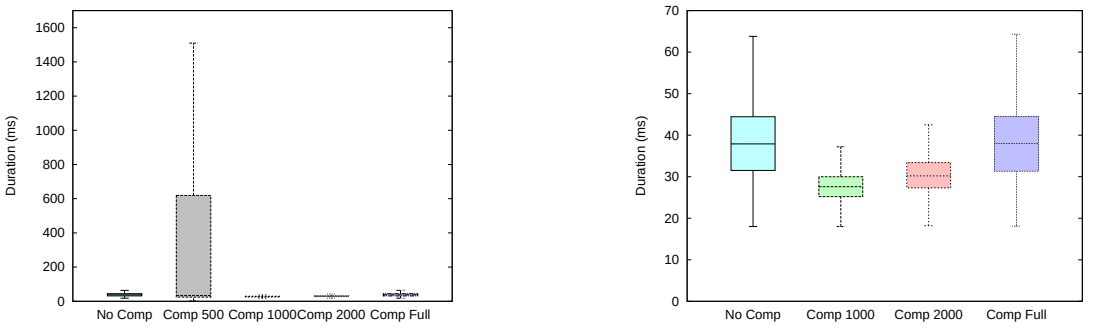
Table 6.3: Nombre total de compressions et taux de perte de paquets.

n'y a pas de compression lorsque la table est pleine.

Installation de nouvelles règles : Impact sur le délai du premier paquet. L'arrivée du premier paquet d'un flot déclenche la détection d'un nouveau flot. Son délai comprend donc le temps de contact avec le contrôleur et le temps d'installation des règles pour le flot. La Figure 6.7 montre le délai de transmission du premier paquet de chaque flot dans le réseau. Nous observons dans la figure 6.6 que, pour les scénarios avec compression à 1000 règles et compression dans les règles 2000, le premier délai de paquets va de 25 ms à 35 ms. Cette augmentation par rapport aux paquets ultérieurs du même flot - qui peut atteindre un facteur de 10 comme on le verra dans la section suivante - met en évidence le prix à payer pour obtenir et installer une règle de transfert dans le logiciel. Les résultats peuvent s'aggraver de manière significative si le contrôleur modifie fréquemment la règle de routage, comme pour la compression 500. En effet, pour ce cas spécial, le troisième quantile atteint jusqu'à 600 ms pour le premier délai de paquets.

De manière surprenante, les cas sans compression et compression à la limite conduisent à des résultats similaires. La compression lorsque la table est pleine devrait conduire intuitivement à de meilleures performances, comme dans un certain nombre de cas, un nombre limité de nouvelles règles sont nécessaires et peuvent être installées par rapport au boîtier sans compression. Cependant, dans nos tests, la table devient pleine après 2 heures et 30 minutes d'expérience (sur 3 heures). D'où la similitude des résultats dans la Figure 6.6.

En fait, lorsque la table est pleine, l'impact est frappant, comme on peut le voir dans la série temporelle de la figure 6.7, qui montre l'évolution du délai des premiers paquets des nouveaux flots au cours de l'expérience. Nous ne comparons ici que les scénarios *sans compression* et *compression avec un seuil de 1000 règles*. Sans compression, le délai croît progressivement de

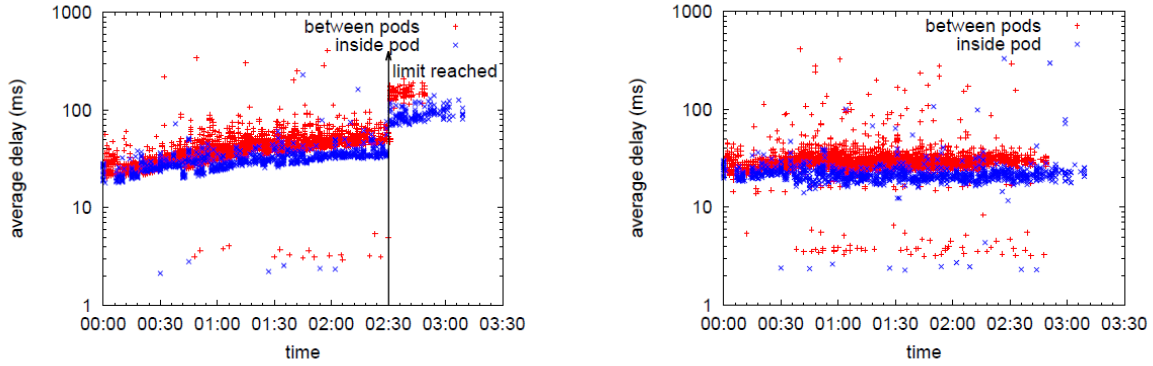


(a) Echelle normale

(b) Echelle réduite

Figure 6.6: Distribution sur tous les flots du délai du premier paquet.

20ms à 60ms alors qu'en compressant régulièrement le délai reste constant, aux alentours de 25 ms. Au bout de 2h30, lorsque la limite des 65 536 règles est atteinte, le délai sans compression bondit au dessus de 100ms, car à chaque saut le contrôleur transmet la règle de "forwarding", mais ne peut la stocker dans la table. Cela démontre l'avantage de la compression en terme de délai.



(a) Sans compression

(b) Compression 1000

Figure 6.7: Délai du premier paquet.

Finalement, notez que les résultats obtenus ici sont influencés par le fait que nous utilisons des règles logicielles, ce qui augmente le délai d'installation des règles. Les résultats des expériences utilisant des règles matérielles (c'est-à-dire TCAM) sont exclusivement fournis dans [J11].

Délai des paquets ultérieurs. Dans nos expériences, le délai observé par les paquets 2 à 5 de chaque flot est inférieur à 4 ms la plupart du temps pour tous les scénario comme on peut le voir dans la Figure 6.8. La compression 500 est légèrement différente (le troisième quartile atteint jusqu'à 5 ms), mettant en évidence l'impact négatif de la fréquence élevée des événements de compression sur le chemin de données des commutateurs.

La figure 6.9 montre les délais au cours du temps. On voit que lorsque toutes les règles

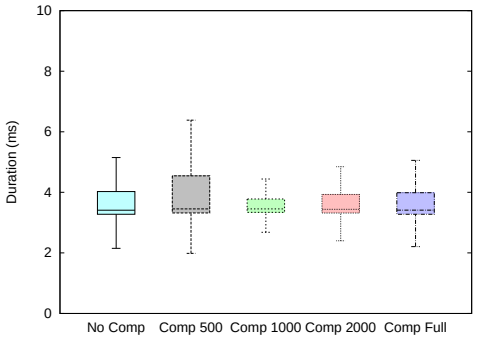
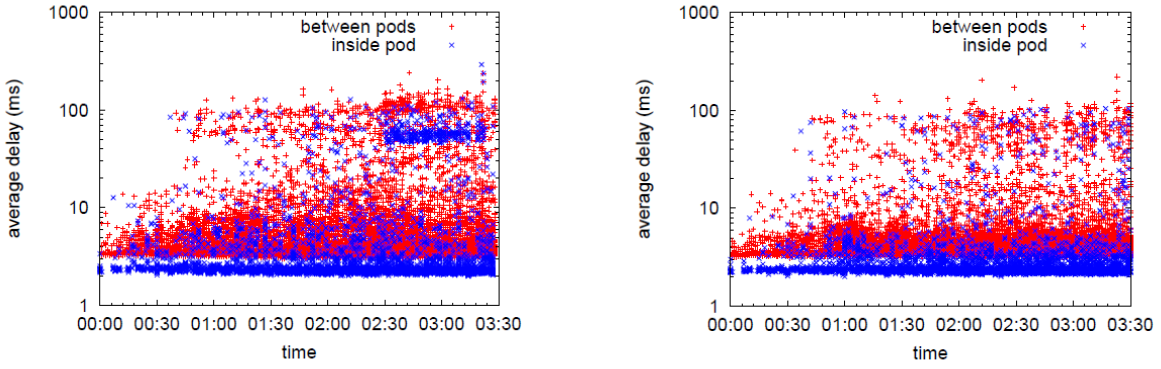


Figure 6.8: Distribution (boxplot) du délai moyen des paquets 2 à 5.

de routage nécessaires sont installées avec succès et que la fréquence de compression est faible le délai des paquets 2 à 5 est très majoritaitement entre 2 ms et 6 ms. (Figures 6.9b). Sans compression, alors que la plupart des paquets connaissent similairement un délai entre 2 ms et 6 ms avant que la limite de table ne soit atteinte, tous les nouveaux paquets entrants verront un délai égal ou supérieur à 40 ms après (Figure 6.9a).



(a) Sans compression

(b) Compression 1000

Figure 6.9: Délai des paquets 2 à 5.

En résumé, nous notons que la mise en place d'une limite de table basse (par exemple, 500) a un impact négatif sur le trafic passant par le réseau, alors que celle d'une limite entre 1000 et 2000 fournit des performances améliorées pour le trafic réseau. Ceci est dû au fait que dans nos scénarios, la table compressée avait une taille supérieure à 500 alors qu'elle était toujours inférieure à 1000. Par conséquent, afin d'exploiter toujours le bénéfice de Minnie, nous conseillons de définir un seuil dynamique qui changera en fonction de la taille de la table compressée - voir la section 6.7.3.

6.7.3 Discussion

Les résultats obtenus dans les sections 6.6.2 et 6.7 par simulation et expérimentation démontrent respectivement la faisabilité et l'efficacité de Minnie. Nous discutons ici de plusieurs points pratiques et des extensions possibles de notre algorithme.

Traitement de différentes charges de travail. Nous avons utilisé un schéma de trafic all-to-all, ce qui constitue le pire cas en termes de charge de travail du trafic qu'une application pourrait générer dans le réseau. Cela a toutefois été réalisé avec 16 IP par serveur dans la partie expérimentation et 1 IP par serveur dans la partie simulation, ce qui pourrait sembler assez limité. Cependant, dans un déploiement opérationnel du réseau, il est raisonnable d'admettre que les règles SDN sont principalement installées sur un sous-réseau IP, alors que les règles basées sur les flots (créées avec la correspondance de tous ou plusieurs champs de la norme OpenFlow) pourraient être rarement utilisées. Nos résultats peuvent donc être interprétés comme un routage du trafic total entre plusieurs sous-réseaux IP par serveur, comme on s'attend à l'observer dans un centre de données typique où la virtualisation est utilisée. Cela signifie que Minnie est capable de faire face au pire scénario de trafic impliquant un grand nombre d'hôtes finaux.

Suppression de règles. Tous les scénarios étudiés dans ce travail ont considéré des flots avec durée de vie illimitée afin d'obtenir le pire scénario concernant le nombre total de règles impliquées. Cependant, dans la pratique, les flots sont actifs pendant un temps limité. Nous discutons ici une extension de Minnie qui permettrait de gérer le départ des flots.

OpenFlow permet d'utiliser des temps d'expiration (*idle or hard timeouts*) pour supprimer des règles si aucun autre paquet du flot n'arrive (idle) ou après un intervalle de temps fixé (hard). Ces temps d'expiration pourraient être définis sur les commutateurs de niveau 0, permettant la détection de flots inactifs par Minnie. Les temps d'expiration fixes permettent au contrôleur de connaître l'état exact de chaque commutateur de niveau 0 sans aucun retour du commutateur. Avec les délais d'expiration *idle*, le contrôleur peut spécifier (dans OpenFlow) lorsqu'une règle est insérée, que le commutateur doit notifier le contrôleur lorsque la règle expire. Avec l'information exacte des règles actuellement activées, Minnie, qui conserve une version non compressée de toutes les règles dans tous les commutateurs, peut supprimer toutes les règles agrégées inutilisées. À mesure que de plus en plus de règles sont supprimées, le module de compression pourrait également être lancé pour produire une table plus petite à insérer à la place de celle actuelle.

Impact de la compression sur la mise à jour des règles. Nous avons discuté de l'impact de la compression des règles sur la performance de la mise à jour des règles dans plusieurs parties du document. Nous résumons ici les résultats. Nous avons 3 cas de mise à jour de règles dans les tables compressées :

- Ajout d'une nouvelle règle simple (en supposant que les tailles des tables sont inférieures au seuil de compression). Cet événement est dû à l'arrivée d'un nouveau flot. Dans ce cas, il n'y a aucun impact de compression sur la mise à jour des règles. Notez que, grâce à des règles agrégées, une nouvelle arrivée de flots nécessitera une nouvelle entrée dans les commutateurs OVS de niveau 0, mais ne nécessitera aucune nouvelle entrée dans les commutateurs d'accès ou à des niveaux de commutateurs plus élevés, si le nouveau flot est acheminé par des règles agrégations déjà existantes. Dans ce cas, nous ne devons pas mettre à jour la table de routage.

- Suppression d'une règle. Cela se fait en particulier lorsqu'un flot se termine. Cette opération est discutée ci-dessus et n'a pas encore été testée. Cependant, le contrôleur sait quel flot utilise quelle règle (simple ou agrégée) et peut donc facilement connaître la

règle à supprimer (ou non) lorsqu'une entrée expire au niveau des commutateurs OVS de niveau 0. C'est une opération rapide.

- Événement de compression. Si une table est pleine, nous compressons totalement la table et nous envoyons la nouvelle table compressée au commutateur correspondant. Nous mettons ensuite à jour la table de commutation en effectuant une première opération de suppression pour supprimer l'ancienne table, puis nous envoyons les nouvelles règles à insérer dans le plus petit nombre de paquets. Nous avons mesuré expérimentalement la durée de ces opérations et testé leurs impacts sur les retards et les pertes de paquets. Nous avons d'abord évalué le temps nécessaire à la réalisation d'un événement de compression (temps de compression, temps d'envoi d'une nouvelle table sur le commutateur et heure des mises à jour). Nous montrons que ce temps est de l'ordre de quelques ms, tel que présenté dans la Figure 6.4. Rappelons que, si un événement de compression est nécessaire lors d'un nouveau flot, nous envoyons d'abord les règles de transfert pour le nouveau flot, et nous compressons uniquement ensuite. Évitant ainsi un délai supplémentaire pour un nouveau flot en raison d'un événement de compression. Nous avons également évalué l'impact de la compression des règles sur le réseau grâce à nos expériences. Nous rapportons les délais de paquets et les taux de perte dans nos expériences et comparons des scénarios avec compression et sans compression. Nous montrons que même avec une charge élevée (1 Gbps) pour le scénario de charge élevée (HLS), le taux de perte et le délai ne sont pas affectés, voir par exemple Tableau 7 et Figure 16.

Limite de compression dynamique. La compression précoce aide à maintenir les tables de routage faibles. Cependant, le seuil ne doit pas être plus petit que la taille réelle de la table compressée, comme en témoigne le cas de la compression à 500 entrées dans la partie expérimentation. Pour contourner ce problème potentiel et profiter pleinement de la compression, il semble judicieux de définir une limite de compression dynamique. Nous pouvons commencer avec une limite basse (par exemple, 100 règles) et une fois qu'un certain pourcentage de notre limite est atteint (par exemple 80 %), nous déclenchons Minnie pour compresser la table de routage. Cette limite de compression est ensuite augmentée chaque fois que la table compressée résultante est supérieure à la limite réelle, par exemple, à 150 % de la taille de la table compressée actuelle.

Traitement de pics de trafic. Une dimension que nous n'avons pas explorée au cours de nos tests est l'arrivée possible de pics de trafic qui pourrait conduire à stresser la communication du commutateur-contrôleur et atteindre la limite de quelques centaines d'événements/s que le commutateur est capable de supporter. Cela pourrait être le cas d'une application qui génère beaucoup de demandes vers un grand ensemble de serveurs à un taux élevé. Dans cette situation, Minnie pourrait aider à atténuer la charge sur le contrôleur. En effet, plus vite on compresse la table de flots, plus il est probable que nous installions des règles d'agrégation qui feront que les commutateurs n'aient pas besoin de demander au contrôleur une règle pour chaque nouvelle connexion. On pourrait soutenir que la compression implique une modification complète de la table de routage des commutateurs, et donc un grand nombre d'événements (suppression, insertion) liés à la gestion de la table. Cependant, dans OpenFlow, ces événements peuvent être regroupés : toutes les insertions peuvent être envoyées en même temps au commutateur. **En résumé, Minnie devrait également aider à atténuer le stress du canal contrôleur-commutateurs en cas d'une arrivée brusque de nouveaux flots (*flash-crowds*).**

Sécurité. Finalement, notez que Minnie ne modifie pas le niveau de sécurité du réseau SDN. En effet, les règles ne sont pas compressées dans les commutateurs de niveau 0 qui connectent les machines virtuelles au réseau. Cela signifie qu'il n'y a pas de possibilité pour un paquet qui

appartient à un *tenant* d'être vu ou d'être inséré dans le réseau d'un autre *tenant*, à condition que les règles SDN au bord soient correctement écrites. La compression au bord pourrait effectivement donner l'opportunité au trafic d'un *tenant* d'entrer dans le réseau d'un autre *tenant* en utilisant une règle joker générique. Notez cependant que nous ne compressons pas au bord, non en raison d'une préoccupation de sécurité, mais pour éviter toute mauvaise conduite dans le processus de routage.

6.7.4 Conclusion et perspectives

SDN permet de formuler des règles d'acheminement complexes. Cependant, une telle flexibilité nécessite l'utilisation de mémoires TCAM coûteuses et de tailles limitées. Même si les capacités des TCAM devraient augmenter dans un proche avenir, nous devons encore accorder une attention particulière aux communication entre les commutateurs et le contrôleur qui ne doivent pas entraîner la surcharge des commutateurs SDN qui sont très limités en CPU. Il est donc nécessaire de réduire autant que possible le nombre de règles que chaque commutateur doit gérer.

Dans ce travail, nous avons introduit Minnie, qui vise à acheminer les flots tout en respectant les contraintes de capacité des liens et de table de routage SDN, en utilisant de la compression de table avec agrégation par source, par destination et par la règle par défaut. Nous avons étudié à travers des expériences numériques la polyvalence de Minnie sur une variété de topologies de centres de données et démontrons qu'il peut gérer près d'un million de flots avec pas plus de 1000 règles par commutateur.

Les résultats numériques ont été complétés par des expériences sur une plateforme émulant une topologie 4 Fat-Tree. Ces expériences ont confirmé la capacité de Minnie à réduire considérablement le nombre de règles à gérer, sans effet négatif notable sur les délais ou les taux de perte.

Chapter 7

Virtualisation des fonctions réseau et chaînes de fonctions de services

La virtualisation de fonctions réseaux (NFV) est une approche émergente dans laquelle les fonctions réseau ne sont plus exécutées par des équipements appelés middlewares mais peuvent être exécutées sur des serveurs génériques situés dans de petits centres de données [174]. Cette technologie vise à traiter les principaux problèmes de l'infrastructure middlebox des entreprises d'aujourd'hui, tels que le coût, la rigidité des capacités, la complexité de la gestion et les défaillances [237]. L'un des principaux avantages de cette approche est que les fonctions réseau virtuelles les (VNF) peuvent être instanciées et mises à l'échelle à la demande sans qu'il soit nécessaire d'installer de nouveaux équipements.

Les flux de réseau doivent souvent être traités par une séquence ordonnée de fonctions réseau. Par exemple, un système de détection d'intrusions peut avoir besoin d'inspecter le paquet avant une compression ou un cryptage. De plus, différents clients peuvent avoir des exigences différentes en termes de séquences de fonctions réseau à exécuter [157]. Cette notion est connue sous le nom de chaîne de fonctions de service (Service Function Chaining ou SFC) [162]. *Au cours de mes travaux, je me suis intéressé principalement aux problèmes algorithmiques et d'optimisation posés par ces chaînes de services.*

Dans le passé, la création d'une chaîne de services pour soutenir une nouvelle application prenait beaucoup de temps et d'efforts. Il s'agissait d'acquérir des appareils réseau et de les câbler ensemble dans l'ordre requis. Chaque service nécessitait un dispositif matériel spécialisé et chaque dispositif devait être configuré individuellement avec sa propre syntaxe de commande. Le transfert des fonctions réseau en logiciel signifie que la construction d'une chaîne de services ne nécessite plus l'acquisition de matériel. De plus, comme les charges applicatives augmentent souvent avec le temps, la construction d'une chaîne qui n'aurait pas besoin d'être reconfigurée dans le futur signifiait trop souvent un surdimensionnement pour soutenir la croissance. L'utilisation de machines virtuelles facilite le processus d'utilisation des ressources en plus d'optimiser leur utilisation : il est facile de (ré)assigner rapidement plus de capacités mémoire ou de calcul.

L'objectif du chaînage de fonctions de services (SFC) est de développer un ensemble de blocs architecturaux qui permettront aux opérateurs réseau de créer une topologie de services et d'instancier un chemin de fonctions de services à travers le réseau. SFC couvre le placement des fonctions réseau, la gestion des chaînes de services, les modèles de diagnostic et de sécurité. Nous étudions ici le problème suivant.

La même fonction virtuelle peut être répliquée et exécutée sur plusieurs serveurs. Il s'ensuit qu'un problème fondamental qui se pose lorsqu'il s'agit de chaînes de fonctions réseau est de

savoir comment placer ces fonctions sur les nœuds (serveurs) du réseau tout en atteignant un objectif spécifique. En effet, l'emplacement des VNFs affecte la latence de bout en bout encourue par les paquets traversant un SFC particulier. De plus, l'instanciation d'une VNF dans un nœud a un coût d'installation (licence, CPU, mémoire, énergie,...). Les objectifs d'optimisation peuvent varier en fonction des besoins de l'opérateur du réseau. Les exemples d'objectifs possibles sont la minimisation du nombre de nœuds réseau utilisés pour placer des fonctions virtuelles, la minimisation du coût du réseau, la minimisation de la latence totale sur tous les chemins et l'optimisation de la bande passante.

Dans ce chapitre, nous abordons la question de savoir comment placer de manière optimale les fonctions virtuelles dans le réseau physique afin de satisfaire aux exigences des chaînes de services de tous les flux du réseau. Le réseau est spécifié par un ensemble de nœuds V et de liens E. Le trafic est donné sous la forme d'un ensemble de demandes $\mathcal{D}$. Chaque demande est associée à une séquence ordonnée de fonctions réseau qui doivent être exécutées pour tous les paquets appartenant au même flux.

Nous présentons deux études différentes. La première [Ci36, J8] vise à réduire la consommation globale de bande passante d'un réseau. Nous considérons le problème commun du routage (choix du chemin physique de chaque demande) et du placement de la fonction virtuelle (choix des nœuds physiques sur lesquels les répliques VNF seront exécutées). Pour résoudre le problème, nous utilisons plusieurs techniques, la principale étant des schémas de décomposition de modèles mathématiques, que nous résolvons en utilisant la génération de colonnes. Le but de la deuxième étude [Ci32] est de réduire le coût d'installation global des fonctions réseau. Nous considérons un scénario dans lequel un opérateur a déjà placé ses demandes (en utilisant les plus courts chemins par exemple). Nous fournissons des algorithmes d'approximation pour résoudre le problème.

Défi principal : Modéliser les contraintes d'ordre entre les VNF. Le problème du placement des ressources virtuelles dans un réseau a été largement étudié, voir par exemple [185] et l'état de l'art de ce chapitre. Cependant, peu d'ouvrages ont abordé le problème des ressources ordonnées. Nous présentons ici deux méthodes pour prendre en compte l'ordre en utilisant des graphes en couches.

7.1 Placement optimal de chaînes de services réseaux

Résumé. La virtualisation des réseaux, portée en partie par l'initiative virtualisation des fonctions de réseau (Network Function Virtualization - NFV), apporte une plus grande flexibilité aux opérateurs de réseaux ainsi qu'une baisse des coûts de gestion. Les fonctions réseau exécutées sur du matériel dédié peuvent maintenant être exécutées par des serveurs virtuels. Un service correspond alors à une séquence de fonctions, appelée chaîne de services, à effectuer dans un ordre précis sur l'ensemble des flots du service. Le problème considéré est alors de satisfaire les requêtes des chaînes de services tout en trouvant le meilleur compromis entre l'utilisation de la bande passante et le nombre d'emplacements pour héberger les fonctions réseau.

Nous proposons un modèle de génération de colonnes pour le routage et le placement de chaînes de service. Nous montrons au travers d'expérimentations poussées que nous pouvons résoudre le problème de façon optimale en moins d'une minute pour des réseaux ayant une taille allant jusqu'à 65 nœuds et 16 000 requêtes. Nous étudions aussi le compromis entre l'utilisation de la bande passante et le nombre de nœuds capables d'héberger des fonctions réseau.

Catégorie	Examples de fonctions
Inspection de paquets	IPFiX, firewalls, IPS, DDoS
Optimisation du trafic	Video transcoding, TCP optimization, traffic shaping, DPI
Proxys pour protocoles	Carrier-grade NAT, DNS cache, HTTP proxy/cache, SIP proxy, TCP proxy, session border controllers, WebRTC gateways
Value-added services (VAS)	Ad insertion, header enrichment, WAN acceleration, advanced advertising, URL filtering, parental control

Table 7.1: Exemples de fonctions de middleboxes qui peuvent être incluses dans des chaînes de services [189].

7.1.1 Introduction

Ces dernières années, les opérateurs de réseaux de télécommunications ont vu l'arrivée de nouveaux paradigmes promettant de faciliter la gestion de leurs réseaux. Parmi ces paradigmes, l'initiative NFV vise à virtualiser les fonctions réseau. En effet, dans les réseaux actuels, les fonctions réseau sont effectuées grâce à du matériel dédié, propriétaire et fermé, qui souffre de coûts opérationnels importants et d'un manque de flexibilité. En virtualisant les fonctions réseaux, les opérateurs peuvent déployer les fonctions sur du matériel générique en utilisant des serveurs virtuels, permettant ainsi une plus grande flexibilité dans la gestion et la création de nouveaux services pour les clients.

Dans un réseau, chaque service est associé à un ensemble de fonctions qui sont appliquées aux requêtes utilisant ce service. Lorsque les fonctions doivent être appliquées dans un ordre spécifique, on parle alors de *Chaîne de Services*. Par exemple, un service de voix sur IP peut demander l'exécution d'un pare-feu suivi d'un contrôle d'accès. Dans ce chapitre, nous considérons le problème du *Placement de Chaînes* qui consiste à placer les fonctions sur le réseau de sorte à ce que les flots puissent parcourir les fonctions correspondantes à leur service dans le bon ordre. Plusieurs variantes du problème existent avec des objectifs tels que la minimisation du nombre de nœuds pouvant héberger des fonctions [160] ou bien le nombre de multiplications/copies de fonctions [169] mais nous considérons ici la minimisation de la bande passante.

Notre contribution est la conception d'un *modèle mathématique avec un schéma de décomposition* qui permet une *résolution exacte qui passe à l'échelle*, alors que presque tous les algorithmes précédents de la littérature (voir la section 7.1.2) sont soit des heuristiques soit des algorithmes exacts mais qui ne passent pas à l'échelle. Au meilleur de notre connaissance, nous sommes les premiers à proposer un modèle exact qui passe à l'échelle pour un grand nombre de nœuds et de requêtes. Nous sommes en mesure de résoudre en quelques minutes des instances avec près de 10.000 demandes différentes. Le modèle permet ensuite d'étudier le *meilleur compromis entre le nombre de répliques de VNF, le meilleur emplacement des VNF, les besoins en ressources et*

les délais de bout en bout.

Cette section est organisée de la manière suivante. Dans la section 7.1.2, nous passons en revue les travaux antérieurs liés au provisionnement SFC et au placement de VNF. Nous présentons formellement le problème, le graphe en couches nécessaire pour la résolution ainsi que le modèle de génération de colonnes dans la section 7.1.3. Nous étudions dans la section 7.1.4 les performances de notre modèle, puis nous utilisons notre modèle pour étudier le compromis entre le nombre de nœuds capables d'héberger des fonctions et la bande passante requise.

7.1.2 État de l'art

Plusieurs travaux proposent des formulations mathématiques partielles et exactes pour le problème de provisionnement de SFC et pour optimiser différentes fonctions objectifs. Notez que certains travaux étudient le problème de provisionnement SFC en utilisant la théorie des jeux [128].

Formulations mathématiques hybrides. Dans Martini *et al.* [168] et Riggio *et al.* [160], les auteurs ne résolvent que le placement et le routage pour chaque requête de manière indépendante sans optimiser l'ensemble des requêtes. Une heuristique basée sur un ILP est proposée dans Gupta *et al.* [175]. Les auteurs ne considèrent que les k-plus courts chemins pour chaque requête dans le réseau et une contrainte de capacité de nœud simplifiée, pour laquelle une seule fonction par nœud peut être déployée.

Formulations mathématiques exactes. Luizelli *et al.* [169] fournit un modèle exact minimisant le nombre d'instances de fonctions dans le réseau. Cependant, ils ne considèrent que quelques dizaines de demandes. Savi *et al.* [158] proposent une formulation exacte dans laquelle le nombre de nœuds VNF est minimisé. Leur modèle prend en compte les coûts supplémentaires inhérents à l'environnement multi-cœurs. Cependant, ils ne fournissent des résultats que sur un petit réseau. Dans Bari *et al.* [151], les auteurs considèrent comme fonction objectif la dépense opérationnelle (OpEx) pour un scénario de trafic quotidien. Mohammadkhan *et al.* [165] proposent un modèle exact avec des heuristiques visant à minimiser l'utilisation maximale du CPU et des liens. La portée des expériences est limitée au cas où le nombre de noyaux par service est limité à un.

Les ILP proposés dans la littérature ne sont pas adaptés aux grands réseaux. Nous indiquons la taille des instances qu'ils résolvent dans la table 7.2. Nous pouvons observer qu'ils sont plutôt petits par rapport à ceux que nous avons utilisés dans nos résultats de calcul, voir la section 7.1.4. À notre connaissance, en utilisant la génération de colonnes, notre travail est le premier à résoudre de manière optimale le problème du placement de SFC dans un réseau de 50 nœuds et pour des scénarios de demandes all-to-all (10 000 requêtes). Même si certains travaux [158, 169] minimisent le nombre de réplications de fonctions dans le réseau, nous sommes également les premiers à considérer les contraintes sur le nombre de réplications de fonctions sur l'ensemble du réseau tout en minimisant l'utilisation de la bande passante. Certains travaux récents ont aussi proposé des méthodes pour résoudre de grandes instance, en particulier [140].

7.1.3 Modèle

Le réseau est représenté par un graphe $G = (V, L)$ où V représente l'ensemble des nœuds du réseau et L l'ensemble des liens. L'ensemble des requêtes est donné par $\mathcal{SD}$, et elles sont caractérisées par une source v_s, une destination v_d, une chaîne de fonctions c et une bande passante requise D^c_{sd}. Soit F l'ensemble des fonctions réseau virtuelles (VNFs). Chaque chaîne

Modèles	#sommets	#demandes
Mohammadkhan *et al.* [165]	22	60
Savi *et al.* [158]	10	2000
Luizelli *et al.* [169]	50	20
Riggio *et al.* [160]	20	1
Martini *et al.* [168]	26	8

Table 7.2: Taille maximum des topologies et nombre de requêtes résolues dans un délai raisonnable pour les modèles trouvés dans la littérature.

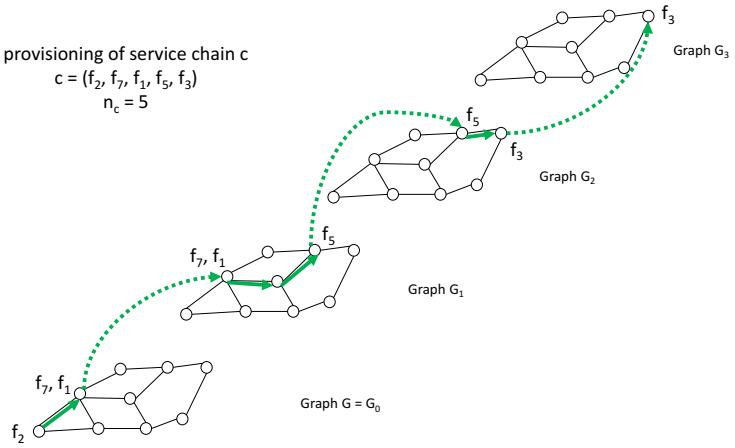


Figure 7.1: Graphe en couches.

de services c est définie par une séquence de n_c fonctions virtuelles qui peuvent être répétées. C note l'ensemble des chaînes. Le nombre de cœurs CPU requis par une fonction f par unité de bande passante est égal à Δ_f. Seul un sous-ensemble de nœuds $V^{VNF} \subseteq V$ peut héberger des fonctions virtuelles[1]. Chaque nœud $v \in V^{VNF}$ possède une capacité en nombre de cœurs CPU, noté CAP_v, qui sont utilisés par les fonctions virtuelles exécutées dessus. Chaque lien ℓ est limité par sa bande passante maximale, notée CAP_ℓ. *L'objectif est de minimiser la bande passante utilisée dans le réseau tout en satisfaisant toutes les requêtes et en respectant les capacités de liens et de nœuds.* Chaque requête doit être affectée à un chemin sur lequel ses fonctions sont placées dans le bon ordre.

Graphe en couches. Sur une idée similaire à [150], nous modélisons le problème grâce à un graphe en couches. Ce graphe est essentiel pour résoudre efficacement le modèle de génération de colonnes proposé. Le graphe G est transformé en un graphe G^L avec $\max_{c \in C} n_c + 1$ couches. Chaque couche est une copie exacte de G et pour chaque nœud v du réseau, on note v^i le nœud correspondant à la couche i. Les couches $i-1$ et i sont alors connectées par des liens allant de v^{i-1} à v^i, voir la figure 7.1 pour une illustration. Trouver un chemin et un placement de fonctions pour la requête (v_s, v_d, c, D^c_{sd}) est alors équivalent à trouver un chemin sur G^L du sommet v_s sur la couche initiale (v^0_s) au sommet v_d sur la n_c-ième couche $(v^{n_c}_d)$. En effet, chaque couche représente la progression de la chaîne, c.à.d., être sur la troisième couche indique que la 2^e fonction a été exécutée. L'emplacement d'une fonction est alors donnée par les arcs entre les couches utilisées par le chemin.

[1] Dans la suite, ces nœuds sont appelés nœuds VNF

Modèle de génération de colonnes. Nous présentons maintenant un modèle de génération de colonnes, appelé NFV_CG. Ce modèle se base sur le concept de *Chemin de Services* qui représente un chemin potentiel pour une requête. Nous introduisons les notations suivantes pour la formulation du modèle :

- $p \in P^c_{sd}$ représente un *Chemin de Services* pour la requête (v_s, v_d, c, D^c_{sd}). Il est composé d'un chemin et d'un ensemble de paires $\{(v, f) : f \in c\}$, où (v, f) indique que la fonction f est installée sur le nœud v.
- $a^p_{fv} \in \{0, 1\}$, où $a^p_{fv} = 1$ si f est exécutée sur le nœud v pour le *chemin de Services* $p \in P^c_{sd}$.
- $\delta^p_\ell \in \mathbb{N}^+$, où δ^p_ℓ représente le nombre d'occurrences du lien ℓ dans le chemin p.

Puisque la formulation de NFV_CG possède un nombre exponentiel de variables, nous utilisons la génération de colonnes afin de ne considérer qu'un sous-ensemble de *Chemins de Services*. Nous rappelons que la résolution par génération de colonnes combine un *Problème Maître Réduit (PMR)*, c.à.d le *Problème Maître* avec un sous-ensemble des variables, ainsi qu'un *Problème auxiliaire, appelé "Pricing" (PP)*. Le PMR et le PP sont résolus en alternance jusqu'à ce que le PP ne puisse plus générer de chemins de services qui améliorent la solution optimale du PMR. Dans ce cas, la solution courante de la relaxation du PMR est optimale et une solution entière peut en être dérivée. Le *Problème Maître* est formulé de la façon suivante :

Variables: $y^{sd,c}_p \geq 0$, où $y^{sd,c}_p = 1$ si la requête (v_s, v_d, c, D^c_{sd}) est routée sur le chemin de service p, 0 sinon.

$$\textbf{Objectif:} \quad \min \sum_{(v_s,v_d)\in\mathcal{SD}} \sum_{c\in C_{sd}} \sum_{p\in P^c_{sd}} \left(D^c_{sd} \sum_{\ell\in L} \delta^p_\ell \right) \times y^{sd,c}_p \tag{7.1}$$

$$\textbf{Contraintes:} \quad \sum_{p\in P^c_{sd}} y^{sd,c}_p = 1 \qquad c \in C_{sd}, (v_s, v_d) \in \mathcal{SD} \tag{7.2}$$

$$\sum_{c\in C_{sd}} \sum_{(v_s,v_d)\in\mathcal{SD}} \sum_{p\in P^c_{sd}} \left(D^c_{sd}\, \delta^p_\ell\right) \times y^{sd,c}_p \leq \text{CAP}_\ell \qquad \ell \in L \tag{7.3}$$

$$\sum_{c\in C_{sd}} \sum_{(v_s,v_d)\in\mathcal{SD}} \sum_{p\in P^c_{sd}} \left(\sum_{f\in F_c} D^c_{sd}\, \Delta_f\, a^f_{vp} \right) \times y^{sd,c}_p \leq \text{CAP}_v \qquad v \in V^{\text{NFV}} \tag{7.4}$$

L'objectif est de minimiser la bande passante utilisée sur le réseau, ce qui correspond à minimiser la longueur des chemins utilisés multiplié par les débits des requêtes. Les contraintes (7.2) assurent qu'exactement un chemin de service est choisi pour chaque requête. Les capacités de liens et de nœuds sont respectées grâce aux contraintes (7.3) et (7.4), respectivement.

Le problème auxiliaire consiste à trouver un *Chemin de Services* pour une requête qui minimise le coût réduit de la variable correspondante dans le PMR. Le coût réduit est calculé en utilisant les valeurs duales $u^{(j)}$ du PMR, où $u^{(j)}$ représente le vecteur des valeurs duales correspondant aux contraintes (j) du PMR. Le coût réduit d'un *Chemin de Services* pour la requête (v_s, v_d, c, D^c_{sd}) est donné par

$$-u^{(7.2)} + \sum_{\ell\in L} \sum_{i=0}^{n_c} D^c_{sd}\, \varphi_{i\ell} \left(1 + u^{(7.3)}_\ell\right) + D^c_{sd} \sum_{v\in V} u^{(7.4)}_v \sum_{i=0}^{n_c} \Delta_{f^i_c}\, \alpha_{iv}$$

où $\varphi_{i\ell} = 1$ si la requête est approvisionnée sur le lien ℓ au niveau i et $\alpha_{iv} = 1$ si la $i$éme fonction de c, notée f^i_c, est exécutée sur le noeud v.

Service	Chaîne	Débit	% trafic
Services web	NAT-FW-TM-WOC-IDPS	$100\,\mathrm{kB\,s^{-1}}$	18.2%
VoIP	NAT-FW-TM-FW-NAT	$64\,\mathrm{kB\,s^{-1}}$	11.8%
Diffusion vidéo	NAT-FW-TM-VOC-IDPS	$4\,\mathrm{MB\,s^{-1}}$	69.9%
Jeux en ligne	NAT-FW-VOC-WOC-IDPS	$50\,\mathrm{kB\,s^{-1}}$	0.1%

Table 7.3: Chaînes de services [158]

Réseau	# req.	# nœuds NFV	# col. gén.	$z^\star_{\mathrm{LP}}$	$\tilde{z}_{\mathrm{ILP}}$	ε	Temps (ms)
`internet2`	360	5	440	2086.7	2086.7	0	22
		6	416	2075.4	2075.4	0	23
`atlanta`	840	7	1024	2625.1	2625.1	0	44
		8	944	2596.1	2596.1	0	36
`germany50`	9800	24	25 766	4217.6	4218.0	7.82×10^{-5}	7354
		25	24 381	4211.9	4212.3	9.01×10^{-5}	6463
`ta2`	16640	33	48 819	4231.6	4231.8	5.20×10^{-5}	29 681
		34	45 212	4233.2	4232.8	8.74×10^{-5}	45 212

Table 7.4: Résultats obtenus avec NFV_CG

Puisque $u^{(7.2)}$ est une constante pour chaque requête (v_s, v_d, c, D^c_{sd}), trouver un *Chemin de Service* est équivalent à trouver un plus court chemin entre v^0_s et $v^{n_c}_d$ dans G_L. Le poids d'un lien (u^i, v^i) entre deux noeuds d'un même niveau est égale à $1 + u^{(7.3)}_\ell$ et le poids d'un lien (v^i, v^{i+1}) entre deux niveaux est égal à $u^{(7.4)}_v \times \Delta_{f^i_c}$. Puisque les valeurs duales $u^{(7.3)}_\ell$ et $u^{(7.4)}_v$ sont positives, ce plus court chemin peut être trouvé avec l'algorithme de Dijkstra.

7.1.4 Résultats

Dans cette section, nous présentons les résultats obtenus avec NFV_CG. Premièrement, nous décrivons le jeu de données utilisé. Nous présentons ensuite les performances de NFV_CG, suivies d'une étude sur le compromis entre le nombre de nœuds NFV et la bande passante requise ainsi que le nombre de sauts.

Jeux de données. Afin d'émuler un trafic réaliste, nous avons extrait du rapport Cisco annuel la distribution du trafic Internet. En utilisant le débit moyen d'un service (voir Table 7.3), nous pouvons en déduire le nombre de requêtes demandant chaque service. Par exemple, sur une charge totale de $1\,\mathrm{TB\,s^{-1}}$, le streaming vidéo représente $699\,\mathrm{GB\,s^{-1}}$ ce qui correspond à environ 175k requêtes différentes. Les sources et destinations sont alors choisies uniformément au hasard et les requêtes possédant la même source, destination et chaîne sont alors agrégées. Nous avons généré nos requêtes sur 4 réseaux différents internet2, atlanta, germany50 et ta2. Lors de l'étude du compromis entre le nombre de nœuds et l'utilisation de la bande passante, nous sélectionnons les nœuds selon sur leur *betweeness centrality* [408].

Performance de NFV_CG. La Table 7.4 montre les résultats obtenus avec NFV_CG sur les

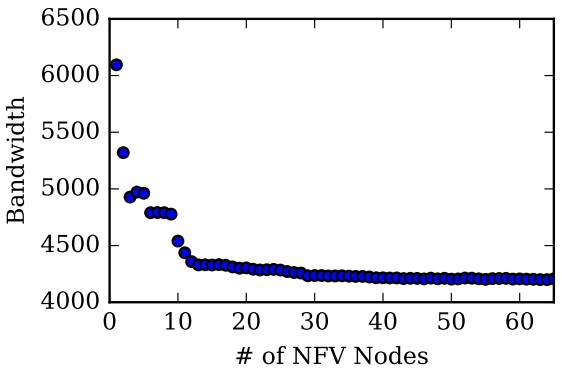


Figure 7.2: Bande passante

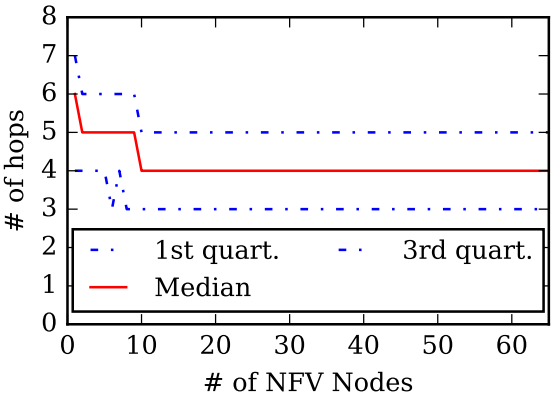


Figure 7.3: Longueurs des chemins en nombre de sauts

quatres réseaux lorsque le réseau possède aux alentours de 50% de nœuds NFV. Le trafic total de chaque instance est 1 Tbps. Les quatre dernières colonnes représentent la valeur optimale de la relaxation continue du MPR ($z^{\star}_{\text{LP}}$), la valeur optimale de la solution entière ($\tilde{z}_{\text{ILP}}$), le ratio entre les deux, ε, et le temps de calcul. Nous observons que $\varepsilon = 0$ dans la plupart des instances, ce qui signifie que la solution entière est optimale. Lorsque $\varepsilon > 0$, le ratio reste petit: $\tilde{z}_{\text{ILP}}$ est très proche de la valeur optimale entière. De plus, les temps de calcul restent très faibles, même pour les plus gros jeux de données (environ 45 s).

Compromis entre bande passante requise et nombre de nœuds NFV. Nous étudions ici le compromis entre le nombre de nœuds NFV présents sur le réseau et la bande passante totale requise. Je ne présente ici que les résultats sur `ta2` pour un trafic total de 1 Tbps. Les mêmes phénomènes peuvent être observés pour `internet2`, `atlanta` et `germany50` [J8].

La Figure 7.2 montre la bande passante en fonction du nombre de noeuds NFV. Sans surprise, la bande passante diminue lorsque le nombre de nœud NFV augmente. Comme chaque requête nécessite une chaîne, le chemin doit alors passer par les nœuds NFV dans le bon ordre, ce qui peut allonger les chemins de façon significative. Cependant, à partir de 50% de noeuds NVF, le gain de bande passante devient beaucoup plus faible.

Dans la Figure 7.3, nous montrons la distribution du nombre de sauts en fonction du nombre de nœuds NFV. Nous observons que la valeur médiane du nombre de sauts se stabilise à partir de 9 nœuds NVF dans le réseau. Bien que le gain en bande passante se stabilise plus tard, nous pouvons observer qu'une faible portion des requêtes est affectée lorsque le nombre de nœuds est supérieur à 10.

7.1.5 Conclusion

Nous avons examinons le problème de Placement de Chaînes de Services Réseau et proposé deux modèles ILP pour le résoudre. Nous montrons que le premier modèle compact d'ILP ne peut pas être utilisé pour de grands réseaux. Au contraire, avec un modèle de décomposition tel que le modèle NFV_CG, nous pouvons résoudre *exactement* le problème de placement dans ce cas. Ceci conduit à un premier modèle qui passe à l'échelle avec un nombre croissant de nœuds, mais aussi, avec une augmentation du nombre de requêtes avec des exigences de chaînes de service.

Nous abordons également la minimisation de la bande passante avec une limite sur le nombre de réplicas VNF. À cette fin, nous avons étendu nos modèles ILP et proposé un algorithme

heuristique basé sur NFV_CG et un problème de clustering k-mean avec capacité. À l'aide du modèle NFV_CG, nous avons étudié le compromis entre l'exigence de bande passante réseau, le nombre de nœuds compatibles VNF et la limite du nombre de répliques VNF. Nous avons découvert que des rendements décroissants se produisent lors de l'ajout de nœuds compatibles VNF, et que lorsque plus de 50% du réseau peut héberger des VNF, cet ajout ne présente presque plus d'avantages. Un compromis similaire existe pour le nombre de réplicas : lorsque vous démarrez à partir des configurations avec le nombre de réplicas minimum, l'ajout d'un petit nombre diminue considérablement l'utilisation de la bande passante, mais l'ajout de réplicas supplémentaires ne permet ensuite que des gains très limités.

7.2 Algorithmes d'approximation pour le placement de chaînes de fonctions de services avec contraintes d'ordre

Résumé. Dans cette section, nous nous intéressons à un *scénario dans lequel un opérateur a déjà routé ses demandes*, par exemple par les plus courts chemins. Il veut maintenant déterminer l'emplacement des fonctions virtuelles dont les services ont besoin en minimisant le coût de déploiement. Nous montrons que le problème peut être ramené à un problème de Set Cover, même dans le cas de séquences ordonnées de fonctions réseau. Cela nous permet de proposer deux algorithmes d'approximation à facteur logarithmique, ce qui est le meilleur facteur possible, à moins que P=NP. Finalement, nous évaluons les performances de nos algorithmes par simulations. Nous montrons ainsi qu'en pratique, des solutions presque optimales peuvent être trouvées avec notre approche.

7.2.1 Introduction

Notre objectif est de placer les fonctions réseau tout en réduisant le coût global de déploiement ou d'installation. Le coût vise à refléter le coût d'une machine virtuelle qui exécute une fonction virtuelle, comme les frais de licence, l'efficacité du réseau ou la consommation d'énergie [138]. Dans notre cadre, nous considérons une fonction de coût générale qui dépend à la fois du sommet du réseau et de la fonction de réseau. Nous appelons ce problème *Placement de SFC*.

Dans le cas où toutes les chaînes de service ne comportent qu'une seule fonction, le problème est connu pour être équivalent au problème de la couverture minimale (Minimum Set Cover Problem), comme indiqué dans [364]. Ceci implique que le problème est NP-difficile et qu'un algorithme ne peut pas atteindre un meilleur facteur d'approximation que $(1-\varepsilon)\ln|S|$ pour tout $\varepsilon > 0$, où S est l'ensemble des éléments à couvrir (sauf si P = NP) [199]. Aucun résultat positif n'est connu lorsque les longueurs des chaînes de fonctions de service sont supérieures à 1.

Nous démontrons ici que le cas générique, dans lequel les demandes ont des contraintes d'ordre sur les fonctions du réseau, correspond également à une instance de set cover. Nous montrons que le nombre exponentiel (en $|V|$) d'ensembles dans l'instance peut être réduit à un nombre polynomial (en $|V|$ et $|\mathcal{D}|$) en exploitant la structure spécifique des instances de set cover définies. Cela nous permet de proposer deux algorithmes efficaces pour le *SFC Placement Problem*. Le premier est basé sur l'arrondi LP. Le second est un algorithme glouton. Pour les deux, nous exploitons la structure spécifique du problème pour obtenir un temps d'exécution court, c'est-à-dire, polynomial également dans la longueur de la plus grande chaîne. Nous montrons que les deux algorithmes obtiennent des solutions dont le coût est dans un à facteur logarithmique de l'optimal.

Nous étudions ensuite les topologies de réseaux en arbres. Nous montrons d'abord que le problème est NP-difficile même dans ce cas restreint. Ensuite, nous étudions le scénario dans lequel tous les flux sont des flux montants ou descendants. Nous concevons un algorithme optimal pour ce cas particulier en utilisant la technique de programmation dynamique.

Nous avons implémenté nos algorithmes et comparé leurs résultats avec les solutions optimales obtenues par un programme linéaire. Nous montrons que le facteur d'approximation logarithmique n'est qu'une borne supérieure du pire cas et que nous pouvons obtenir des solutions proches de l'optimum dans la plupart des cas.

Bien qu'il existe de nombreux travaux sur le placement VNF ont été, aucun ne fournit des algorithmes avec des résultats théoriques prouvés pour le placement des chaînes de VNFs avec des contraintes d'ordre. La plupart des solutions sont basées sur des ILP, ne passant pas à l'échelle, ou des heuristiques, sans garantie d'approximation. À notre connaissance, *nous sommes les premiers à proposer un algorithme prouvé efficace pour placer des chaînes de fonctions de réseau virtualisées dans le réseau.*

Le reste de la section est organisée de la façon suivante. Dans la section 7.2.2, nous passons en revue les travaux connexes plus en détail. Dans la section 7.2.3, nous présentons la formulation du problème. Dans la section 7.2.4, nous montrons d'abord que *SFC Placement Problem* est équivalent à Set Cover même dans le cas général. Nous présentons ensuite les détails et l'analyse de nos algorithmes de placement. Notre algorithme optimal pour les topologies en arbre n'est pas présenté ici mais peut être trouvé dans [Ci32]. Dans la section 7.2.5, nous évaluons nos algorithmes proposés. Des conclusions sont tirées dans la section 7.2.6, ainsi que des questions ouvertes pour de futurs travaux.

7.2.2 Etat de l'art

Comme discuté, les algorithmes existants de placement fonctions virtuelles peuvent être grossièrement classés en deux catégories : ceux basés sur un ILP et les algorithmes heuristiques. Ces approches n'ont généralement aucune garantie théorique de performance.
Les travaux les plus proches du nôtre sont [185] et [120] car ils fournissent des résultats théoriques pour la performance des algorithmes qu'ils proposent.
[185] adresse le problème du placement des fonctions virtuelles dans le réseau physique. Chaque demande a un ensemble requis de VNF qui doivent être exécutées. L'objectif des auteurs est de minimiser le coût du réseau, donné par le coût d'installation d'une fonction sur un nœud et le coût de connexion qui dépend de la distance entre les clients et les nœuds à partir desquels ils obtiennent un service. Ils fournissent des algorithmes d'approximation quasi-optimaux avec des performances théoriquement prouvées. Cependant, l'ordre d'exécution des fonctions réseau n'est pas pris en compte dans leur modèle.
Dans [120], les auteurs se concentrent sur le problème du placement optimal et de l'allocation des VNF pour fournir un service à toutes les demandes. L'objectif est de minimiser le nombre total de fonctions réseau. Dans leur modèle, les routes des demandes sont fixées (comme pour nous). Cependant, ils étudient le scénario d'une seule fonction de réseau et laissent le placement de fonctions virtuelles avec contrainte d'ordre comme un *problème ouvert* pour de futures recherches.

7.2.3 Formulation du problème

Étant donné un digraphe $G = (V, E)$, avec un ensemble de fonctions $\mathcal{F}$ et un coût d'installation c, le problème prend comme entrée un ensemble de demandes $\mathcal{D}$. Une demande $d \in \mathcal{D}$ est

$G = (V, E)$	digraphe	$L(P)$	longueur du chemin P (nombre de sauts)
$\mathcal{P}$	ensemble de chemins	$c(v, f)$	coût de la fonction $f \in \mathcal{F}$ dans le sommet $v \in V$
$\mathcal{F}$	ensemble de fonctions	L	longueur du plus long chemin
$s(P)$	chaine de services du chemin P	k	taille de la plus longue chaîne de services

Table 7.5: Résumé des notations

modélisée par une paire composée par un chemin, c'est-à-dire une séquence de sommets, et une chaîne de fonctions de services, c'est-à-dire une séquence ordonnée de fonctions. La sortie du problème est une fonction de placement de coût minimum qui satisfait toutes les demandes de $\mathcal{D}$. Nous nommons ce problème *Problème de placement de SFC*.
Entrée: Un digraphe $G = (V, E)$ et un ensemble $\mathcal{D}$ de demande. Chaque demande $d \in \mathcal{D}$ est associée à un chemin `path(d)` $\in V^*$ et à une séquence de fonctions `sfc(d)` $\in \mathcal{F}^*$. Finalement, un coût $c : V \times F \to c(v, f)$, qui définit le coût d'installation de la fonction f dans le sommet v.
Sortie: Une *fonction de placement* qui est un sous-ensemble $\Pi \subset V \times F$ de positions de fonctions, tel que toutes les demandes de $\mathcal{D}$ sont satisfaites, c'est-à-dire, pour chaque demande les fonctions réseaux apparaissent sur son chemin dans l'ordre exact de sa chaine.
Objectif: minimiser $\sum_{(v,f)\in\Pi} c(v, f)$

7.2.4 Algorithmes d'approximation pour SFC-Placement

Nous montrons ici que le *Problème de placement SFC* est équivalent à une instance du Minimum Weight Set Cover Problem. Pour chaque demande $d \in \mathcal{D}$, nous notons $l(d)$ et $s(d)$ les longueurs respectives du chemin et de la chaîne associés. Soit `path(d)` $= u_1, u_2, ..., u_{l(d)}$ et supposons que d nécessite une séquence de fonctions $sfc(P) = r_1, r_2, ..., r_{s(d)}$.
Étant donné une demande d, nous construisons un réseau associé avec capacité $H(d, \Pi)$, comme montré en figure 7.4.

Définition 48. Le réseau $H(d, \Pi)$ associé avec la demande d est construit de la façon suivante :

- $H(d, \Pi)$ a $s(d)$ couches $L_1, L_2, ..., L_{s(d)}$. Chaque couche contient $l(d)$ sommets correspondant aux sommets de `path(d)`. Nous notons (u_i, j) le i-ème sommet de la couche j.
- Il y a un arc entre le sommet (u, j) et le sommet $(v, j + 1)$ si $u = v$ ou si u précède v dans `path(d)`.
- $H(d, \Pi)$ a deux sommets supplémentaires, s_d et t_d. Il y a un arc entre le sommet s_d et tous les autres sommets de la première couche et un arc entre tous les sommets de la dernière couche et t_d.
- Tous les arcs ont une capacité infinie, mais chaque sommet a une capacité limitée. La capacité du sommet u de la couche i est 1 si $(u, r_i) \in \Pi$ et 0 sinon.

Lemme 49. *Une demande $d \in \mathcal{D}$ est satisfaite par Π si et seulement s'il existe un $st - path$ dans le réseau associé avec capacité $H(d, \Pi)$.*

En utilisant cette notion de réseau associé, nous définissons le problème suivant:

Problem 50. Hitting-Cut-Problem $(\mathcal{D}, c)$ est une instance du Weighted Hitting Set problem pour laquelle les éléments sont les positions des fonctions (u, f), pour tous $u \in V$ et $f \in \mathcal{F}$. Son coût est $c(u, f)$. Les sous-ensemble de l'univers correspondent à toutes les st-coupes-sommets (st-vertex-cuts or st-separators) des réseaux associés $H(d, \Pi)$ pour tous $d \in \mathcal{D}$.

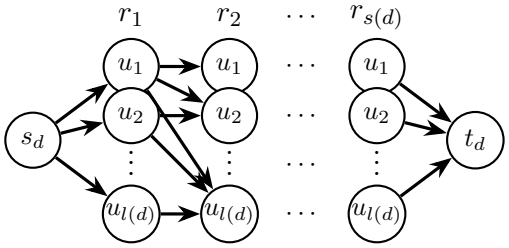


Figure 7.4: Le réseau associé de la demande $d \in \mathcal{D}$ routée sur le chemin `path(d)` $= u_1, u_2, ..., u_{l(d)}$ et qui a la chaîne `sfc(d)` $= r_1, r_2, ..., r_{s(d)}$

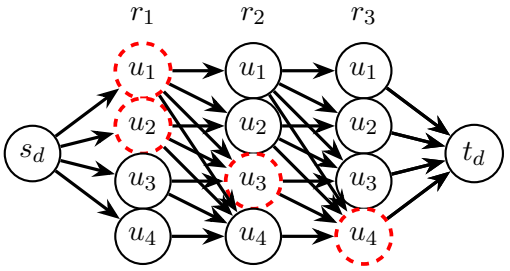


Figure 7.5: Exemple de coupe propre (sommets hachurés en rouge) pour le graphe en couche de la demande d qui a un chemin de longueur 4 et une chaîne de longueur 3.

Le problème est donc de trouver un sous-ensemble S d'éléments (positions de fonctions) touchant tous les sous-ensembles (coupes) de l'univers de coût minimum.

Proposition 51. Hitting-Cut-Problem $(\mathcal{D}, c)$ *est équivalent à* SFC-Placement $(\mathcal{D}, c)$.

Notre problème est donc équivalent à un Hitting Set Problem, pour lequel nous connaissons des algorithmes d'approximation. Cependant, le nombre de *st*-coupes-sommets est exponentiel dans le nombre de sommets du digraphe. Pour obtenir un algorithme polynomial, nous devons réduire la taille d'une instance du Hitting-Cut-Problem. À cette fin, nous utilisons le fait que vérifier seulement les coupes *extrémales* est suffisant (une coupe *extrémale* est une coupe qui n'est pas strictement incluse dans une autre coupe) et que, dans notre problème, les coupes extrémales des graphes associés ont une forme spécifique que nous appelons *st-coupes propres*. Voir la figure 7.5 pour un exemple.

Définition 52. Une *st*-coupe propre du graphe associé $H(d, \Pi)$ est une coupe de la forme :

$$\{\underbrace{(u_1,1),...,(u_{j_1},1)}_{\text{layer 1}}, \underbrace{(u_{j_1+1},2),...,(u_{j_1+j_2},2)}_{\text{layer 2}}, ..., \underbrace{(u_{j_1+j_2+\cdots+j_{s(d)-1}+1},s(d)),...,(u_{l(d)=j_1+j_2+\cdots+j_{s(d)}},s(d))}_{\text{layer } s(d)}\}$$

for $j_1, j_2, ..., j_{s(d)} \geq 0$, such that $\sum_{i=1}^{s(d)} j_i = l(d)$.

Proposition 53. *Le problème* SFC-Placement $(\mathcal{D}, c)$ *est équivalent à Hitting Set Problem avec* $\sum_{d\in\mathcal{D}} \binom{l(d)+s(d)-1}{s(d)-1}$ *ensembles comme entrée. Si chaque demande nécessite au plus* $s_{\max}$ *fonctions réseau et est associée à un chemin de longueur plus petite ou égale à* $l_{\max}$*, alors la taille de l'instance est au plus* $O(|\mathcal{D}| \cdot (l_{\max})^{s_{\max}-1})$.

Nous proposons deux algorithmes pour résoudre SFC-Placement, un algorithme glouton et un algorithme par arrondi LP.

Algorithme glouton. L'idée principale de l'algorithme glouton est d'éviter de générer toutes les coupes propres en montrant qu'il suffit de compter le *nombre de coupes propres non touchées.* Nous montrons ici que, en utilisant la programmation dynamique, ce nombre peut être compté en temps $O(|\mathcal{D}| l_{\max}^2 s_{\max})$.

Pour une demande $d = (\texttt{path(d)}, \texttt{sfc(d)})$, une fonction de placement Π peut être vue comme une matrice A_d avec $l(d)$ lignes et $s(d)$ colonnes et pour laquelle $A_d[i,j] = 1$ si et seulement si $(u_i, r_j) \in \Pi$. Nous notons $A_d[i : j, k : l]$ la sous-matrice de A_d qui ne considère que les lignes

de i à j et les colonnes de k à l. Pour une demande $d = (\texttt{path(d)}, \texttt{sfc(d)})$ et une fonction de placement Π (ou de façon équivalente A_d), nous notons $N(d)$ le nombre de coupes propres non touchées par A_d. Ce nombre peut être calculé en utilisant la fonction $N(r, c)$ définie de la façon récursive suivante. Nous avons $N(d) = N(l(d), s(d))$ avec

$$N(r, c) = \mathbb{1}_{i^*(r,c)=0} + \sum_{j_c=0}^{r-i^*(r,c)} N(n - j_c, c - 1), \text{if } c \geq 2$$
$$N(r, 1) = \mathbb{1}_{i^*(r,c)=0}$$

où $i^*(r, c)$ est définie de la façon suivante. Nous considérons la matrice $A_d[1 : r, 1 : c])$. Nous considérons les 1 placés dans la dernière colonne de la matrice, la colonne c. S'il n'y en a aucun, $i^*(r, c) = 0$. Sinon, $i^*(r, c)$ est l'index maximum d'un tel 1, càd, $i^*(r, c) = \max_{0 \leq i \leq l(d)}\{i, \text{ such that } A_d[i, c] = 1\}$.
$N(r, c)$ peut être calculé en utilisant la programmation dynamique. A chaque itération, l'algorithme sélectionne la paire (u, f) avec le plus petit coût moyen par nouvelle coupe propre touchée. La paire avec le coût minimal est ajoutée à la solution Π. Ensuite, le nombre de coupes propres restant à toucher est mis à jour. Ce processus est répété jusqu'à ce que toutes les coupes propres aient été touchées. La complexité de toute la procédure est $O(l_{\max}^2 s_{\max} |V|^2 |\mathcal{F}|^2 |\mathcal{D}|)$.

Approche par arrondi LP. L'idée ici est d'utiliser la formulation du problème qui recherche un chemin dans les réseaux associés $H(d, \Pi)$. Les variables de décision binaires sont maintenant de deux sortes:
(i) Variables de position ou de capacité : $x(u, f)$ indique si la fonction f est installée sur le sommet u. Elle correspond à la capacité partagée du sommet (u, f) des réseaux associés.
(ii) Variables de flots. Pour chaque demande $d \in \mathcal{D}$, nous avons une variable de flot f_{uv}^d pour chaque arc du réseau associé $H(d, \Pi)$. Les contraintes sont (i) des contraintes de capacité des sommets et (ii) des contraintes de conservation de flots. Il y a $O(|V| + s_{\max} l_{\max} |D|)$ contraintes, un nombre polynomial en $s_{\max}$. Nous renvoyons à [Ci32] pour une présentation plus détaillée du modèle et des algorithmes.

7.2.5 Résultats numériques

Dans cette section, nous évaluons les performances des algorithmes que nous avons proposés. Nous étudions comment le coût total d'installation et la précision de nos algorithmes varient en fonction (i) de différentes longueurs de chemin et (ii) d'un nombre croissant de demandes. Nous effectuons des expériences sur une topologie du monde réel: `germany50` (50 nœuds et 88 liens). Nous construisons nos instances de la manière suivante. Les nœuds source et destination d'une demande sont choisis aléatoirement[2] dans l'ensemble des sommets. Le chemin de la demande est donné par un plus court chemin entre ces deux nœuds et sa chaîne est composée de 2 à 6 fonctions choisies aléatoirement parmi un ensemble de 30 fonctions. Enfin, le coût d'installation d'une fonction sur un nœud est aléatoirement entre 1 et 5.
Dans la figure 7.6, nous comparons les performances des algorithmes dans le cas d'un nombre croissant de demandes. Le coût d'installation augmente avec le nombre de demandes, car le nombre de fonctions à placer augmente. Cependant, l'augmentation est sous-linéaire. La raison en est que plus les demandes dans un réseau sont nombreuses, plus les possibilités de partage des fonctions sont grandes. Le rapport d'optimalité est au plus de 21 % pour les deux algorithmes.
Dans la figure 7.7, nous considérons uniquement les demandes avec des paires de nœuds à des distances égales, de 1 à 7. Le coût total d'installation diminue strictement lorsque la longueur du chemin augmente. En effet, lorsque les chemins sont plus longs, les demandes tendent (en

[2] Aléatoirement signifie pour moi uniformément au hasard quand la distribution n'est pas précisée.

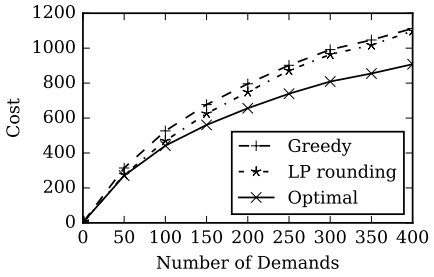


Figure 7.6: Coût d'installation moyen en fonction du nombre de demandes.

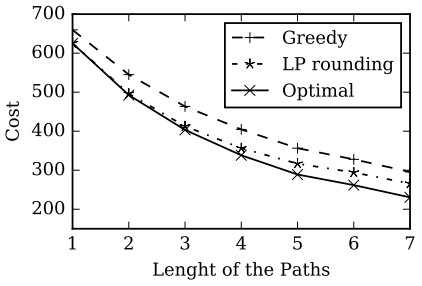


Figure 7.7: Coût d'installation moyen en fonction de la longueur des chemins.

moyenne) à partager plus de nœuds, réduisant le nombre de fonctions requises pour satisfaire toutes les demandes et donc le coût. Le rapport à la solution optimale ne dépasse jamais 25% pour les deux algorithmes.

7.2.6 Conclusion et Perspectives

NFV est une approche novatrice pour le déploiement de services réseau qui ouvre la voie à une gestion de réseau plus efficace et plus flexible. Par conséquent, placer les fonctions de réseau de manière rentable est une étape essentielle vers l'adoption du paradigme NFV. Dans ce travail, nous avons étudié le problème de placer des VNF pour satisfaire les contraintes d'ordre des flux et dans le but de minimiser le coût total d'installation. Puisque le problème formulé est NP-difficile, nous avons proposé deux algorithmes qui atteignent un facteur d'approximation logarithmique. À notre connaissance, aucun algorithme d'approximation n'avait été proposé dans la littérature pour le problème de placement de SFC jusqu'à présent. De plus, pour le cas particulier des topologies de réseau en arbre avec uniquement des flux montants ou descendants, nous avons conçu un algorithme optimal. Des résultats numériques valident l'efficacité de nos algorithmes.
Ce travail vise à proposer un premier cadre théorique pour l'étude du problème de placement avec contraintes d'ordre. Cependant, un problème non-résolu restant concerne les débits et la comptabilisation de contraintes pratiques telles que les *capacités logicielles* sur les fonctions réseau ou *capacités du hardware* sur les nœuds réseau. Une direction de recherche future intéressante peut concerner une étude de la possibilité d'obtenir des algorithmes d'approximation efficaces pour ces problèmes.

Une autre piste prometteuse pour d'autres recherches serait d'utiliser la faible dimension VC [156] des instances de Set Cover que nous avons construites. La VC-dimension est une mesure importante de la complexité d'une structure, et des algorithmes polynomiaux obtenant des performances supérieures à $\log N$ ont été proposés pour Set Cover pour des collections d'ensembles de faible dimension VC [390]. Dans notre cas, la structure très spécifique de nos instances (que nous utilisons pour fournir des algorithmes gloutons ou par arrondi LP efficaces) est en effet associée à une dimension VC beaucoup plus faible que celle des instances arbitraires. On peut voir cela comme une explication du bon comportement pratique de nos algorithmes. Cependant, notre tentative d'utiliser les algorithmes mentionnés ci-dessus conduit à des ratios d'approximation prouvables qui sont presque toujours (à l'exception de quelques ensembles de valeurs de paramètres qui ne correspondent pas à des cas pratiques) pire que celui fourni par LP-

arrondi et Greedy. Nous espérons néanmoins pouvoir utiliser cette approche de VC-dimension ou mieux comprendre la structure afin d'améliorer les ratios d'approximation que nous avons obtenus.

Chapter 8

Vers la mise en pratique de politiques vertes

8.1 Introduction

Comme discuté auparavant, l'efficacité énergétique de l'infrastructure des réseaux (que ce soit d'ISP, de centre de données ou d'entreprises) est un problème qui fait l'objet d'une préoccupation croissante, en raison à la fois de l'augmentation des coûts d'l'énergie et des inquiétudes concernant les émissions de CO2.

Pour commencer à résoudre le problème, plusieurs solutions éco-énergétiques (protocoles, algorithmes ou conception d'infrastructures) ont été proposées. *La plupart de ces solutions économes en énergie impliquent d'adapter dynamiquement l'utilisation des ressources réseaux en fonction de l'état courant du trafic et des demandes.* Nous pouvons citer comme exemples de ces solutions, le routage efficace en énergie [253], l'affectation de téléphones cellulaires à une station de base [305] ou l'adaptation dynamique du taux d'envoi de données (dynamic rate adaptation) [335]. Cependant, les protocoles et les équipements des réseaux existants ne permettent qu'avec difficulté de s'adapter à la dynamique de la demande. En effet, la configuration d'un réseau est une tâche difficile. Chaque équipement a de nombreuses configurations possibles et ces configurations doivent souvent être faites à la main. De plus, chaque fabricant de matériel utilise un langage spécifique de configuration. Les opérateurs sont donc très réticents à effectuer des changements sur leur réseau. La mise en pratique des solutions économes en énergie a donc été impossible pour le moment.

La situation a changé récemment avec l'émergence de nouvelles technologies qui sont en train de changer en profondeur les réseaux, à savoir les réseaux logiciels et la virtualisation des (fonctions) réseaux (correspondant aux réseaux virtuels ou VN en bref pour *Virtual Networks* et à la virtualisation des fonctions réseaux). En effet, celles-ci facilitent le contrôle des réseaux et permettent de mettre en œuvre des politiques dynamiques de gestion, tout en diminuant les coûts. Ces technologies offrent un nouveau potentiel de mise en œuvre de solutions efficaces en énergies.

Réseaux logiciel. Comme discuté dans les chapitres précédent, les technologies SDN permettent un meilleur contrôle des réseaux. Elles peuvent ainsi conduire à des économies d'énergie substantielles en rendant possible :

- une reconfiguration rapide du trafic par des modifications des tables de routage des nœuds

qui peut être directement changée par le contrôleur SDN, en se basant sur des données métrologiques et

- la virtualisation du réseau, car les clients peuvent pousser des règles pour leur trafic directement vers les nœuds, et la virtualisation des fonctions réseaux qui peuvent être instanciées à la demande.

SDN permet donc d'activer ou de désactiver des équipements réseau de façon dynamique, tout en contrôlant que tout se passe bien dans le réseau. Le contrôleur centralisé peut collecter la matrice du trafic puis calculer une solution de routage satisfaisant la QoS tout en minimisant la consommation d'énergie. Ensuite, le contrôleur met à jour les tables de routages des nœuds et désactive certaines interfaces réseau si possible afin d'économiser de l'énergie.

Virtualisation des fonctions réseaux (NFV). Dans les réseaux existants, les fonctions réseau, telles que pare-feu ou optimisation TCP, sont exécutées par du matériel spécifique. Dans les réseaux permettant la virtualisation couplée au paradigme des réseaux logiciels, les fonctions réseaux virtuelles peuvent être implémentées dynamiquement sur du matériel générique. Couplé au paradigme SDN, NFV apporte une grande flexibilité pour gérer les flux réseau. En effet, avec le contrôle centralisé autorisé par SDN, le flux peut être géré dynamiquement de bout en bout et les fonctions de service ne peuvent être installées que le long des chemins pour lesquels et quand ils sont nécessaires. Ces nouveaux paradigmes offrent donc la possibilité de réaliser des économies d'énergie dans les réseaux.

Objectif. Un des objectifs pour les prochaines années est donc d'explorer le potentiel des nouveaux paradigmes réseaux comme les réseaux logiciels et la virtualisation réseau, que ce soit les réseaux virtuels ou les fonctions réseaux virtuelles pour concevoir des solutions efficaces en énergie pour les réseaux de télécommunication et de centres de données.

Contributions. J'ai mené plusieurs travaux dans ce sens. Mes contributions sont à la fois théoriques et pratiques.

- J'ai tout d'abord étudié le problème de routage efficace en énergie (EAR) en prenant en compte les nouvelles contraintes de tailles de table de routage dues à l'utilisation de mémoire TCAM introduite dans le chapitre 6. J'ai étudié l'impact des politiques de routages multi-dimensionelles (prenant la décision de routage en utilisant plusieurs champs) sur les économies d'énergie potentielles (section 8.2).
- Puis je me suis intéressé à comment utiliser la technologie NFV pour améliorer l'efficacité des réseaux. J'ai en particulier proposé un modèle de décomposition résolu ensuite par génération de colonnes pour effectuer une optimisation jointe de l'EAR et du placement de fonctions réseaux virtuelles discuté dans le chapitre 7 (section 8.3).
- J'ai ensuite regardé comment mettre en œuvre progressivement l'EAR dans la section 8.4.
  - J'ai d'abord étudié l'introduction progressive des technologies SDN dans des réseaux SDN hybrides en section 8.4.2.
  - Puis nous avons testé nos solutions sur des plateformes logicielles (section 8.4.3) et matérielles (section 6.7). Nous avons pu vérifier que l'on pouvait mettre en œuvre des politiques de routage *dynamiques* efficaces en énergie tout en n'augmentant pas significativement les délais et en ne générant pas de surplus de pertes de paquets.

8.2 Utiliser les réseaux logiciels

Ce travail est le fruit d'une collaboration avec N. Huin, J. Moulierac et K. Phan. Il a donné lieu aux publications [Ci43, J10].

8.2.1 Introduction

SDN est un paradigme réseau en plein essor qui propose une gestion centralisée du réseau, contrairement aux réseaux décentralisés actuels. Les routeurs et les commutateurs deviennent de simples dispositifs de transfert tandis qu'un ou plusieurs contrôleurs s'occupent du travail de fond en calculant les chemins et en expliquant aux routeurs comment manipuler les paquets dans le réseau en utilisant le protocole OpenFlow [420]. Le remodelage du trafic est donc plus facile dans un réseau logiciel puisque les contrôleurs ont une connaissance totale de la topologie et de son utilisation. Cette flexibilité facilite le déploiement de politiques vertes sur le réseau. Le trafic peut être facilement agrégé sur un sous-ensemble du réseau avec des changements dans les tables de routage (aussi appelé table de transmission ou FIB en bref pour *Forwarding Information Base*) des commutateurs et les liens inutilisés peuvent alors être mis en veille. Cependant, nous avons vu dans le chapitre 6 que le nombre de règles de routage des switches SDN est très limités, de 750 à quelques milliers pour le matériel actuel.

Dans ce travail, nous utilisons des réseaux définis par logiciel pour déployer un routage efficace en énergie, EAR, qui achemine les demandes sur le réseau en respectant les *contraintes de capacité des liens et des tables de routage tout en minimisant la consommation d'énergie du réseau.* Nous utilisons des règles d'agrégation (ou wildcards ou règles joker) pour réduire la taille des tables de routage. Ces règles d'agrégation regroupent les règles avec la même action sur les champs correspondants. Nous étudions en particulier deux types de compression des tables de routage correspondant à deux type de règles d'agrégation : *la compression port par défaut*, qui utilise une règle joker routant tous les paquets vers un port par défaut, et *la compression multi-champs*, qui utilise des règles jokers supplémentaires agrégeant tous les flots avec un champ ayant une valeur spécifique (par exemple, tous les flots allant vers une destination spécifique). Nous nommons le problème considéré ici, Energy Aware Routing with Compression (EARC).

Nos contributions au problème EARC sont :

- A notre connaissance, ceci a été le premier travail qui définit et formule le problème d'optimisation de l'espace de règles dans SDN pour le routage efficace en énergie.
- Nous avons fourni des programmes linéaires en nombres entiers (ILP) pour résoudre de façon optimale EARC pour deux niveaux de compression de tables de routage : compression port par défaut et compression multi-champs dans [Ci43, J10].
- Comme EAR (et donc EARC) est connu pour être NP-hard [ch5], nous avons proposé des algorithmes heuristiques pour EARC qui sont efficaces pour les grandes topologies de réseau. L'algorithme comporte trois modules principaux : un module de compression chargé de compresser la table de routage, un module de routage chargé de trouver un itinéraire pour chaque demande répondant aux contraintes de capacité et un module d'énergie décidant quels liens réseau doivent être désactivés. En particulier, nous proposons plusieurs solutions au problème de compression.
- Nous avons comparé les différentes solutions du problème de compression. Nous les validons sur des tables de routage aléatoires ainsi que sur des tables réseau issues de simulations sur les réseaux de la librairie SDNlib [280].

- En utilisant les traces de trafic réelles de SNDlib, nous quantifions les économies d'énergie réalisées grâce à nos approches. De plus, nous présentons également d'autres aspects de QoS tels que la longueur des routes et la charge des liens pour des solutions EAR dans la section 8.2.3.

Dans cette section, nous ne présentons que les résultats pratiques sur les réseaux de SNDlib. Nos algorithmes de compression de tables de routage SDN sont présentés et discutés dans le chapitre 6.

8.2.2 Etat de l'art

Routage efficace en énergie avec SDN. Quelques travaux récents [149, 190, 137, 152] considèrent le problème de l'optimisation de la consommation d'énergie dans les réseaux logiciel en utilisant une approche d'ingénierie du trafic qui minimise le nombre de liens utilisés pour diminuer la consommation. Ils présentent des formulations du problème d'optimisation pour le routage des données et du trafic de contrôle. Ils évaluent leurs propositions par le biais de simulations. Ils assurent certaines contraintes de performance qui sont cruciales pour le bon fonctionnement des réseaux logiciels telles que le délai limité pour les données et un équilibre de charge entre les contrôleurs. [187] présente un état général de l'art des approches d'efficacité énergétique dans les réseaux définis par logiciel. Cependant, ces travaux n'abordent pas le problème de l'espace limité des règles.

8.2.3 Gains énergétiques

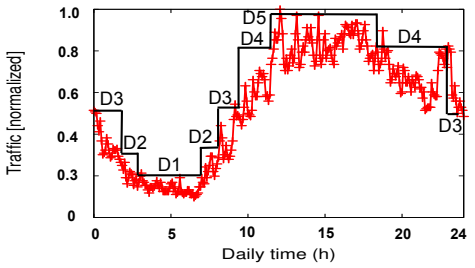
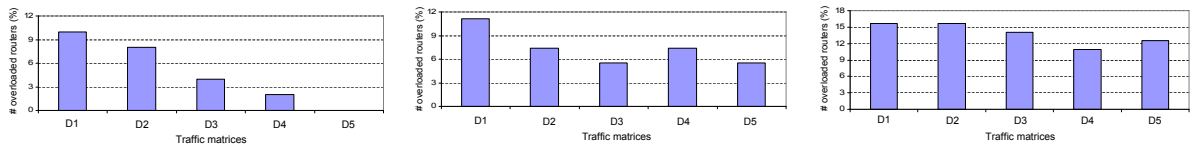


Figure 8.1: Trafic journalier d'un lien réseau et approximation multi-périodes.

Dans cette section, nous étudions l'énergie économisée sur plusieurs périodes de temps pour les quatre réseaux suivants : `atlanta`, `germany50`, `ta2` et `zib54`. Nous comparons les résultats obtenus pour les différentes solutions proposées pour résoudre le problème EARC, le problème EAR sans compression et le routage classique (CR) sans énergie. Les différentes solutions testées diffèrent par leur module de compression :

- EAR : routage efficace en énergie sans compression.
- EAR-with-limit : aussi sans compression, mais avec une limite sur le nombre de règles dans une table de routage.
- EARC-Default : compression port par défaut.
- EARC-Greedy : compression multi-champs avec une heuristique gloutonne qui ajoute itérativement la règle de compression sur les sources ou destinations qui élimine le plus de règles de routage.



(a) Germany50 network. (b) Zib54 network. (c) Ta2 network.

Figure 8.2: Nombre de routeurs dont la taille de la table de routage dépasse la limite du nombre de règles dans les trois réseaux considérés avec un algorithme ne considérant pas de limite pour le nombre de règles. Dans ces cas, le routage n'est pas valide.

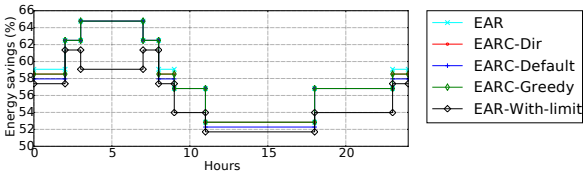
- EARC-Dir : compression multi-champs avec l'heuristique proposée dans la section 6.3 et qui est donne une solution 3-approchée du problème. Le principe est prendre la table de routage la plus petite parmi celles obtenues en utilisant (i) seulement la compression par source, (ii) seulement celle par destination et (iii) avec le port par défaut.

Pour les paramètres énergétiques, nous avons étudié la base de données powerlib [234] qui recueille des données représentatives pour les principaux équipements réseau tels que routeurs, commutateurs, transpondeurs... Dans cette base de données, la variabilité des valeurs de puissance maximale est énorme, allant, par exemple, de 10 W à 9000 W pour des composants de routeurs IP. Par conséquent, afin de présenter des résultats qui ne dépendent pas d'un équipement spécifique d'un fournisseur spécifique, nous choisissons pour les paramètres du modèle de puissance un modèle de puissance classique On-Off . Plusieurs autres articles utilisent de ce même modèle énergétique, entre autres, l'article le plus cité du domaine : [343].

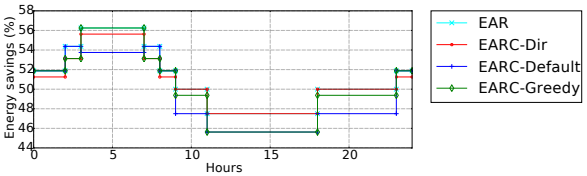
Sauf indication contraire, la limite de la table de routage est de 750 règles. Nous avons considéré un modèle typique de trafic quotidien, tel qu'illustré à la Figure 8.1. Les données proviennent d'un lien typique du réseau de Orange/France Télécom. Pour chaque réseau considéré, nous ajustons le trafic sur la base de la matrice de trafic fournie par SNDib. Nous divisons ensuite la journée en cinq périodes, avec différents niveaux de trafic comme le montre la Figure 8.1. D1 représente les heures creuses avec le moins de trafic sur le réseau et D5 les heures de pointe. Nous choisissons un petit nombre de périodes car les opérateurs de réseau préfèrent effectuer le moins possible de changements de configurations de leurs équipements de réseau afin de minimiser les risques d'introduire des erreurs ou de produire une instabilité de routage. De plus, la plupart des économies d'énergie peuvent être réalisées avec un très petit nombre de configurations, voir par exemple [J15]. Les économies d'énergie sont calculées en divisant le nombre de liens mis en veille par le nombre total de liens du réseau ($|E|$).

Besoin de plus de place. Dans la Figure 8.2, nous montrons le nombre de routeurs avec plus de 750 règles installées en appliquant l'heuristique proposée dans [ch5] pour le problème de routage efficace en énergie (EAR). Cette heuristique EAR ne prend pas en compte la contrainte de taille de la table. Par conséquent, nous constatons que pour presque tous les modèles de trafic (à l'exception de D5 sur l'Allemagne50), un EAR a besoin de plus de 750 règles pour être déployé. Pour `germany50`, jusqu'à 10% des appareils sont surchargés. Pour `zib54`, ce nombre va jusqu'à 11% et 16% pour `ta2`. Cela confirme que pour pouvoir déployer des politiques énergétiques sur un réseau logiciel, il faut résoudre ce problème de taille des tables de routage.

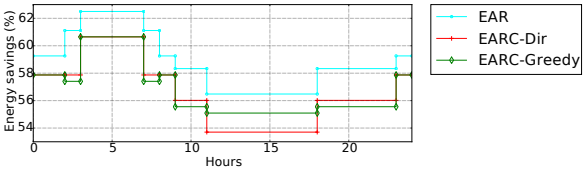
Gains énergétiques pendant une journée. Dans la figure 8.3, nous comparons les résultats donnés par les solutions proposées. Nous vérifions également la possibilité d'un routage SDN



(a) germany50



(b) zib54



(c) ta2

Figure 8.3: Gain énergétique pendant une journée pour les différentes heuristiques avec une limite de 750 règles.

sans compression (correspondant à un simple EAR). L'ILP n'est pas pris en compte dans la comparaison car les réseaux sont trop grands pour être résolus de manière optimale dans un délai acceptable.

Importance de la compression. Tout d'abord, nous constatons qu'à mesure que la taille des réseaux augmente, toutes les heuristiques ne donnent pas un routage SDN valide (i.e., certains switches auraient trop de règles). Aucune compression n'est nécessaire pour trouver un routage SDN valide sur `germany50`. Cependant, il est impossible de trouver un routage satisfaisant à la contrainte de capacité pour `zib54` et `ta2` sans utiliser un algorithme de compression. De plus, la compression multiple doit être utilisée pour trouver un routage valide pour `ta2`. En effet, il est impossible de trouver un routage valide pour `ta2` en utilisant seulement la compression par défaut.

Résultats des solutions proposées. Pour `germany50`, toutes les heuristiques donnent des résultats similaires entre 52% d'économie d'énergie pour les heures de pointe et jusqu'à 65% pendant la nuit. Les différences entre solutions sont petites, environ 2 %. Les heuristiques EARC-Greedy and EARC-Dir montrent les meilleurs résultats et aucune compression donne les pires pour toutes les périodes.

Pour le réseau `zib54`, la différence entre les heuristiques est un peu plus visible. Entre 46% et 56% est économisé pendant la journée. Une fois de plus, soit EARC-Greedy soit EARC-Dir donnent les meilleurs résultats en fonction des périodes. La seule exception est pendant les périodes D2, où la compression par défaut met en veille environ 1% plus de liens que les deux autres heuristiques.

Enfin, dans le réseau `ta2`, l'heuristique EARC-Greedy économise un peu plus d'énergie que EARC-Dir car la première économise presque 2% de plus que la seconde.

La quantité d'énergie économisée par les heuristiques pour chaque réseau est différente. L'explication est que l'ordre dans lequel chaque lien est éteint dépend de sa charge. Une petite modification de l'itinéraire peut donc affecter l'énergie totale économisée.

EAR vs. EARC. Nous comparons les résultats des solutions proposées avec l'approche classique de l'EAR dans laquelle aucune limite sur le nombre de règles n'est prise en compte. Nous montrons qu'en utilisant un moyen efficace pour acheminer les demandes et pour compresser les tables de routage, il est possible d'économiser presque autant d'énergie que l'approche EAR

dans la Figure 8.3). En effet, nous voyons que pour le réseau `zib54`, la meilleure solution réussit à économiser presque la même quantité d'énergie. Seulement un demi pour cent de l'énergie est perdu pour certaines périodes de temps. Pour `germany50`, les résultats des heuristique sont presque aussi bons. Pour certaines périodes de temps, aucune solution ne peut faire aussi bien que l'EAR, mais la différence n'est encore une fois que d'un demi pour cent. En général, les résultats de EARC-Greedy à 1% de ceux de EAR. Pour le réseau `ta2`, la différence entre EAR et nos solutions est plus élevée, mais reste à 2%.

8.2.4 Conclusion

A notre connaissance, il s'agit du premier travail prenant en compte les contraintes d'espace des règles d'un commutateur OpenFlow pour le routage efficace en l'énergie (EAR). Nous soutenons qu'en plus de la contrainte de capacité, l'espace des règles est également important car il peut changer la solution de routage et affecter la qualité de service. Nous avons proposé des solutions utilisant la compression de tables de routage, définissant le problème de routage énergétique avec compression (EARC) pour les réseaux SDN. Nous réussissons à modéliser le problème à l'aide de programmes linéaires entiers, même pour la compression complexe pour laquelle un flot peut être acheminé en fonction de deux champs de paquets. Nous fournissons également des algorithmes heuristiques efficaces pour les grands réseaux.

Basé sur des simulations avec des traces de trafic réelles, nous montrons que, en utilisant des règles joker (wildcard), notre allocation de règles intelligente peut atteindre une *efficacité énergétique élevée pour un réseau coeur tout en respectant les contraintes de capacité et d'espace de règles: Grâce à la compression des tables de routage, les économies d'énergie sont presque aussi élevées que dans le cas de l'EAR classique sans limite sur le nombre de règles de transfert.* Nous évaluons également l'impact des solutions proposées sur les délais dans [Ci43, J10]. Nous montrons que, si le délai est inévitablement augmenté, le *délai maximum* reste toujours en dessous des valeurs typiques données par les accords de niveau de service (Service Level Agreements).

8.3 Utiliser la virtualisation réseau

Ce travail est le fruit d'une collaboration avec N. Huin, *B. Jaumard* et A. Tomassilli et a donné lieu aux publications [Ci33, J6].

8.3.1 Introduction

Une des méthodes classiques pour réduire la consommation d'énergie des réseaux est d'essayer d'agréger le trafic réseau sur un petit nombre d'équipements réseau afin de mettre en veille le matériel inutilisé. Cependant, un défi supplémentaire est donné par le fait que le trafic d'aujourd'hui doit passer par un certain nombre de fonctions de réseau. Les fonctions réseau doivent souvent être appliquées dans un ordre spécifique, par exemple, dans un scénario de sécurité, le pare-feu doit être appliqué avant d'effectuer un DPI, car ce dernier est plus intensif en CPU que le premier. Ces fonctions sont traditionnellement exécutées par du matériel spécifique, qui est installé à des endroits spécifiques du réseau. Les trajets suivis par les demandes sont donc très limités, ce qui réduit les possibilités d'agréger le trafic.

Avec l'émergence des techniques de virtualisation des fonctions réseau (NFV), les fonctions peuvent maintenant être exécutées par du matériel générique au lieu d'équipements dédiés. Couplé au paradigme SDN, NFV apporte une grande flexibilité pour gérer les flots réseau. En effet, avec le contrôle centralisé autorisé par SDN, les flots peuvent être gérés dynamiquement de bout en bout et les fonctions de services ne peuvent être installées que le long des chemins pour lesquels et quand elles sont nécessaires. Ces nouveaux paradigmes offrent donc la possibilité de réaliser des économies d'énergie dans les réseaux.

Dans ce travail, nous explorons les économies d'énergie potentielles de l'utilisation du paradigme NFV pour le provisionnement des chaînes de service. Une difficulté est que les fonctions du réseau doivent être exécutées dans un ordre spécifique et peuvent être répétées plusieurs fois dans la même chaîne.
En résumé, les contributions de ce travail sont les suivantes :

- Nous montrons comment la virtualisation peut être utilisée pour améliorer l'efficacité énergétique des réseaux, lorsque les demandes doivent passer par une chaîne de services. A notre connaissance, nous sommes les premiers à proposer une telle méthode.

- Nous proposons une façon de modéliser ce problème basée sur la Programmation Linéaire en Nombres Entiers (ILP). L'ILP peut résoudre de manière optimale les instances de petites tailles.

- Pour gérer des instances de plus grandes tailles, nous proposons et validons un *algorithme heuristique*, GreenChains, et nous formulons un *modèle de génération de colonnes.*

- Nous fournissons des améliorations du modèle avec l'utilisation de coupes, puisque notre problème est un problème d'optimisation difficile. En effet, il contient un phénomène On-Off aigu, car un appareil réseau consomme une grande partie de son énergie dès qu'il est utilisé, même s'il est très peu utilisé. Les coupes permettent de réduire l'écart d'intégralité.

- Cela nous permet d'effectuer des simulations étendues sur des réseaux de différentes tailles. Nous étudions trois scénarios différents : un *scénario classique* qui sert de base de comparaison, un *scénario matériel* dans lequel le routage peut être modifié dynamiquement par un contrôleur SDN centralisé, mais dans lequel les fonctions réseau sont exécutées par du matériel spécifique, et enfin, un *scénario NFV* dans lequel les fonctions réseau

sont virtualisées et peuvent être placées dynamiquement. Nous montrons que de 22 à 62% d'énergie peut être économisée pendant la nuit tout en respectant les contraintes des chaînes de services.

– Enfin, nous proposons une analyse de latence pour évaluer l'impact sur les retards de l'extinction de certains éléments du réseau pour économiser de l'énergie.

La section est organisée comme suit. Dans la section 8.3.2, nous passons en revue les travaux en cours sur l'efficacité énergétique et les chaînes de fonctions de services. Le problème est présenté dans la section 8.3.3 avec le modèle de puissance et le modèle de graphe en couches utilisés dans nos formulations mathématiques. Nous présentons dans la section 8.3.4 la formulation ILP, GreenChains et le schéma de génération de colonnes, respectivement. Nous comparons ensuite les modèles et évaluons leur qualité dans Section 8.3.5.

8.3.2 Etat de l'art

SDN et efficacité énergétique des réseaux. Récemment, les chercheurs ont commencé à explorer comment l'introduction du paradigme SDN avec un contrôle centralisé et une connaissance en direct des données de métrologie peut permettre un routage dynamique. J'ai été un des premiers à m'intéresser à cette thématique. En particulier, j'ai montré qu'il permettait la mise en œuvre d'algorithmes de routage efficaces en énergie dans la section précédente, la section 8.2. Toutefois, je ne prends pas en compte les contraintes liées aux fonctions du réseau. Certains travaux particuliers ont considéré certaines classes spécifiques de fonctions de réseau, comme par exemple l'élimination de redondance dans mes travaux de la section 3.2 du chapitre 8, mais pas le problème général de s'assurer que les flux soient traités par les fonctions de réseau.
Virtualisation du réseau et efficacité énergétique du réseau. Seuls deux articles exploraient le potentiel de la virtualisation des réseaux pour l'efficacité énergétique quand nous avons proposé ce travail. Dans Bolla *et al.* [204], les auteurs présentent une extension d'un framework logiciel open source, Distributed Router Open Platform (DROP), pour permettre un nouveau paradigme distribué pour NFV. DROP comprend des mécanismes sophistiqués de gestion de l'énergie, qui sont exposés dans un Green Abstraction Layer. Dans [166], les auteurs estiment les économies d'énergie qui pourraient résulter des trois principaux cas d'utilisation de NFV, Virtualized Evolved Packet Core, Virtualized Customer Premises Equipment et Virtualized Radio Access Network. Toutefois, les deux documents ne tiennent pas compte des contraintes des chaînes de services. Un autre travail récent [108] étudie l'efficacité énergétique avec ces contraintes d'ordre.

This paper proposes an Integer Linear Program (ILP) to address Virtualized Network Function Forwarding Graph (VNF-FG) placement and chaining with Virtualized Network Functions (VNFs) shared across tenants to optimize resource usage and increase provider revenue.

8.3.3 Enoncé du problème : placement de SFC et VNF

Notations.

Nous supposons que le réseau est représenté par un graphe dirigé $G = (V, L)$, où V est l'ensemble des nœuds (indexés par v), et L est l'ensemble des liens (indexés par ℓ). Chaque nœud $u \in V$ possède un ensemble de ressources de calcul, de stockage et de réseau, désigné par C_u pour

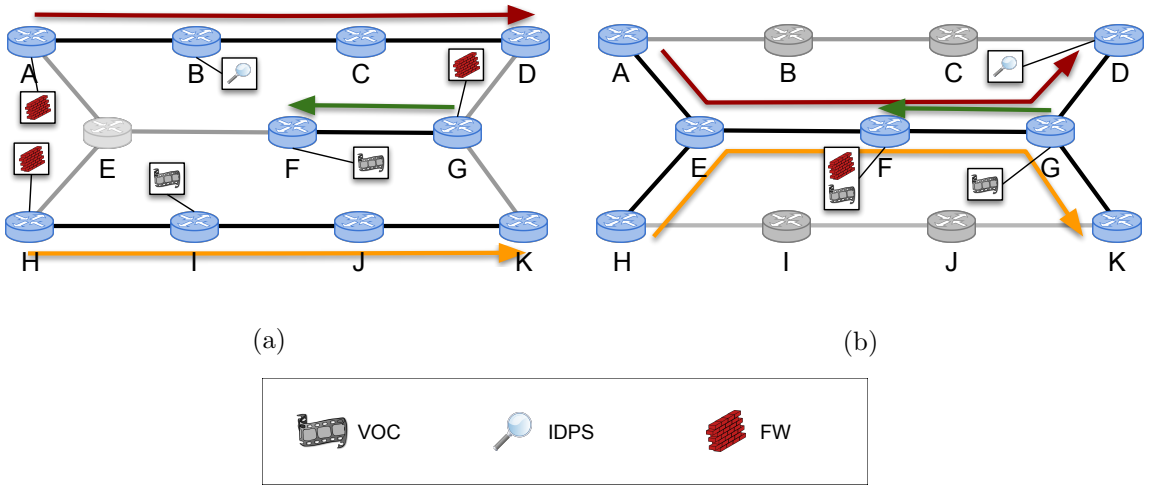


Figure 8.4: Exemple de placement de chaînes de fonctions de services efficace en énergie. Les liens et nœuds en gris sont inactifs.

héberger les fonctions de réseau. Dans le cadre de cette étude, nous supposons que l'unique ressource est décrite par un nombre donné de cœurs de CPU.

Le trafic est décrit par un ensemble de requêtes D, dans lequel chaque requête d est définie par un quadruplet (v_s, v_d, c, D^c_{sd}), où v_s est la source de la requête, v_d sa destination, D^c_{sd} ses besoins en bande passante, et c la chaîne de service demandée. En effet, chaque demande d est associée à une application donnée, qui doit passer par une SFC donnée. Soit F l'ensemble des fonctions virtuelles apparaissant dans les chaînes de services et C l'ensemble des chaînes de services. Chaque chaîne de services c correspond à une séquence de n_c fonctions. $f^c_1, \ldots, f^c_i, \ldots, f^c$, où f^c_i désigne la i-ème fonction de la chaîne c. Notez que certaines fonctions peuvent apparaître plus d'une fois dans une chaîne donnée, et que pour diverses raisons (par exemple, résilience, confidentialité), deux fonctions différentes f et f' peuvent ne pas pouvoir être installées dans le même nœud. Chaque fonction virtuelle f a ses propres besoins en ressources, et nous dénotons par Δ_f le nombre (fraction) de cœurs requis par la fonction f par unité de bande passante.

Le problème *Energy Efficient Service Function Provisioning* (EE-SFCP) consiste à provisionner conjointement un ensemble D de requêtes couplées à des chaînes de fonctions de services C et à placer les fonctions virtuelles apparaissant dans les chaînes, afin de minimiser la consommation d'énergie du réseau, sous réserve des capacités des liens et des nœuds.

La figure 8.4 montre des exemples du problème de placement de chaînes de fonctions de services éco-énergétique. Nous avons trois demandes et deux types de services. La demande $D_1 = (A \rightsquigarrow D)$ nécessite 1 unité de bande passante et l'exécution d'un pare-feu (FW) et d'un inspecteur de paquets (IDPS). Les demandes $D_2 = (H \rightsquigarrow K)$ et $D_3 = (G \rightsquigarrow F)$ ont besoin d'une unité de bande passante, et l'exécution d'un pare-feu suivi d'un optimiseur vidéo (VOC). Une instance d'un pare-feu utilise $\Delta_{\text{FW}} = 0,33$ par unité de bande passante, une instance d'un optimiseur vidéo nécessite $\Delta_{\text{VOC}} = 1$ cœur, et une instance d'un inspecteur de paquets utilise $\Delta_{\text{FW}} = 2$ cœurs. Chaque liaison a une capacité de 3 unités de bande passante, et chaque nœud héberge 2 cœurs.

Dans la figure 8.4a, les flots sont acheminés selon le le plus court chemin entre leur source et leur destination. Nous devons placer une instance de fonction de pare-feu sur trois nœuds différents (à savoir A, G et H) pour couvrir les trois demandes. Les flots D_2 et D_3 ont leur optimiseur vidéo sur les nœuds F et I, respectivement. L'inspecteur de paquets du flot D_1 est

installé sur le nœud B. Dans cette dernière configuration, cinq liens $((A,E), (H,E), (E,F), (G,D), (G,D), (G,K))$ et le nœud E peuvent être mis en veille. Un total de 7 cœurs sont actifs (1 sur les nœuds A, F, G, G, H, H et I, et deux sur le nœud B).

Cependant, il existe un autre itinéraire qui minimise l'énergie consommée par le réseau. Il permet la mise en veille d'un lien supplémentaire et la réduction du nombre de cœurs actifs de deux unités. En effet, si l'on considère le routage donné dans la figure 8.4b, on peut regrouper toutes les instances de pare-feu sur le nœud F. Puisque les nœuds n'hébergent que 2 cœurs, nous devons mettre une instance de l'optimiseur vidéo sur le nœud F, en charge de D_2 ; D_3 est servi par l'instance sur le nœud G. Nous utilisons maintenant seulement 5 cœurs au total, et nous pouvons maintenant mettre en veille les liens (A,B), (B,C), (B,C), (C,D), (H,I), (I,J), (I,J) et (J,K).

Modèle énergétique

Des campagnes de mesures de consommation d'énergie (voir, par exemple, [344]) montrent qu'un périphérique réseau consomme une grande quantité d'énergie dès qu'il est allumé et que la consommation d'énergie ne dépend pas beaucoup de la charge. Suite à cette observation, des modèles de puissance on/off ont été proposés et étudiés, cf les sections 8.2 et 3.1 du chapitre 8 de ce document par exemple. Plus tard, les chercheurs et les constructeurs de matériel ont proposé des modèles de matériel proportionnel à l'énergie [239]. Pour englober ces différents modèles, nous utilisons un modèle de puissance hybride dans lequel la puissance d'un lien actif ℓ est exprimée par un modèle affine comme suit

$$P_\ell = P_\ell^{\text{ON}} + \frac{\text{BW}_\ell}{C_\ell^{\text{LINK}}} P_\ell^{\text{MAX}},$$

où P_ℓ^{ON} représente l'énergie utilisée lorsque le lien ℓ est activé, BW_ℓ la bande passante qui est transportée sur ℓ, et P_ℓ^{MAX} l'énergie supplémentaire consommée par ℓ lorsqu'il est à pleine capacité, c'est-à-dire lorsque la quantité de bande passante transportée est égale à la capacité de transport (C_ℓ^{LINK}) du lien ℓ.
Nous supposons que les liens peuvent être mis en mode veille, en mettant en veille les deux interfaces terminales. Les deux liens en sens opposés entre une paire de nœuds sont supposées être dans le même état (actif ou en mode veille), car les éléments d'émission et de réception d'une fibre unidirectionnelle sont généralement contrôlés par la même interface. Les routeurs ne peuvent pas être mis en mode veille, car ce sont les sources/destinations du trafic réseau. Cependant, les cœurs peuvent être mis en mode veille et l'énergie utilisée par le nœud v est égale à

$$P_v = P_v^{\text{UNIT}} \times \#\text{cores}$$

avec P_v^{UNIT} la consommation énergétique d'un cœur.

8.3.4 Méthodes de résolution

Pour résoudre le problème, nous proposons plusieurs méthodes dans [Ci33, J6]. Nous ne détaillons ici que le modèle de décomposition, et discutons seulement succinctement les autres méthodes. Nous présentons le modèle de décomposition, même s'il présente des similitudes avec celui de Section 7.1 du chapitre 6 développé pour minimiser la bande passante et non l'énergie. En effet, le problème ici est beaucoup plus difficile, ce qui entraîne des écarts d'intégralité plus importants. Nous introduisons donc des techniques pour réduire cet écart.

Nous développons d'abord une formulation ILP pour notre problème. Nous utilisons un graphe en couches G^{L}, comme expliqué dans la section 7.1.3 du chapitre 6. Cela permet de

modéliser le problème de placement des ressources virtuelles en tant que problème de routage. Comme l'ILP proposé ne peut pas fournir de solutions pour les grands réseaux, nous proposons ensuite un algorithme heuristique basé sur l'ILP appelé GreenChains pour résoudre le problème. Celui-ci peut être décomposé en trois sous-problèmes.

- Tout d'abord, le *problème d'économie d'énergie* essaie de mettre en mode veille autant de liens et de cœurs que possible pour diminuer la consommation d'énergie du réseau.

- Deuxièmement, le *problème de routage* calcule un chemin pour chaque requête, en respectant les contraintes de capacité des liens.

- Enfin, le but du *problème de placement de la chaîne de service* est de trouver un placement des NVF respectant les capacités des nœuds et l'ordre défini par les chaînes de service, selon le chemin calculé pour chaque requête.

GreenChains résoud les deux premiers sous-problèmes de façon similaire à l'heuristique Less Loaded Edge Heuristic proposée dans le chapitre 8. Il essaie itérativement de mettre en veille un lien peu utilisé, puis de trouver un routage. Le placement de chaînes de services alors résolu (si une solution existe) en utilisant un ILP. L'algorithme est détaillé dans [J6].

Modèles de décomposition

Comme l'ILP ne passe pas à l'échelle, nous proposons un schéma de génération de colonnes pour aider à valider notre heuristique pour les grands réseaux. Nous présentons d'abord ici un modèle utilisant la Génération de Colonne, *CG-simple*. Nous introduisons ensuite deux variantes des modèles, *CG-cuts*, et *CG-cut+*. En effet, les problèmes liés à l'efficacité énergétique conduisent souvent à un grand écart d'intégralité et à une mauvaise précision. Ceci est dû au phénomène On-Off des modèles de puissance, qui se traduit par de grandes marches dans la fonction objectif. Nous essayons donc d'améliorer la précision du modèle en introduisant différents ensembles de contraintes. Nous discutons de la précision des modèles dans la section 8.3.5.

Formulation en génération de colonnes Nous proposons une formulation de génération de colonnes qui repose sur le concept de *Service Path* : chaque *Service Path* p est associé à un 4-uplet (v_s, v_d, c, D^c_{sd}) et définit : *(i)* une route potentielle pour la requête (v_s, v_d, c, D^c_{sd}) entre v_s et v_d, *(ii)* un placement sur des nœuds des fonctions de la chaîne c le long de la route potentielle. Un chemin est décrit par les paramètres δ^p_ℓ, égaux au nombre d'occurrences du lien ℓ dans le chemin p. Le placement des nœuds est donné par a^p_{vi}, égal à 1 si la i^e fonction de la chaîne c est située au nœud v, et sinon 0. Nous dénotons par P^c_{sd} l'ensemble global de *Service Path* pour chaque requête (v_s, v_d, c, D^c_{sd}).

Nous définissons maintenant l'ensemble des variables. Premier ensemble de variables de décision : $x_\ell = 1$ si le lien ℓ est activé (actif), 0 sinon. Notez que les liens sont éteints par paire, c'est-à-dire $x_{\ell=(v,v')} = x_{\ell'=(v',v',v)}$. Deuxième ensemble de variables de décision : $y^p_d = 1$ si la demande d est routée en utilisant la configuration p, 0 sinon. Variables entières : $k_v = \#$ cœurs requis dans le nœud v.

L'objectif, c'est-à-dire la minimisation de l'énergie, peut être écrit

$$\min \quad \underbrace{\sum_{\ell \in L} P_\ell^{\text{ON}} x_\ell}_{\substack{\text{énergie de mise} \\ \text{en route}}} + \underbrace{\sum_{\ell \in L} \sum_{p \in P_{sd}^c} \delta_\ell^p \left(\sum_{d=(v_s,v_d,c) \in D} \frac{D_{sd}^c}{C_\ell^{\text{LINK}}} P_\ell^{\max} \right) y_d^p}_{\text{énergie bande passante}} + \underbrace{\sum_{u \in V} P_v \, k_v}_{\text{énergie des noeuds}} \tag{8.1}$$

Les contraintes se divisent en trois ensembles de contraintes.

Un chemin par demande

$$\sum_{p \in P_{sd}^c} y_d^p = 1 \qquad (u_s, u_d) \in \mathcal{SD}, c \in C_{sd} \in D. \tag{8.2}$$

Capacité des liens

$$\sum_{d=(v_s,v_d,c) \in D} \sum_{p \in P_{sd}^c} D_{sd}^c \, \delta_\ell^p \, y_d^p \leq x_\ell \, C_\ell^{\text{LINK}} \qquad \ell \in L. \tag{8.3}$$

Capacité des nœuds

$$\sum_{d \in D} \sum_{p \in P_{sd}^c} D_{sd}^c (\sum_{i=1}^{n_c} \Delta_{f_i} a_{v f_i}^p) y_d^p \leq k_v \leq C_v^{\text{NODE}}$$

$$u \in V. \tag{8.4}$$

Comme nous avons fait face à des problèmes avec de grands écarts d'intégrité, nous avons amélioré le modèle(8.1)-(8.4) avec différents ensembles de coupes, dans les deux modèles suivants.

Modèle *CG-cuts*. La première série de coupes dans (8.5) indique que, pour chaque noeud, au moins un lien incident doit toujours être activé. De plus, la deuxième inégalité donnée par l'équation (8.6) impose qu'au moins $n-1$ liens doivent être actifs pour avoir un réseau connecté (ou différent si pas all-to-all).

$$\sum_{\ell \in \omega^+(v)} x_\ell \geq 1 \qquad u \in V \tag{8.5}$$

$$\sum_{\ell \in L} x_\ell \geq n - 1. \tag{8.6}$$

Modèle *CG-cut+*. Nous améliorons à nouveau le modèle *CG-cuts* avec :

$$x_\ell \geq \sum_{p \in P_{sd}^c} \gamma_\ell^p \, y_d^p \qquad \ell \in L, (u_s, u_d) \in \mathcal{SD}, c \in C_{sd} \tag{8.7}$$

où $\gamma_\ell^p = 1$ si le lien ℓ appartient au chemin p. En utilisant (8.2), on obtient que $\sum_{p \in P_{sd}^c} \gamma_\ell^p \, y_d^p \leq 1$. Cela évite l'utilisation d'une formulation avec un "big M" au prix d'un plus grand nombre de contraintes.

Schéma de solutions Pour résoudre efficacement le modèle de la section 8.3.4, il faut recourir à la génération de colonnes pour résoudre la relaxation linéaire, puis dériver une solution entière,

en utilisant le dernier problème maître restreint (RMP pour *Restricted Master Problem*). L'idée est la suivante. Le RMP est résolu (en fractionnaire) avec un sous-ensemble des colonnes. Nous cherchons ensuite une colonne à ajouter à l'aide du problème de *pricing* discuté ci-dessous. Si une colonne qui améliore la solution du RMP existe, nous l'introduisons. Sinon, la solution fractionnaire trouvée est optimale. La dernière étape est de résoudre le RMP en entier. Pour plus de détails sur la programmation linéaire et les schémas de génération de colonnes, voir, par exemple, [100].

Nous avons un problème de pricing, pour chaque requête (v_s, v_d, c, D^c_{sd}). Son objectif est défini par la minimisation du coût réduit défini à partir des valeurs des variables duales du problème maître. Le but est de sélectionner le service path qui améliore le plus l'objectif du RMP. Nous notons $u(j)$ le vecteur des variables duales correspondant aux contraintes (j) du problème maître. Deux ensembles de variables de décision sont ensuite nécessaires. Le premier ensemble est composé de variables φ^i_ℓ telles que $\varphi^i_\ell = 1$ si le positionnement de la demande d utilise le lien ℓ dans la couche i du graphe en couches G^{L}, et sinon 0. Le second ensemble contient des variables a^i_v telles que $a^i_v = 1$ si la i-e fonction (f^c_i) de la chaîne c pour la requête (v_s, v_d, c, D^c_{sd}) est placée sur le nœud NFV v, et 0 sinon. La formulation du générateur de *Service Path* est donnée comme suit.

$$\min -u^{(8.2)}_{sd} + \sum_{\ell \in L} \sum_{i=0}^{n_c} \varphi^i_\ell \times \left(P^{\max}_\ell \frac{D^c_{sd}}{{C^{\text{LINK}}}_\ell} + u^{(8.3)}_\ell D^c_{sd} \right) + \sum_{u \in V} \sum_{i=0}^{n_c - 1} a^i_v \times \left(u^{(8.4)}_v \Delta_{f_i} D^c_{sd} \right). \quad (8.8)$$

Calcul des chemins (contraintes de conservation des flots) :

$$\sum_{\ell \in \omega^+(v)} \varphi^i_\ell - \sum_{\ell \in \omega^-(v)} \varphi^i_\ell + a^i_v - a^{i-1}_v = 0 \qquad u \in V, 0 < i < n^c \quad (8.9)$$

$$\sum_{\ell \in \omega^+(v)} \varphi^0_\ell - \sum_{\ell \in \omega^-(v)} \varphi^0_\ell + a^0_v = \begin{cases} 1 \text{ if } v = v_s \\ 0 \text{ sinon} \end{cases} \qquad u \in V \quad (8.10)$$

$$\sum_{\ell \in \omega^+(v)} \varphi^{n_c}_\ell - \sum_{\ell \in \omega^-(v)} \varphi^{n_c}_\ell - a^{n_c}_v = \begin{cases} -1 \text{ if } v = v_d \\ 0 \text{ sinon} \end{cases} \qquad u \in V. \quad (8.11)$$

Capacité des liens : $D^c_{sd} \sum_{i=0}^{n_c} \varphi^i_\ell \leq C^{\text{LINK}}_\ell \qquad \ell \in L. \quad (8.12)$

Capacité des noeuds : $D^c_{sd} \sum_{i=0}^{n_c} \Delta_{f_i} a^i_v \leq C^{\text{NODE}}_v \quad u \in V. \quad (8.13)$

Accélérer le problème de pricing Le problème de pricing correspond à un plus court chemin contraint avec des poids négatifs sur le graphe en couches, et nous pouvons utiliser CPLEX pour le résoudre. Cependant, si nous écartons les contraintes de capacité, le problème

devient un problème de *plus court chemin avec poids négatifs*. Il peut être résolu beaucoup plus rapidement que le problème original en utilisant l'algorithme de plus court chemin de Bellman-Ford. Puisque nous éliminons les contraintes de capacité, l'ensemble des solutions considérées est un sur-ensemble de l'ensemble initial de solutions. Il est possible de trouver un chemin qui pourrait utiliser plus de ressources que ce qui est disponible. Dans ce cas, on se replie sur le solveur ILP pour obtenir un chemin valide. Le poids des arcs à l'intérieur des couches est ainsi donné par

$$w_{iv} = P_\ell^{\max} \frac{D_{sd}^c}{C^{\text{LINK}}{}_\ell} + u_\ell^{(8.3)} D_{sd}^c \qquad 0 \leq i < n_c, u \in V.$$

et le poids des arcs entre couches par

$$w_{i\ell} = u_v^{(8.4)} \Delta_{f_i} D_{sd}^c \qquad 0 \leq i \leq n_c, \ell \in L.$$

Particularités de *CG-cut+* En introduisant les contraintes (8.7) dans le modèle, nous devons également introduire un nouvel ensemble de variables γ_l dans le problème de pricing qui indique si le lien ℓ est utilisé dans le chemin. La fonction objectif devient

$$\begin{aligned} \min \quad -u_{sd}^{(8.2)} + \sum_{\ell \in L} \sum_{i=0}^{n_c} \varphi_\ell^i \times \left(P_\ell^{\max} \frac{D_{sd}^c}{C^{\text{LINK}}{}_\ell} + u_\ell^{(8.3)} D_{sd}^c \right) \\ + \sum_{u \in V} \sum_{i=0}^{n_c} a_v^i \times \left(u_v^{(8.4)} \Delta_{f_i} D_{sd}^c \right) \\ + \sum_{\ell \in L} \gamma_l u_{sd,c,l}^{(8.7)}. \end{aligned} \tag{8.14}$$

et les contraintes de capacité des liens deviennent

$$D_{sd}^c \sum_{i=0}^{n_c} \varphi_\ell^i \leq C_\ell^{\text{LINK}} \times \gamma_\ell \qquad \ell \in L. \tag{8.15}$$

De plus, l'ajout de coupes améliorées crée des cycles négatifs dans le graphe en couches utilisé pour le problème de prix. Nous choisissons de ne pas nous débarrasser des cycles en les énumérant tous. Au lieu de cela, nous vérifions si la solution fournie par le solveur contient des cycles négatifs. Si c'est le cas, nous ajoutons les contraintes correspondantes dans la formulation et rappelons le solveur. Nous répétons ce processus jusqu'à ce que la solution obtenue ne contienne plus de cycles négatifs ou que le coût réduit ne soit plus négatif. La suppression du cycle a un impact négligeable sur les performances du schéma de génération de colonnes car il n'est exécuté que quelques fois au début de l'algorithme.

8.3.5 Expérimentations numériques

Dans cette section, nous examinons les économies d'énergie obtenues par le modèle de génération de colonnes. Nous comparons les résultats avec ceux de l'algorithme heuristique GreenChains. Nous présentons d'abord les ensembles de données que nous utilisons pour les expériences. Nous examinons ensuite la précision des solutions obtenues par le modèle de génération de colonnes (CG) et GreenChains. Nous étudions différentes améliorations du modèle présenté dans la section 8.3.3. Nous présentons ensuite les économies d'énergie réalisées pour des topologies de réseaux de différentes tailles. Enfin, nous discutons de l'impact des solutions sur l'utilisation des liens et la longueur des chemins.

Service Chains	Chained VNFs	Rate	traffic (%)
Web Service	NAT-FW-TM-WOC-IDPS	100 kbps	18.2
VoIP	NAT-FW-TM-FW-NAT	64 kbps	11.8
Video Streaming	NAT-FW-TM-VOC-IDPS	4 Mbps	69.9
Online Gaming	NAT-FW-VOC-WOC-IDPS	50 kbps	0.1

Table 8.1: Caractéristiques des chaines de fonctions de services [158].

Jeux de données

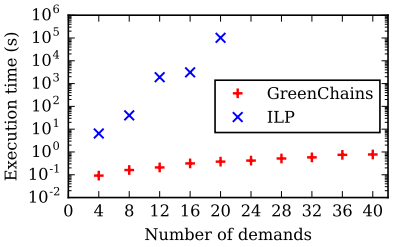
Dans les réseaux, chaque type de flots doit passer par une chaîne différente de services réseau. Dans nos expériences, nous considérons quatre des types de flots les plus fréquents, tels que présentés dans le tableau 8.1 : Streaming vidéo, service Web, voix sur IP (VoIP) et jeux en ligne. Les pourcentages de trafic sont tirés de [164]. Pour chacun d'eux, nous donnons l'ensemble ordonné des fonctions requises et la bande passante utilisée. Au total, nous avons six fonctions différentes, et chaque fonction nécessite un nombre différent de cœurs à exécuter.

Nous avons testé les modèles CG et GreenChains sur trois topologies de tailles différentes de SNDlib [280] : *pdh* (11 nœuds et 64 liens dirigés), *atlanta* (15 nœuds et 44 liens dirigés), et *germany50*. (50 nœuds et 176 liens dirigés).

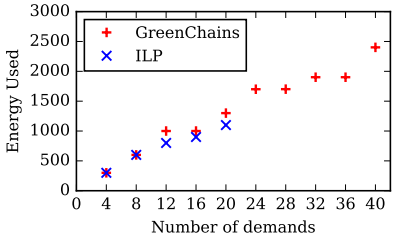
Pour obtenir des matrices de trafic réalistes, nous générons, pour chaque réseau, un ensemble de demandes à partir des matrices de trafic fournies dans SNDlib : nous divisons chaque flux d'agrégats d'une source à une destination en quatre demandes correspondant aux quatre types de trafic différents. Nous considérons que les flux fournis dans l'ensemble de données SNDlib représentent des flots agrégés de divers services. Ainsi, nous pouvons les subdiviser en quatre services différents. La charge originale du flot est conservée, et la charge de chaque sous-flot est donnée par la distribution de la dernière colonne du Tableau 8.1. Par exemple, un flot avec une charge de 1 est divisé en un service Web, un service VoIP, un sous-flot de streaming vidéo et un de jeux en ligne avec des charges de 0,182, 0,118, 0,699 et 0,001, respectivement. Cela correspond a un mix de trafic réaliste [158].

Nous avons testé la solution sur un trafic journalier (Figure **??**) pour voir combien d'énergie peut être économisée pendant la journée ou la nuit. Les travaux antérieurs (cf la discussion de la section 3.3.1 du chapitre 8 et [J15]) indiquent que l'utilisation d'un petit nombre de configurations au cours de la journée suffit pour obtenir la plupart des économies d'énergie. Dans notre cas, nous avons pris en compte cinq niveaux de trafic différents appelés de D1 à D5. D1 représente la période où le trafic est le plus faible et D5 celle où le trafic est le plus élevé.

Traditionnellement, les réseaux de FAI utilisent le chemin le plus court et exploitent leur réseau avec un facteur de sur-dimensionnement de 2 ou 3 [373, 369], afin de pouvoir faire face aux pannes et à la croissance du trafic. Cela signifie que les liens ne sont généralement utilisés qu'entre 30 et 50 % de leur capacité. Nous avons fixé les capacités en conséquence au début de la simulation. Pour chaque réseau, nous résolvons le scénario traditionnel (legacy) en acheminant les demandes sur les plus courts chemins entre chaque emplacement des fonctions de services. Chaque emplacement de fonction est choisi au hasard dans le scénario traditionnel. Nous choisissons ensuite les capacités de liens de telle sorte que chaque lien soit utilisé au maximum à 33 % de sa capacité. Enfin, nous avons considéré des valeurs égales pour la consommation énergique



(a) Temps d'exécution



(b) Consommation énergétique

Figure 8.5: Comparaison entre la formulation compacte et GreenChains.

des liens et des nœuds.

Évaluation de la formulation compacte

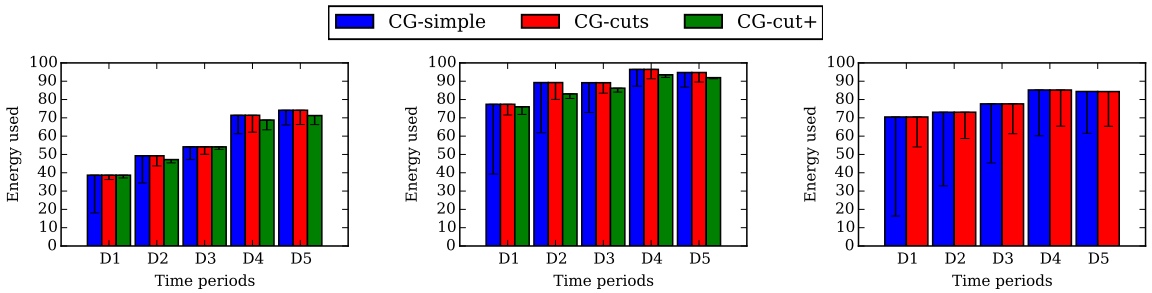
Nous comparons les résultats obtenus par l'algorithme heuristique, GreenChains, avec les résultats optimaux donnés par le programme linéaire en nombres entiers sur un petit réseau, *pdh*, avec 11 nœuds et 64 liens. Nous considérons des cas de plus en plus complexes : le nombre de demandes varie de 4 à 40. Notez que nous considérons des multiples de 4 pour les demandes, car le trafic entre une paire de nœuds est divisé en quatre demandes différentes correspond à différentes catégories de trafic.

Nous comparons les temps d'exécution du modèle ILP et de l'algorithme dans la figure 8.5a. Les expériences sont faites sur un Intel Xeon E5620 avec 24 Go de RAM. Nous voyons que le modèle ILP peut être utilisé pour résoudre le problème en un temps raisonnable pour un nombre maximum de 16 demandes. Dans ce cas, il faut environ 45 minutes pour retourner la solution optimale. L'augmentation du temps est ensuite exponentielle : pour 20 demandes, le temps d'exécution est de près de 3 heures. D'autre part, GreenChains est beaucoup plus rapide car il peut trouver une solution en moins d'une seconde pour 20 demandes (0.38 s). Il résout une instance avec 40 demandes en 0.78 s et l'instance all-to-all (avec 440 demandes), considérée dans ce qui suit, en moins de 7 s. Nous voyons que l'ILP ne peut pas être utilisé dans la pratique pour résoudre des instances avec un grand nombre de demandes, et nous utilisons donc GreenChains pour les expériences sur des réseaux plus grands dans ce qui suit.

Les résultats concernant les économies d'énergie sont donnés dans la Figure 8.5. GreenChains trouve des résultats avec une précision (écart maximum à l'optimal) comprise entre 0% et 16% pour des nombres de demandes différents. Nous considérons qu'il s'agit là de bons résultats étant donnée la difficulté du problème EE-SFCP.

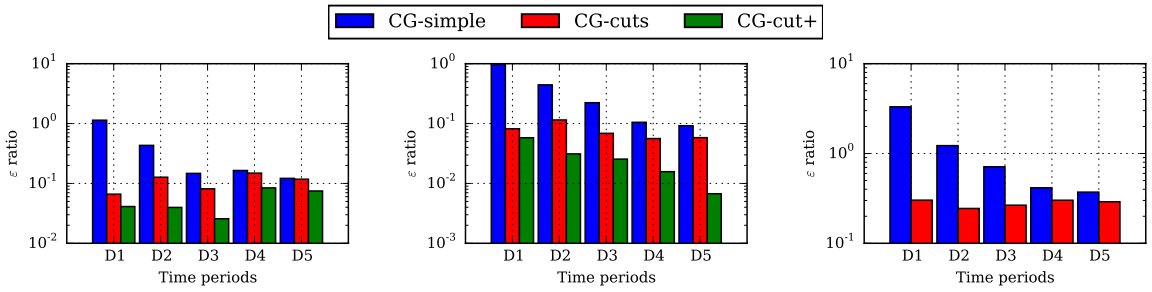
Qualité des modèles de génération de colonnes CG

Nous comparons maintenant la performance des trois modèles différents de CG (*CG-simple*, *CG-cuts*, *CG-cuts*, et *CG-cut+*) en ce qui concerne leur précision telle que donnée par $\varepsilon = (\tilde{z}_{\text{ILP}} - \tilde{z}_{\text{ILP}} - z^{\star}_{\text{LP}})/z^{\star}_{\text{LP}}$, où $z^{\star}_{\text{LP}}$ représente la valeur optimale de la relaxation du problème maître restreint, et $\tilde{z}_{\text{ILP}}$ la solution entière obtenue à la fin de l'algorithme de génération de colonnes. Dans les figures 8.6, 8.7, et 8.7, nous comparons les solutions trouvées par les trois modèles CG pour les trois réseaux et pour les 5 différents niveaux de trafic. Nous observons d'abord dans la figure 8.6, dans laquelle les barres d'erreur représentent l'écart entre les solutions fractionnaires relâchées et les solutions entières, que *CG-simple* et *CG-cuts* fournissent des solutions similaires. Cependant,



(a) pdh (b) atlanta (c) germany50

Figure 8.6: Performance des trois modèles de CG sur les topologies réseaux (a) *pdh*, (b) *atlanta* et, (c) *germany50*. 100 représente l'énergie utilisée dans le scénario traditionnel. *Notez les précisions de chaque méthode pour chaque scénario indiquées à l'intérieur des barres correspondantes.*



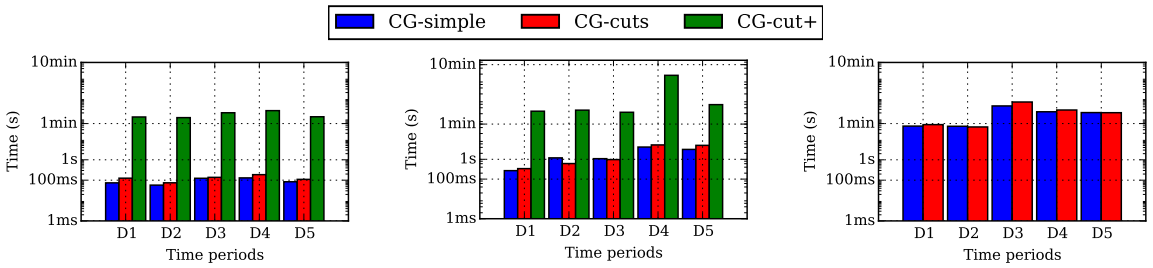
(a) pdh (b) atlanta (c) germany50

Figure 8.7: Précision, ε, des trois modèles de CG pour les topologies réseaux (a) *pdh*, (b) *atlanta* and, (c) *germany50*.

ε varie considérablement, comme le montre la Figure 8.7. Cuts améliore significativement ε : pour *CG-simple*, il varie entre 12% et 113% pour pdh, 10% et 97% pour atlanta, et 37% et 330% pour `germany50`. Pour *CG-cuts*, ε se situe entre 7 et 15 % pour pdh, 6 et 12 % pour atlanta, et 24 et 30 % pour `germany50`. Le ratio est encore amélioré avec *CG-cut+*: entre 4 et 8% pour pdh, 1 et 6% pour atlanta. Cependant, aucune solution n'a été trouvée dans un délai raisonnable pour la topologie `germany50`. Comme les économies d'énergie sont similaires pour les trois modèles, cela montre que les *trois modèles CG fournissent des solutions assez précises, comme le confirment les solutions et la précision des modèles* CG-cuts *et* CG-cut+.

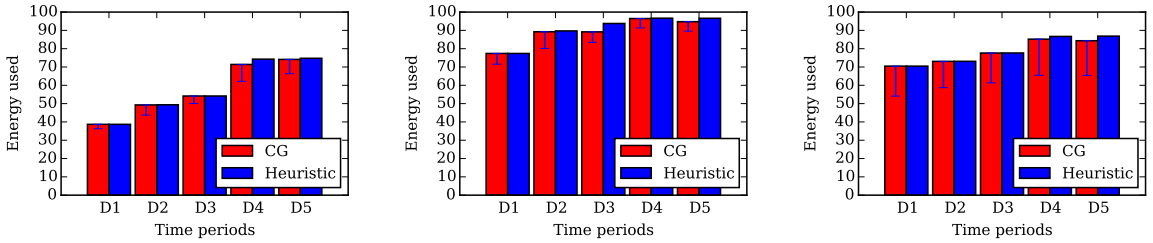
Enfin, dans la figure 8.8, nous comparons les temps d'exécutions des modèles. Nous observons que les temps d'exécution de *CG-cut+* (entre 17 s et 5 h) sont plus grands de plusieurs ordres de magnitude que ceux de *CG-simple* (entre 50 ms et 440 s) et *CG-cuts* (entre 70 ms et 670 s). Cela est dû principalement à ce que nous accélérons la résolution des deux modèles précédant en utilisant l'algorithme plus courts chemins Bellman-Ford dans le problème de pricing. Le deuxième facteur est que les coupes de *CG-cut+* ralentissent drastiquement le temps de convergence de la génération de colonnes.

Nous nous *concentrons maintenant sur le modèle* CG-cuts *model* pour résoudre des instances de grandes tailles, comme il offre le meilleur compromis en termes de précision (par rapport au modèle *CG-simple*) et de temps de calculs (par rapport au modèle *CG-cut+*).



(a) pdh (b) atlanta (c) germany50

Figure 8.8: Temps d'exécution des trois modèles de CG sur les topologies réseaux (a) *pdh*, (b) *atlanta* et, (c) *germany50*.



(a) pdh (b) atlanta (c) germany50

Figure 8.9: Consommation d'énergie pour GreenChains (désigné par Heuristic ici) pour le modèle CG sur les trois topologies.

Économies d'énergie

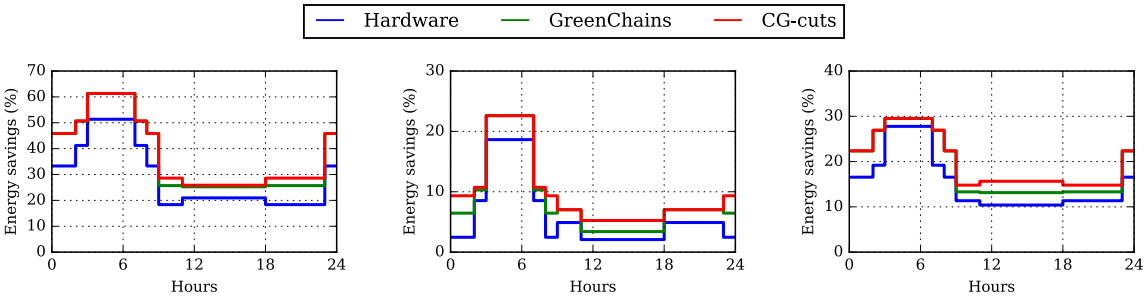
Nous comparons maintenant les économies d'énergie obtenues par GreenChains et *CG-cuts*. Nous considérons trois scénarios dans les expériences :

- *Scénario traditionnel.* Ce scénario correspond à celui d'un réseau existant, dont l'opérateur ne cherche pas à réduire la consommation d'énergie de son réseau. Son objectif est de minimiser la bande passante totale utilisée tout en respectant la capacité des liens et les contraintes des chaînes. Ce scénario est utilisé comme *point de comparaison* pour les algorithmes sensibles à l'énergie.

- *Scénario SDN (matériel).* Le scénario matériel correspond à un réseau SDN (non virtualisé) dans lequel un opérateur tente de réduire sa consommation d'énergie en adaptant le routage à la demande. Dans ce scénario, les fonctions réseau sont exécutées par du matériel spécifique placé à des positions données dans le réseau.

- *Scénario NFV.* Le scénario NFV est celui d'un réseau SDN virtualisé dans lequel des nœuds matériels génériques peuvent exécuter toutes les fonctions d'un réseau virtuel. C'est le scénario résolu par les solutions fournies dans la section 8.3.4.

Nous fournissons dans la figure 8.9 l'énergie utilisée pour les cinq niveaux de demande pour *pdh*, *atlanta*, et *germany50*. Les valeurs sont normalisées : 100 correspond au niveau énergétique du scénario traditionnel. Nous présentons également dans la figure 8.10 les économies d'énergie



(a) pdh (b) atlanta (c) germany50

Figure 8.10: Economie d'énergie pour GreenChains pour les trois topologies réseaux.

correspondantes pendant la journée. Nous voyons que nous avons obtenu des économies importantes en utilisant la virtualisation : entre 25 et 61 % pour *pdh*, 5 et 22 % pour *atlanta*, et 15 et 30 % pour *germany50*.

Valider GreenChains avec *CG-cuts* Nous comparons maintenant les solutions fournies par GreenChains et *CG-cuts* dans la figure 8.9. Les barres d'erreur sur les solutions de *CG-cuts* représentent les bornes inférieures données par $z^{\star}_{\text{LP}}$. Pour les périodes de bas trafic (D1, D2 et en D3), les deux méthodes fournissent des solutions similaires pour *pdh* et *germany50*. Le modèle CG fournit des solutions légèrement meilleures lorsque le trafic est plus élevé, avec une différence de 3 et 1% pour *pdh*, de 5 et 2 pour *atlanta*, et de 2 et 3% pour *germany50* respectivement dans la période D5. Remarquez que, même si le CG ne fournit qu'une légère amélioration des solutions de l'heuristique, il montre (c.f. ε valeur de précision) que l'heuristique donne de bons résultats, quelle que soit la période de trafic.

Impact sur le délai. Lorsque certains liens sont mis en mode veille, les chemins ont tendance à s'allonger. Cependant, nous avons montré dans [J6] que le délai maximum de chaque chemin reste en dessous de l'habituelle valeur de 50 ms de latence présente dans les accords de niveaux de service (Service Level Agreements ou SLAs) : le délai est inférieur à 5,4, 10,8 et 16.2 ms pour *pdh*, *atlanta* et *germany50* respectivement. De plus, la médiane du délai reste constante pour *pdh*, *atlanta* à 3,6 et 5.4 ms, respectivement. Pour *germany50*, il passe de 7.2 pour D5 (pas de lien en mode veille) à 9 ms pour D1.

8.3.6 Conclusions

Dans ce travail, nous avons étudié le potentiel de la virtualisation des réseaux pour réduire la consommation d'énergie des réseaux. Nous introduisons un modèle de génération de colonnes pour résoudre le problème de la minimisation de la consommation d'énergie du réseau tout en satisfaisant les exigences de chaînes de services. Nous proposons également GreenChains, une heuristique basée sur un ILP que nous validons à l'aide de notre modèle de génération de colonnes. Nous comparons ensuite trois scénarios différents correspondant à un déploiement continu du paradigme SDN et NFV pour l'efficacité énergétique. Nous montrons qu'un opérateur, en utilisant le contrôle SDN, peut économiser de l'énergie en choisissant dynamiquement les trajets des flux en fonction des variations de la demande au cours de la journée. En effet, cela permet d'éteindre une grande partie des équipements du réseau. En effet, par rapport à un

scénario classique, SDN permet de réaliser entre 18 et 51 % d'économies d'énergie pendant la nuit. Nous avons également démontré que le déploiement du paradigme NFV dans un réseau SDN permet de réaliser des économies d'énergie supplémentaires comprises entre 4 et 12 %. En effet, le choix dynamique des emplacements des fonctions réseau en fonction des variations des demandes permet une plus grande flexibilité dans le choix des chemins réseau et conduit à l'utilisation de moins d'équipements réseau.

8.4 Aller vers la mise en pratique et expérimentations avec des plateformes SDN

8.4.1 Introduction

Les réseaux logiciels offrent la possibilité d'implémenter des algorithmes qui adaptent les configurations réseau à la charge/demande, et vont donc permettre de diminuer la consommation énergétique des résaux. Mais, ils induisent de *nouvelles problématiques* qu'il faut étudier et de nouveaux coûts (en délai ou bande passante) qu'il est important d'estimer. En particulier,

- L'utilisation d'un (ou de plusieurs) contrôleur SDN induit plusieurs problèmes potentiels.
  - Cela peut causer des délais supplémentaires pour les flots. En effet, un routeur peut avoir à demander des informations au contrôleur pour traiter certains flots.
  - Le contrôleur peut former un goulot d'étranglement en raison de ses performances CPU ou de sa bande passante limitée.
  - Il représente de plus un point de vulnérabilité unique en cas de pannes.

  Cela pourrait dans certains cas induire des pertes de paquets. Il est important de pouvoir mesurer les effets sur le réseau.
- Comment gérer la co-existence avec les protocoles classiques qui vont continuer de tourner dans les réseaux, en particulier.
  - protocoles de routage du matériel non SDN, par exemple OSPF.
  - protocoles de détection de pannes, par exemple BFD (*Bidirectional Forwarding Detection*).

  Comment ne pas perturber ces protocoles ? Est-il possible d'avoir de meilleurs résultats que ces protocoles vénérables en pratique ?
- Comment utiliser de façon efficace la métrologie procurée par SDN (à savoir des mesures de trafic sur les liens et les noeuds) ?

Expérimentations avec une plateforme matérielle (petit rappel). Dans le chapitre 6 (section 6.7), nous avons répondu à certaines de ces interrogations en nous concentrant sur un scénario particulier, le routage efficace en énergie dans un routage SDN avec un nombre limité de règles de routage.

- l'utilisation d'un contrôleur induit-elle un délai supplémentaire sensible pour les paquets ?
- le contrôleur est-il capable de supporter la charge réseau ?
- le contrôleur peut-il gérer les algorithmes d'optimisation en même temps que ses attributions de routage classique ?

En utilisant la plateforme décrite dans 6.7.1. Nous avons étudié un certain nombre de métriques : le *délai* introduit par les communications avec le contrôleur, l'augmentation potentielle du *taux de perte* due à la gestion dynamique du routage et de la compression et la *charge du contrôleur* avec ou sans compression. Nous montrons ainsi que MINNIE est capable de *minimiser le nombre d'entrées dans les switches, tout en gérant avec succès la dynamique des clients et en maintenant*

la stabilité du réseau. En utilisant SDN, il est ainsi possible de mettre en œuvre des politiques dynamiques de routage sans augmenter significativement les délais, ni générer de pertes de paquets.

Contributions. Nous donnons ici plusieurs autres pistes pour répondre à ces questions.

- Dans la section 8.4.2, nous proposons plusieurs mécanismes pour mettre en pratique le routage économique en énergie dans des réseaux SDN hybrides, à savoir des réseaux dans lesquels coexistent du matériel traditionnel et du matériel SDN. En particulier, nous étudions la coexistence du routage OSPF avec du routage dynamique SDN, nous proposons plusieurs mécanismes de gestions pannes et de réaction à des variations brusques de traffic, ainsi que des mécanismes pour les détecter en utilisant de la métrologie SDN.

- Nous testons en pratique ces solutions en utilisant une plateforme logicielle SDN en section 8.4.3.

8.4.2 Mécanismes pour mettre en pratique le routage économe en énergie

Je présente ici un travail [Ci35, J7] fait en collaboration avec deux doctorants N. Huin et M. Rifai, ainsi que D. Lopez, G. Urvoy- Keller et J. Moulierac des équipes COATI et SigNet du laboratoire I3S.

Introduction

Différents scénarios peuvent être envisagés pour la transition des réseaux traditionnels (*legacy networks*) vers des réseaux SDN [191]. L'un des plus réalistes est une migration progressive, où les équipements existants sont remplacés sur une longue période par des équipements SDN. Il y a donc une période de coexistence des équipements et protocoles traditionnels avec ceux de SDN. Par exemple, pour router des paquets à l'intérieur du réseau, les nœuds traditionnels doivent suivre des protocoles traditionnels, tels que OSPF, tandis que les noeuds SDN peuvent choisir les sauts suivants (next hops) des paquets en utilisant un algorithme d'optimisation exécuté dans le contrôleur.

Nous considérons ici le problème du routage efficace en énergie dans un réseau SDN hybride. Afin d'optimiser l'énergie dans les réseaux hybrides, nous présentons SENAtoR pour *Smooth Energy Aware Routing*. L'idée principale est que le contrôleur choisit d'abord l'ensemble d'itinéraires qui minimise le nombre d'équipements réseau utilisés pour le trafic actuel, puis nous mettons les interfaces ou les nœuds SDN en veille. Nous considérons le trafic dynamique typique d'un opérateur et, par conséquent, notre solution adapte le nombre d'équipements réseau actifs et inactifs pendant la journée.

Lorsque les nœuds SDN sont mis en veille et leurs liens désactivés[1], le trafic doit être redirigé, tout en évitant des pertes de paquets. Il est donc impossible d'attendre la convergence des protocoles traditionnels (par exemple OSPF). De plus, si le trafic réseau ISP présente habituellement des variations de débit lentes, il subit également des changements soudains qui peuvent correspondre à des défaillances (de liens ou de nœuds) ou à des pics de trafic brusques (*flash crowds*) [2] [376]. Dans ces cas, la solution économe en énergie doit être capable de réagir très rapidement et d'allumer des dispositifs précédemment éteints. Nous proposons donc trois mécanismes (détaillés ci-dessous) : tout d'abord, nous utilisons des tunnels pré-configurés en tant que routes

[1] Dans ce travail, nous utilisons les termes désactiver et mettre en mode veille de façon interchangeable.

de secours en cas de défaillance ou de mise en veille de liens réseau. Deuxièmement, nous utilisons le contrôleur SDN pour supprimer tout paquet OSPF dirigé vers une interface que nous désirons mettre en veille, dans le but de simuler une panne de lien. Cela oblige les nœuds OSPF à converger vers un autre routage. Enfin, nous utilisons les fonctionnalités de surveillance (*monitoring*) de SDN pour détecter rapidement des pics de trafic inattendus ou des pannes de liens.

Nos contributions sont les suivantes :

- Nous rapprochons les solutions énergétiques de leurs mises en pratique dans des réseaux ISP.

- Nous modélisons le problème de l'EAR dans un réseau SDN hybride. Nous formulons un ILP qui décide des équipements réseau à mettre en mode veille, et en même temps, de quels tunnels configurer pour rediriger le trafic. La formulation présente plusieurs difficultés. Tout d'abord, les nœuds traditionnels (qui suivent les protocoles traditionnels) doivent acheminer les flots via des plus courts chemins, alors que les nœuds SDN peuvent router un flot librement vers tous les voisins. Deuxièmement, les tunnels doivent être définis de sorte à ce qu'il existe un chemin pour chaque flot, même lorsque plusieurs équipements réseau sont mis en mode veille. De plus, l'ensemble des tunnels qui sont utilisés, dépend du niveau du trafic. L'ILP offre des solutions optimales pour les petits réseaux. Pour les grands réseaux, nous proposons des algorithmes heuristiques.

- Nous proposons plusieurs mécanismes pour éviter les pertes de paquets lors de la mise en veille des équipements réseau : tunnels, arrêt doux des liens et détection des variations de trafic.

- Pour valider nos solutions, nous avons effectué des simulations sur plusieurs topologies de réseau et évalué les économies d'énergie pour différents niveaux de pénétration de la technologie SDN.

- Les mécanismes ont été mis en œuvre et testés sur une petite plateforme SDN. Cela nous a permis d'une part d'utiliser le modèle énergétique d'un véritable routeur SDN haut de gamme, un switch de technologie SDN HP5412zl que nous affublons du doux nom de switch SDN par la suite, et d'autre part de montrer qu'il est possible d'implémenter des solutions économes en énergie tout en réduisant les pertes de paquets par rapport aux protocoles existants.

État de l'art

Réseaux hybrides SDN. Comme le scenario le plus réaliste d'adoption du paradigme SDN est une migration progressive, nous nous concentrons sur les réseaux hybrides. Dans ces réseaux, des équipements SDN coexistent avec des équipements traditionnels. La difficulté principale est de faire coexister les protocoles très différents de ces deux technologies. Les auteurs de [191] discutent de défis et d'opportunités de recherche sur les réseaux hybrides, tandis que [229] se concentre sur comment router efficacement dans les résaux SDN hybrides. Les auteurs montrent comment tirer avantage de la technologie SDN pour améliorer l'utilisation des liens et réduire les pertes de paquets et les délais. Nous étendons ce travail en considérant l'efficacité énergétique.

Gérer les pannes et pics de trafic (Flash Crowds). L'extinction des périphériques SDN dans les réseaux hybrides IP-SDN pourrait être interprétée comme des pannes de liens ou de

noeuds par équipements traditionnels et pourrait diminuer la capacité du réseau à supporter des augmentations de trafic soudaines, mais pas malveillantes (En raison, par exemple, d'événements exceptionnels tels que les tremblements de terre). Par conséquent, notre solution met en œuvre certaines fonctionnalités pour faire face correctement aux pannes de liens et aux flash crowds. La communauté du réseau a traité de tels problèmes avec l'aide de SDN :

- **Détection et gestion des pannes.** Comme dans les équipements traditionnels, les équipements SDN peuvent compter sur l'algorithme BFD (Bidirectional Forwarding Detection ou Détection de transfert bidirectionnel) pour détecter les défaillances d'un lien [172]. Une fois que l'échec de la liaison a été détectée, OpenFlow offre déjà une technique de gestion de panne grâce à la notion de règles de groupe FAST-FAILOVER, où plusieurs règles par flux peuvent être installées. La protection du lien et du canal de contrôle d'OpenFlow requiert cependant des solutions plus complexes, comme celle proposée dans [134]. Pour éviter les pertes en cas de panne de liens dans les réseaux hybrides, [186] propose d'introduire des tunnels préétablis à partir d'un routeur traditionnel vers un routeur SDN, pour construire des chemins de sauvegarde. Les nœuds SDN peuvent ensuite re-router le trafic sur des chemins disponibles en utilisant ces tunnels. Nous empruntons cette idée et proposons d'utiliser des tunnels préétablis, lorsqu'un nœud est désactivé. Il s'agit d'une adaptation et d'une généralisation de la solution proposée dans [186] pour gérer un échec de liaison. En effet, nous l'utilisons pour l'efficacité énergétique lorsque plusieurs liens sont désactivés. Nous autorisons également la définition de tunnels entre n'importe quelle paire de nœuds (OSPF ou SDN) et nous effectuons des expérimentations pratiques pour valider la méthode.

- **Détection des variations de trafic dans les réseaux SDN.** Les variations de trafic des réseaux cœur sont habituellement faibles, car le trafic y est une agrégation de millions de flots [376, 378]. Cependant, des variations fortes se produisent en cas de pannes de liens ou de pics de trafic (flash crowds) [365]. Des méthodes ont été proposées pour détecter ces variations dans les réseaux existants, voir par exemple [362, 374]. Dans les réseaux SDN de centres de données, Netfuse [208] a été proposé pour atténuer l'effet des variations de trafic. Dans cette étude, nous proposons une méthode pour détecter les variations de trafic brusques dans un réseau hybride SDN.

Routage efficace en énergie dans un réseau hybride

Modèle. Router dans un réseau hybride. Un réseau est modélisé par un graphe dirigé $D = (V, A)$ dans lequel un nœud représente un point de présence (PoP or Point of Presence) et un arc représente un lien entre deux PoPs. Un PoP est constitué de plusieurs routeurs reliés entre eux par un graphe complet [Ci68]. Chaque lien $(u, v) \in A$ est connecté à deux routeurs dédiés dans les PoP u et v. Un lien (u, v) a une capacité maximum C_{uv}. Nous considérons des réseaux hybrides dans lesquels les équipements SDN sont installés au côté d'équipements traditionnels. Nous considérons un scénario dans lequel les PoP ne contiennent pas d'équipements hétérogènes, c'est-à-dire, qu'en un PoP donné, tous les routeurs sont SDN ou non. Les routeurs traditionnels suivent un protocole de routage traditionnel, comme OSPF. Nous notons $n^t(u)$ le prochain saut (next hop) vers la destination t à partir du routeur traditionnel u. Les switches SDN sont contrôlés par un ou plusieurs contrôleurs et peuvent être configurés dynamiquement pour router vers n'importe lequel de leurs voisins.

Estimation du trafic. Nous supposons qu'un opérateur d'un ISP est capable d'estimer la matrice de trafic de son réseau en utilisant un échantillonage de son trafic fourni par des mesures

netflow [366] ou, dans le cas d'un réseau SDN hybride, en combinant des données SDN et OSPF-TE [229]. Ainsi, notre solution va surveiller le trafic en continu et calculer les équipements, liens ou nœuds, à mettre en veille.

Modéle énergétique et mécanisme d'efficacité énergétique. Pour modéliser la consommation d'énergie d'un lien, nous utilisons un modèle hybride composé d'un coût de référence, représentant la puissance utilisée lorsque le lien est actif et un coût linéaire en fonction de son débit. Cela permet, selon la valeur des paramètres, d'exprimer les différents modèles de puissance (entre ON-OFF et énergie proportionnelle) trouvés dans la littérature, voir [143] pour une discussion. L'utilisation de l'énergie d'un lien est exprimée comme suit

$$P_l(u,v) = x_{uv}U_{uv} + \mathcal{F}_{uv}L_{uv}$$

où x_{uv} représente l'état du lien (ON ou OFF), U_{uv} est la consommation d'énergie de base d'un lien actif, $\mathcal{F}_{uv}$ le montant total de bande passante sur le lien, et L_{uv} le coefficient de puissance du lien. Les routeurs ont deux états de puissance : actif ou dormant, et leur consommation totale $P_n(u)$ est donnée par

$$P_n(u) = B_u + A_u \cdot \mathbb{1}_{\text{actif}} + \sum_{v \in N^+(u)} P_l(u,v),$$

où B_u est l'utilisation énergétique de l'état de veille et A_u la puissance supplémentaire utilisée lorsque l'équipement est actif.

Pour économiser de l'énergie, les liens doivent être mis hors tension et les routeurs en veille. Seuls les switches SDN peuvent être mis en mode veille sans impact négatif sur le réseau. Comme cela devrait être fait de manière dynamique selon le trafic réseau, la décision est prise par le contrôleur SDN. Ainsi, seuls les liens avec un switch SDN comme extrémité peuvent être éteints. Étant donné que les PoP sont interconnectés à l'aide de routers dédiés à l'intérieur de leur infrastructure, si un lien entre deux PoPs est éteint, alors chaque router du lien peut être arrêté s'il est SDN.

Notre proposition: SENAtoR SENAtoR met en veille des nœuds et des liens en fonction de la charge de trafic sur les liens entre PoPs. Il implémente trois mécanismes principaux pour éviter des pertes de données.

1. Tunnels. Ce premier mécanisme s'inspire de la solution proposée dans [186] pour traiter une panne unique d'un lien. L'objectif était d'éviter d'attendre la convergence des protocoles de routage traditionnels en utilisant des tunnels à partir d'un nœud avec un lien défaillant. Similairement, l'arrêt d'un lien avec le contrôleur SDN entraîne une détection d'une panne de lien par OSPF et une période de convergence. Pour éviter de perdre des paquets pendant la phase de convergence, nous utilisons des tunnels pré-établis qui constituent des chemins de backup pour rediriger le trafic qui autrement serait perdu. L'idée est de renvoyer le trafic qui utiliserait ce lien ou ce nœud vers un nœud intermédiaire dont le plus court chemin vers la destination n'utilise pas de liens éteints.

Avec la plupart des mécanismes des réseaux actuels, les tunnels ne peuvent pas être déployés dynamiquement pendant le fonctionnement du réseau. Ils doivent donc être préétablis statiquement. Nous considérons donc deux variantes du problème : (i) avec la sélection du tunnel (ii) avec un ensemble de tunnels préconfigurés.

2. Désactivation des liens en douceur. Pour empêcher les routeurs OSPF d'envoyer des paquets vers un nœud qui vient d'être mis en mode veille par le mécanisme d'économie d'énergie, nous proposons de forcer la reconfiguration OSPF avant que la carte d'interface réseau (NIC) ne soit vraiment désactivée. L'idée est que le contrôleur SDN va simuler une panne du nœud qui

doit être désactivé. Nous faisons en sorte que le contrôleur soit en charge d'envoyer les paquets de contrôle OSPF de BFD à la place des nœuds SDN. Ensuite, quand le contrôleur décide de mettre en veille un nœud SDN, il arrête d'envoyer les paquets de contrôle OSPF en réponse aux paquets hello envoyés par ses nœuds OSPF voisins. Cela va donc simuler une panne de ce nœud pour ses voisins. Cela permet aux routeurs OSPF voisins de converger vers une vue réseau excluant ce nœud. En effet, après la valeur par défaut, *dead interval* de $3 \times$ *hello interval* sans recevoir aucun paquet hello, un routeur OSPF déclare son voisin comme mort et cesse d'utiliser le lien. Cependant, jusqu'à la fin du dead interval, le lien est actif et le trafic circule sur ce lien. Après le dead interval plus une marge de sécurité de 10 secondes supplémentaires, et si aucun trafic n'est reçu via ses liens (que nous définissons comme la période de convergence attendue d'OSPF), le nœud SDN est mis en mode veille. Cette stratégie simple empêche toute perte supplémentaire de paquets.

3. Détection des pannes ou Flash Crowds avec SDN. Le sur-dimensionnement des capacités des réseaux est exploité par des algorithmes éco-énergétiques. Cependant, si les réseaux sont surdimensionnés, c'est en particulier pour pouvoir gérer des variations de trafic importantes qui peuvent être dues à des pannes ou à des flash crowds. Il est donc crucial pour les mécanismes d'économie d'énergie, qui éteignent des équipements, de ne pas avoir d'impact sur la tolérance aux pannes des réseaux. Dans ce but, nous proposons d'utiliser les données métrologiques reçues par le contrôleur à partir des nœuds SDN pour détecter les variations de trafic importantes. Dans ce cas, nous réactivons tous les nœuds du réseau pour éviter les pertes de paquets.

3a. Gestion des pics de trafic. Des pics de trafic sont relativement rares en raison du fort multiplexage statistique des réseaux cœurs des FAI. Cependant, des événements exceptionnels (tels que des tremblements de terre) peuvent entraîner des flash crowds. Par conséquent, nous complétons SENAtoR avec un mécanisme de sauvegarde qui vise à réactiver les nœuds SDN inactifs en cas de pics de trafic soudains. Ce dernier événement est défini sur une base par lien comme suit: le contrôleur collecte la charge de trafic sur chaque interface de chaque commutateur actif SDN à petite échelle de temps (dans nos expériences, une fois par minute). Nous comparons ensuite le niveau de trafic réel reçu à l'interface i, $E_i(t)$, au taux estimé, $E_i^{ES}(t)$, à la dernière époque où SENAtoR a pris sa décision de désactiver certains liens. Dans le cas de $E_i(t) \geq 1.5 \times E_i^{ES}(t)$, pour toute interface i, tous les routeurs SDN inactifs sont réactivés. La valeur de 50 % a été choisie de manière conservatrice, car, en général, les réseaux ISP sont surdimensionnés. Après la période de convergence d'OSPF, le contrôleur réexécute SENAtoR pour obtenir une nouvelle architecture verte si possible.

3b. Gestion d'une panne de lien. Nous employons un mécanisme similaire au mécanisme d'atténuation des pics de trafic en cas de pannes de liens. Lorsqu'un lien connecté à un routeur actif SDN ou entre des nœuds OSPF est en panne, SENAtoR réactive tous les nœuds SDN inactifs. Le trafic est aussi re-routé via un chemin différent si possible (y compris via les tunnels préétablis). Une panne de liens entre des noeuds OSPF peut être détectée par des nœuds SDN proches en raison de la variation du trafic sur leurs liens. Un lien en aval d'un lien défaillant, observe une diminution du trafic d'une interface par rapport à ce que la matrice de trafic prédit. Nous bénéficions du fait que, dans les réseaux ISP typiques, le trafic est all-to-all, c'est-à-dire de tout PoP vers tout autre PoP. Par conséquent, tout routeur SDN du réseau est susceptible de détecter la perte d'un lien, car une fraction du trafic qu'il gère est affecté par la panne. Encore une fois, nous utilisons un seuil prudent de 50 %, c'est-à-dire qu'un commutateur SDN doit détecter une diminution de 50 % de l'une des charges de ses liens pour déclencher le mécanisme de gestion de panne de lien. Encore une fois, après la période de convergence OSPF prévue, le

contrôleur réutilise SENAtoR pour obtenir une nouvelle architecture verte.

En résumé. Lorsqu'un switch SDN doit être mis en mode veille et que des liens doivent être éteints, le mécanisme est le suivant : le contrôleur SDN redirige le trafic d'abord afin qu'aucun flux ne traverse ce nœud ou ce lien. Ensuite, le contrôleur SDN envoie l'ordre au switch SDN d'entrer en mode veille ou de désactiver l'interface correspondant au lien. Étant donné qu'aucun autre paquet de donnée n'utilise le lien, l'interface du nœud SDN est désactivée et l'interface du routeur traditionnel peut entrer automatiquement en mode veille, en utilisant par exemple IEEE 802.3az Energy-Efficient Ethernet [291].

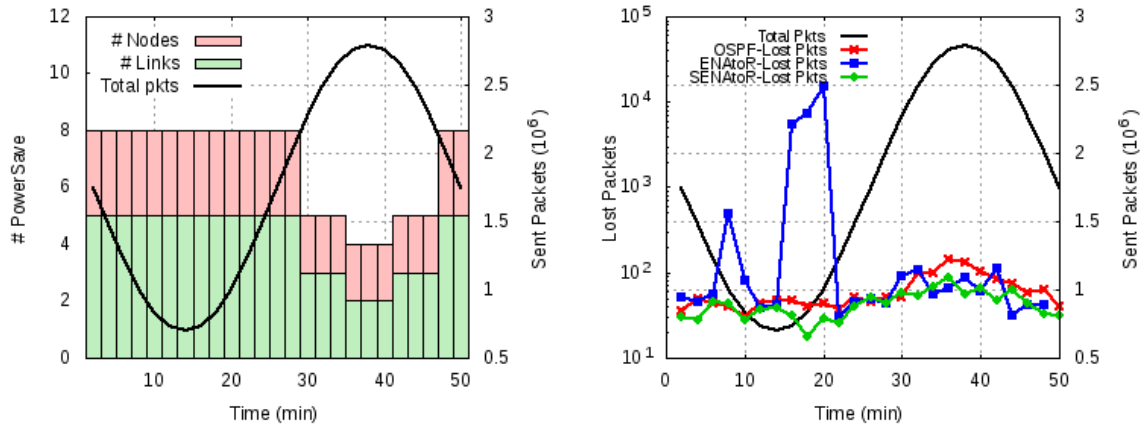
Comment router et sélectionner les équipements à mettre en veille avec SENAtoR. Dans [J11], nous avons proposé un ILP pour décider quels équipements mettre en veille et, dans le même temps, quels tunnels utiliser pour re-router le trafic. Cette formulation présente plusieurs difficultés. D'abord, les noeuds non-SDN doivent router en suivant les plus courts chemins utilisés par les protocoles classiques, alors que, dans le même temps, les noeuds SDN peuvent router de façon libre vers n'importe lequel de leurs voisins. Ensuite, les tunnels doivent être configurés de façon à ce qu'il existe toujours un chemin possible pour chaque flot, même quand un grand nombre d'équipements réseau ont été mis en veille. Nous avons aussi proposé un algorithme heuristique pour résoudre le problème sur des instances de grandes tailles. Nous avons évalué ces solutions sur différentes topologies de SNDLib. Nous montrons que *SENAtoR obtient des gains d'énergie entre 5% et 35% pour différents niveaux de pénétration du matériel SDN.*

8.4.3 Expérimentations avec une plateforme logicielle

Dans cette section, nous présentons les résultats obtenus avec une plateforme Mininet. Notre objectif est de démontrer que SENAtoR peut effectivement désactiver les liens et mettre des switches SDN en mode économie d'énergie sans perdre de paquets, grâce à une intégration harmonieuse avec OSPF et à nos mécanisme pour anticiper la mise en veille de liens.

Plateforme

Nous avons assemblé une plateforme de test pour émuler un réseau SDN hybride à l'aide de Mininet [424] et d'un contrôleur Floodlight distant [421]. La topologie du réseau Mininet est basée sur la topologie *atlanta* avec un déploiement SDN de 50%. Les routeurs OSPF sont matérialisés en tant que nœuds hôtes dans Mininet et exécutent le logiciel Quagga [418], alors que des Open vSwitches (OvS) [422]) agissant comme switches SDN. Notre contrôleur Floodlight est capable d'analyser (parse) et de répondre aux paquets OSPF hello reçus et transmis par les commutateurs SDN OvS (via des règles Openflow adéquates installées dans les switches SDN). Cela assure ainsi le bon fonctionnement des routeurs OSPF adjacents. Les tunnels sont implémentés sous forme de tunnels GRE simples et l'interaction entre l'interface du tunnel et les interfaces régulières est contrôlée en réglant la distance administrative afin que les interfaces ordinaires aient une priorité plus élevée. Lorsque SENAtoR notifie un commutateur SDN PoP de passer en mode veille, nous éteignons toutes ses interfaces et nous le déconnectons du reste du réseau. Pendant ce mode veille, la mémoire maintient le jeu de règles de routage précédemment installé par le contrôleur afin d'être capable de rétablir rapidement le mode actif normal.



(a) Numbre de liens éteints

(b) Pertes de paquets

Figure 8.11: Topologie *atlanta*

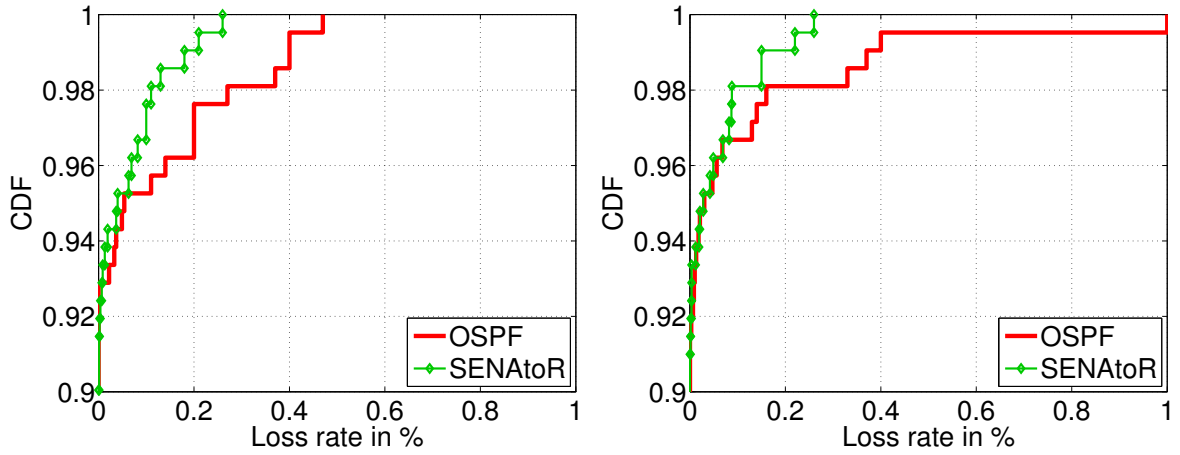
Extinction des liens sans pertes

Dans la figure 8.11a, nous modifions le trafic au fil du temps afin de simuler des variations douces du débit moyen de trafic. Ceci est réalisé en prenant une matrice de trafic et en la dimensionnant en utilisant la même fonction sinusoïdale que dans la Figure 8.1(b).

Les résultats d'économie d'énergie dans la figure 8.11a sont consistants avec ceux des simulations [R82], c'est-à-dire que le même nombre de liens et de nœuds est désactivés dans tous les cas, ce qui est rassurant car nous utilisons le même code au niveau du contrôleur. La valeur ajoutée de l'expérience consiste à évaluer si l'interaction entre SDN et OSPF est efficace, c'est-à-dire que notre approche douce d'arrêt de liens évite efficacement les pertes de données. La figure 8.11b dépeint les séries temporelles de perte de paquets avec OSPF pur (OSPF exploite le réseau complet et aucun lien n'est désactivé dans ce cas), SENAtoR et ENAtoR (SENAtoR sans le "s" pour *smooth*, c'est-à-dire sans le mécanisme doux d'arrêt de liens). La figure montre l'importance d'anticiper l'arrêt du lien (résultant du fait de mettre des switchs SDN en mode veille) comme cela se fait dans SENAtoR puisque les pertes explosent à 10^4 paquets (soit 1% des paquets envoyés) lorsque cette fonctionnalité est désactivée (ENAtoR). Dans ce cas, le taux de perte élevé d'ENAtoR est proportionnel à la quantité de temps qu'il faut pour que OSPF déclare le lien éteint, multiplié par de débit du trafic. En revanche, SENAtoR parvient à maintenir la même taux de perte de paquets qu'un réseau purement OSPF sans extinction de liens, avec des taux de perte négligeables (10^{-4}%), tout en utilisant un sous-ensemble des liens et des nœuds dans le réseau.

Gérer les pics de trafic

Pour illustrer le mécanisme d'atténuation des pics de trafic, nous considérons une matrice de trafic fixe (pas de mise à l'échelle) et nous provoquons un pic de trafic soit sur un noeud OSPF directement connecté à un switch (Figure 8.12a) ou entre des nœuds OSPF (Figure 8.12b). Nous montrons la distribution cumulative (cdf) des taux de perte de toutes les connexions. De toute évidence, l'algorithme de détection de pic de trafic de SENAtoR lui permet de surpasser OSPF. L'une des raisons d'un tel phénomène est que les noeuds OSPF réguliers n'ont pas de mécanismes pour faire automatiquement de l'équilibrage de charge du trafic en cas de pic de trafic.



(a) Pour un lien SDN-OSPF (b) Pour un lien OSPF-OSPF

Figure 8.12: Expérimentation d'un pic de trafic sur la topologie *atlanta*.

Panne de liens

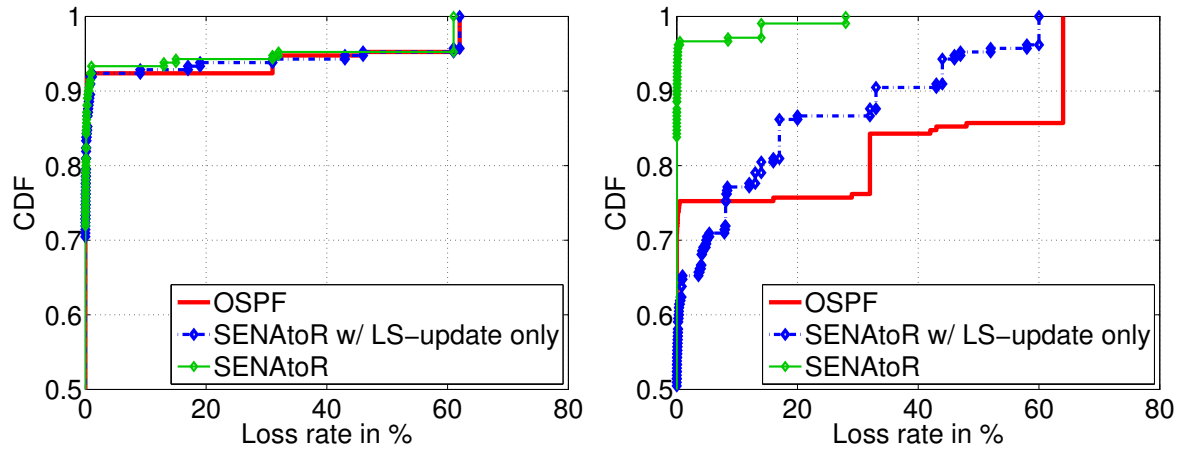
Nous considérons à nouveau une matrice de trafic fixe (pas de mise à l'échelle) et nous provoquons une panne de liaison entre un switch et un routeur OSPF ou entre deux routeurs OSPF et signalons les taux de perte correspondants sur les figures 8.13a et 8.13b. Nous comparons trois cas : (*i*) le scénario OSPF traditionnel, dans lequel une panne de lien est traités avec un temps de convergence long, (*ii*) SENAtoR qui utilise seulement les mises à jour des OSPF Link State (LS) pour détecter les changements de réseau et (*iii*) SENAtoR avec son mécanisme de détection et d'atténuation des *pannes de liens*.

Nous observons d'abord que, même dans le cas (*ii*), *SENAtoR n'a pas de taux de perte plus élevés que le cas (i)* (et des taux de perte nettement inférieurs lors d'une panne d'un lien entre deux nœuds OSPF). Cela se produit *même si certains des switchs et des liens étaient à l'arrêt au moment de l'échec* et devaient donc être rallumés. En effet, les switches SDN n'ont pas besoin d'attendre la convergence OSPF avant de réacheminer le trafic à travers les tunnels préétablis. Le mécanisme d'atténuation de panne du lien améliore encore la situation.

Nous observons ensuite un résultat contre-intuitif: les taux de perte utilisant SENAtoR sont plus petits lorsque l'échec survient sur un lien OSPF-OSPF plutôt que sur un lien SDN-OSPF. Deux facteurs contribuent à ce résultat. Tout d'abord, les nœuds SDN sont placés dans des emplacements clés du réseau afin qu'ils transmettent davantage de trafic. Par conséquent, une panne de ces nœuds induit des taux de perte plus élevés. Deuxièmement, dès qu'un nœud SDN en aval détecte une défaillance d'un lien OSPF-OSPF, SENAtoR limite le trafic circulant sur ce lien en demandant aux nœuds SDN en amont de re-router leur trafic.

8.4.4 Conclusion

Dans ce travail, nous avons présenté une solution de routage, SENAtoR, qui garantit la tolérance aux pannes et la gestion de la surcharge de trafic du réseau. SENAtoR a été enrichi d'une méthode pour éteindre de façon douce les équipements résau (sans perted de paquets) et par des services de détection de pannes et de pics de trafic. Les simulations et expérimentation avec des équipements émulés exécutant des agents OSPF complets montrent que SENAtoR peut traiter correctement des événements de réseau inattendus. De manière plus frappante, nos expériences



(a) Sur un lien SDN-OSPF (b) Sur un lien OSPF-OSPF

Figure 8.13: Link failure Expérimentation d'une panne de lien sur la topologie *atlanta*.

montrent que, même lorsque les services verts sont activés et que des pics de trafic se produisent dans un nœud non compatible SDN, SENAtoR a des taux de perte inférieurs au cas de l'OSPF. En effet, le contrôleur SDN peut fournir les itinéraires les plus appropriés. En conclusion, SENAtoR permet des économies d'énergie tout en étant compatible avec les infrastructures de réseau actuelles. En tant que travail futur, SENAtoR pourrait être enrichi avec une étude plus approfondie des variations du réseau de trafic afin de fournir les seuils les plus adaptés pour la détection de pannes ou de pics de trafic.

Chapter 9

Conclusions

9.1 Perspectives

Les prochaines années vont être passionnantes pour qui aime l'algorithmique et l'optimisation des infrastructure réseaux et de stockage. En effet, nous sommes en train d'assister à **plusieurs révolutions majeures**. La première est l'introduction au niveau industriel des **réseaux logiciels**, en particulier pour les réseaux d'entreprises. Ce nouveau paradigme est poussé par de grandes entreprises ayant une forte culture logiciel, en particulier Google qui l'a utilisé pour gérer son réseau inter-centre de données B4 [218]. En parallèle, on assiste à la **convergence des infrastructures de réseaux et des centres de données**. Cela est dû principalement à deux phénomènes : les grands opérateurs de centres de données comme Google, Facebook, Amazon, Microsoft, commencent à installer leurs propres infrastructures réseaux. Dans le même temps, les opérateurs réseaux, voyant que la chaîne de valeurs s'est transformée et que le profit est passée du transport vers les applications, se sont aussi mis à gérer leurs propres centres de données. Par exemple, en France, Orange a racheté Dailymotion. Cette convergence et ce rapprochement de culture a poussé l'**introduction de techniques de virtualisation dans les réseaux,** similaires à celles existants dans les centres de données. Ces techniques portent la promesse de faciliter la gestion des réseaux, de diminuer les coûts matériels et de maintenance, et de pouvoir proposer des services complexes aux utilisateurs, en particulier en gérant des chaînes de fonctions de services virtualisées ou des *slices*.

En parallèle de cette modification profonde des paradigmes réseaux, on assiste au **développement de nouveaux types de communications réseaux, la future 5G, l'internet des objets (IoT) et les communications de machine à machines (M2M)**. Les nouvelles architectures qui se mettent en place vont intégrer ces nouveaux paradigmes. Nous pouvons citer comme exemple le 5G pour laquelle la virtualisation réseau est envisagée, en particulier pour gérer certaines fonctions des nouvelles antennes ou pour proposer des services à fortes valeurs ajoutées à l'utilisateur mobile en utilisant le slicing réseau.

Tous ses bouleversements apportent de nouveau ou renouvellent d'anciens problèmes algorithmiques et d'optimisation des réseaux auxquels il va être crucial de trouver de bonnes réponses dans les prochaines années.

Réseaux logiciels → Routage et placement de ressources dynamiques. La séparation du plan de contrôle dans les réseaux logiciels avec l'introduction d'un contrôleur qui dispose d'informations métrologiques sur le réseau permet de prendre des décisions rapides en fonction de la demande et de l'utilisation des ressources. Cela n'était pas possible dans les réseaux tra-

ditionnels. Il faut ainsi développer de nouveaux algorithmes pour faire du routage et placement de ressources, en particulier virtuelles, dynamiques.

Convergence centre de données et réseaux → Optimisation jointe du placement des applications et du trafic réseau. Comme discuté ci-dessus, les fonctions réseaux peuvent maintenant être effectuées par des machines virtuelles comme les applications traditionnelles. De plus, les routes réseaux peuvent être changées dynamiquement par le contrôleur SDN. Avec la convergence des réseaux et centres de données, et en particulier de leurs contrôle, il existe maintenant un triple problème joint de placement des applications, des fonctions réseaux et de choix des routes. L'importance de ce problème a bien été exhibé dans [270] qui montre que dans certains centres de données 50% du temps de complétion des applications vient des communications réseaux. Cela va amener à un changement des procédures de routage et de scheduling.

Virtualisation → Plongement de réseaux virtuels en particulier network slicing + Tolérance aux pannes de ressources virtuelles. La virtualisation des réseaux a renouvelé les thématiques de plongement d'un réseau logique dans un réseau physique qui avait été très étudiées par exemple pour le choix des routes IP sur un réseau physique optique WDM. Le problème VNE pour Virtual Network Embedding [320] correspond au plongement d'un réseau d'un opérateur virtuel louant une partie du réseau physique d'un autre opérateur. Le placement de chaînes de fonctions de services correspond à un cas simplifié de ce problème dans lequel les sommets sources et destination sont fixés et le réseau logique se réduit à un chemin. Finalement, la 5G a fait apparaître la notion de network slicing pour laquelle un DAG particulier doit être plongé sur le réseau physique avec des contraintes de délai très strictes [119, 118].

9.2 Publications

Brevets

[B1] Jaideep Chandrashekar, Eve M Schooler, Nina Taft, and Frederic Giroire. Method and system for detecting and reducing botnet activity, dec 2013. US Patent 8,612,579.

[B2] F. Giroire, A. Nucci, N. Taft, and C. Diot. Method and systems for identifying optimal mapping in a network, November 2008. Patent number US 7,453,824 B1.

[B3] F. Giroire, A. Nucci, N. Taft, and C. Diot. Method and systems for correlating practical constraints in a network, July 2008. Sprint, Patent number US 7,394,760 B1.

Conference Proceedings

[L4] F. Giroire and D. Mazauric, editors. *11es Journées Doctorales en Informatique et Réseaux (JDIR 2010)*, number 11, Sophia Antipolis, France, March 2010. Mascotte, INRIA, I3S(CNRS / Univ. of Nice-Sophia).

Chapitre de livre

[ch5] F. Giroire, D. Mazauric, and J. Moulierac. *Energy Efficient Routing by Switching-Off Network Interfaces*, chapter 10 - Energy-Aware Systems and Networking for Sustainable Initiatives, pages 207–236. IGI Global, June 2012.

Journaux internationaux

[J6] Nicolas Huin, Andrea Tomassilli, Frédéric Giroire, and Brigitte Jaumard. Energy-efficient service function chain provisioning. *IEEE/OSA Journal of Optical Communications and Networking*, 10(3):114–124, March 2018.

[J7] Nicolas Huin, Myriana Rifai, Frédéric Giroire, Dino Lopez Pacheco, Guillaume Urvoy-Keller, and Joanna Moulierac. Bringing Energy Aware Routing closer to Reality with SDN Hybrid Networks. *IEEE Transactions on Green Communications and Networking*, 2018. To appear.

[J8] Nicolas Huin, Brigitte Jaumard, and Frédéric Giroire. Optimization of network service chain provisioning. *IEEE/ACM Transactions on Networking (ToN)*, 26(3):1320–1333, June 2018.

[J9] Frédéric Giroire, Frédéric Havet, and Joanna Moulierac. On the complexity of compressing two dimensional routing tables with order. *Algorithmica*, 80(1):209–233, January 2018.

[J10] F. Giroire, N. Huin, J. Moulierac, and K. Phan. Energy-aware routing in software-defined network using compression. *The Computer Journal*, 2018. To appear.

[J11] M. Rifai, N. Huin, C. Caillouet, F. Giroire, Dino Lopez, J. Moulierac, and G. Urvoy-Keller. Minnie: An sdn world with few compressed forwarding rules. *Computer Networks*, 121:185 – 207, 2017.

[J12] Frédéric Giroire, Remigiusz Modrzejewski, Nicolas Nisse, and Stéphane Pérennes. Maintaining Balanced Trees For Structured Distributed Streaming Systems. *Discrete Applied Mathematics*, 2017.

[J13] F. Giroire, S. Perennes, and I. Tahiri. Grid spanners with low forwarding index for energy efficient networks. *Discrete Applied Mathematics*, 2017.

[J14] F. Giroire and J.-C. Maureira. Analysis of the failure tolerance of linear access networks. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–12, 2017.

[J15] Julio Araujo, Frédéric Giroire, Joanna Moulierac, Yaning Liu, and Modrzejewski Remigiusz. Energy efficient content distribution. *Computer Journal*, pages 1–18, 2016. 2017 Wikes Award, rewarding the best paper of Computer Journal in 2016.

[J16] Frédéric Giroire, Stéphane Pérennes, and Issam Tahiri. On the Hardness of Equal Shortest Path Routing. *Networks*, 65(4):344–352, 2015.

[J17] Frédéric Giroire, Joanna Moulierac, Truong Khoa Phan, and Frédéric Roudaut. Minimization of network power consumption with redundancy elimination. *Computer Communications*, 59:98–105, 2015.

[J18] F. Giroire, I. Lamprou, D. Mazauric, N. Nisse, S. Pérennes, and R. Soares. Connected surveillance game. *Theoretical Computer Science*, 584:131 – 143, 2015.

[J19] Fedor Fomin, Frédéric Giroire, Alain Jean-Marie, Dorian Mazauric, and Nicolas Nisse. To Satisfy Impatient Web surfers is Hard. *Theoretical Computer Science*, 526:1–17, 2014.

[J20] S. Caron, F. Giroire, D. Mazauric, J. Monteiro, and S. Pérennes. P2p storage systems: Study of different placement policies. *Peer-to-Peer Networking and Applications*, 7(4):427–443, December 2014.

[J21] J. Araujo, V. Campos, F. Giroire, N. Nisse, L. Sampaio, and R. Soares. On the hull number of some graph classes. *Theoretical Computer Science*, 475:1–12, 2013.

[J22] J. Araujo, N. Cohen, F. Giroire, and F. Havet. Good edge-labelling of graphs. *Discrete Applied Mathematics*, 160(18):2502 – 2513, 2012.

[J23] J. Araujo, J-C. Bermond, F. Giroire, F. Havet, D. Mazauric, and R. Modrzejewski. Weighted improper colouring. *Journal of Discrete Algorithms*, 16:53–66, October 2012.

[J24] D. Coudert, F. Giroire, and I. Sau. Circuits in graphs through a prescribed set of ordered vertices. *Journal of Interconnection Networks*, 11(3-4):121–141, 2011.

[J25] O. Amini, F. Giroire, F. Huc, and S. Pérennes. Minimal selectors and fault tolerant networks. *Networks*, 55(4):326–340, July 2010.

[J26] F. Giroire. Order statistics and estimating cardinalities of massive data sets. *Discrete Applied Mathematics*, 157(2):406–427, 2009.

Soumis à des journaux internationaux

[S27] Frédéric Giroire, Nicolas Huin, and Andrea Tomassilli. The structured way of dealing with heterogeneous live streaming systems.

[S28] F. Giroire, S. Perennes, and I. Tahiri. How to design graphs with low forwarding index and limited number of edges.

[S29] F. Giroire, F. Havet, and J. Moulierac. Compressing two-dimensional routing tables with order: Fixed parameter tractability.

Conférences

Internationales

[Ci30] Frédéric Giroire, Nicolas Huin, Andrea Tomassilli, and Stéphane Pérennes. When network matters: Data center scheduling with network tasks. In *IEEE International Conference on Computer Communications (INFOCOM)*, Paris, France, 2019.

[Ci31] Andrea Tomassilli, Nicolas Huin, Brigitte Jaumard, and Frédéric Giroire. Resource requirements for reliable service function chaining. In *IEEE International Conference on Communications (ICC)*, pages 1–7, Kansas City, Kansas, US, May 2018.

[Ci32] Andrea Tomassilli, Nicolas Huin, Brigitte Jaumard, and Frédéric Giroire. Path protection in optical flexible networks with distance-adaptive modulation formats. In *International Conference on Optical Network Design and Modeling (ONDM)*, pages 30–35, Dublin, Ireland, May 2018.

[Ci33] Andrea Tomassilli, Frédéric Giroire, Nicolas Huin, and Stéphane Pérennes. Provably efficient algorithms for placement of service function chains with ordering constraints. In *IEEE International Conference on Computer Communications (INFOCOM)*, Honolulu, Hawai, US, 2018.

[Ci34] Nicolas Huin, Andrea Tomassilli, Frédéric Giroire, and Brigitte Jaumard. Energy-efficient service function chain provisioning. *Electronic Notes in Discrete Mathematics*, 64:265 – 274, 2018. 8th International Network Optimization Conference - INOC 2017.

[Ci35] Christelle Caillouet, Frederic Giroire, and Tahiry Razafindralambo. Optimization of mobile sensor coverage with uavs. In *The 11th International Workshop on Wireless Sensor, Robot and UAV Networks - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Honolulu, Hawai, US, 2018. To appear.

[Ci36] Nicolas Huin, Myriana Rifai, Frédéric Giroire, Dino Lopez Pacheco, Guillaume Urvoy-Keller, and Joanna Moulierac. Bringing Energy Aware Routing closer to Reality with SDN Hybrid Networks. In *IEEE Global Communications Conference (GLOBECOM)*, Singapore, Dec 2017.

[Ci37] Nicolas Huin, Brigitte Jaumard, and Frédéric Giroire. Optimization of network service chain provisioning. In *IEEE International Conference on Communications (ICC)*, Paris, France, 2017.

[Ci38] F. Giroire and J.-C. Maureira. Analysis of the failure tolerance of linear access networks. In *IEEE Global Communications Conference (GLOBECOM)*, Washington, US, Dec 2016.

[Ci39] M. Rifai, N. Huin, C. Caillouet, F. Giroire, Dino Lopez, J. Moulierac, and G. Urvoy-Keller. Too many sdn rules? compress them with minnie. In *IEEE Global Communications Conference (GLOBECOM)*, San Diego, US, Dec 2015.

[Ci40] F. Giroire, S. Perennes, and I. Tahiri. How to design graphs with low forwarding index and limited number of edges. In *26th International Workshop on Combinatorial Algorithms (IWOCA)*, pages 221–234, Verona, Italy, Oct 2015. Springer.

[Ci41] F. Giroire, S. Perennes, and I. Tahiri. Grid spanners with low forwarding index for energy efficient networks. In *International Network Optimization Conference (INOC)*, Warsaw, Poland, May 2015.

[Ci42] F. Giroire and N. Huin. Study of repair protocols for live video streaming distributed systems. In *IEEE Global Communications Conference (GLOBECOM)*, San Diego, US, Dec 2015.

[Ci43] F. Giroire, F. Havet, and J. Moulierac. Compressing two-dimensional routing tables with order. In *International Network Optimization Conference (INOC)*, Warsaw, Poland, May 2015.

[Ci44] F. Giroire, J. Moulierac, and K. Phan. Optimizing rule placement in software-defined networks for energy-aware routing. In *IEEE Global Communications Conference (GLOBECOM)*, Austin, United States, December 2014.

[Ci45] Esther Le Rouzic, Edoardo Bonetto, Luca Chiaraviglio, Frédéric Giroire, Filip Idzikowski, Felipe Jimenez, Christoph Lange, Julio Montalvo, Francesco Musumeci, Issam Tahiri, Alessandro Valenti, Ward Van Heddeghem, Yabin Ye, Andrea Bianco, and Achille Pattavina. Trend towards more energy-efficient optical networks. In *Conference on Optical Network Design and Modelling (ONDM)*, pages 211–216, Brest, France, 2013.

[Ci46] Remigiusz Modrzejewski, Luca Chiaraviglio, Issam Tahiri, Frederic Giroire, Esther Le Rouzic, Edoardo Bonetto, Francesco Musumeci, Roberto Gonzalez, and Carmen Guerrero. Energy Efficient Content Distribution in an ISP Network. In *IEEE Global Communications Conference (GLOBECOM)*, Atlanta, United States, December 2013.

[Ci47] Frédéric Giroire, Stéphane Pérennes, and Issam Tahiri. On the Hardness of Equal Shortest Path Routing. In *International Network Optimization Conference (INOC)*, volume 41, pages 439–446, 2013.

[Ci48] Frédéric Giroire, Remigiusz Modrzejewski, Nicolas Nisse, and Stéphane Pérennes. Maintaining Balanced Trees For Structured Distributed Streaming Systems. In *20th Colloquium on Structural Information and Communication Complexity (SIROCCO)*, pages 177–188, Ischia, Italy, 2013. Springer.

[Ci49] Frédéric Giroire, Sandeep Kumar Gupta, Remigiusz Modrzejewski, Julian Monteiro, and Stéphane Pérennes. Repair time in distributed storage systems. In *Data Management in Cloud, Grid and P2P Systems (Globe)*, pages 99–110, Prague, Czech Republic, 2013. Springer.

[Ci50] Julio Araujo, Frédéric Giroire, Yaning Liu, Modrzejewski Remigiusz, and Joanna Moulierac. Energy Efficient Content Distribution. In *IEEE International Conference on Communications (ICC)*, pages 4233–4238, Budapest, Hungary, June 2013. IEEE.

[Ci51] F. Giroire, J. Moulierac, T.K. Phan, and F. Roudaut. Minimization of network power consumption with redundancy elimination. In *IFIP Networking*, pages 247–258, Prague, Czech Republic, May 2012. Springer.

[Ci52] F. V. Fomin, F. Giroire, A. Jean-Marie, D. Mazauric, and N. Nisse. To satisfy impatient web surfers is hard. In *6th International Conference on FUN with Algorithms (FUN)*, volume 7288, pages 166–176. Springer, LNCS, 2012.

[Ci53] J. Araujo, F. Giroire, and J. Monteiro. Hybrid approaches for distributed storage systems. In *Proceedings of Fourth International Conference on Data Management in Grid and P2P Systems (Globe'11)*, Toulouse, France, September 2011.

[Ci54] J. Araujo, V. Campos, F. Giroire, L. Sampaio, and R. Soares. On the hull number of some graph classes. In *Proceedings of European Conference on Combinatorics, Graph Theory and Applications (EuroComb)*, volume 38 of *Electronic Notes in Discrete Mathematics*, pages 49–55, Budapest, Hungary, September 2011.

[Ci55] J Araujo, J-C Bermond, F. Giroire, F. Havet, D. Mazauric, and R. Modrzejewski. Weighted improper colouring. In *Proceedings of International Workshop on Combinatorial Algorithms (IWOCA'11)*, volume 7056 of *Lecture Notes in Computer Science*, pages 1–18, Victoria, Canada, June 2011. Springer-Verlag.

[Ci56] F. Giroire, J. Monteiro, and S. Pérennes. Peer-to-peer storage systems: a practical guideline to be lazy. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Miami, EUA, 12 2010.

[Ci57] F. Giroire, D. Mazauric, J. Moulierac, and B. Onfroy. Minimizing routing energy consumption: from theoretical to practical results. In *IEEE/ACM International Conference on Green Computing and Communications (GreenCom)*, page 8, Hangzhou, China, 2010.

[Ci58] S. Caron, F. Giroire, D. Mazauric, J. Monteiro, and S. Pérennes. Data life time for different placement policies in p2p storage systems. In *Proceedings of the 3rd Intl. Conference on Data Management in Grid and P2P Systems (Globe)*, volume 6265 of *Lecture Notes in Computer Science*, pages 75–88, Bilbao, Spain, September 2010.

[Ci59] F. Giroire, J. Monteiro, and S. Pérennes. P2P storage systems: How much locality can they tolerate? In *Proceedings of the 34th IEEE Conference on Local Computer Networks (LCN)*, pages 320–323, Oct 2009.

[Ci60] F. Giroire, J. Chandrashekar, N. Taft, E. Schooler, and K. Papagiannaki. Exploiting temporal persistence to detect covert botnet channels. In Springer, editor, *12th International Symposium on Recent Advances in Intrusion Detection (RAID)*, volume 5758 of *Lecture Notes in Computer Science*, pages 326–345, Saint Malo, France, September 2009.

[Ci61] O. Dalle, F. Giroire, J. Monteiro, and S. Pérennes. Analysis of failure correlation impact on peer-to-peer storage systems. In *Proceedings of the 9th IEEE International Conference on Peer-to-Peer Computing (P2P)*, pages 184–193, sep 2009.

[Ci62] D. Coudert, F. Giroire, and I. Sau. Edge-simple circuits through 10 ordered vertices in square grids. In J. Kratochvíl and M. Miller, editors, *20th International Workshop on Combinatorial Algorithms – IWOCA*, volume 5874 of *Lecture Notes in Computer Science*, pages 134–145, Hradec nad Moravicí, Czech Republic, June 2009. Springer.

[Ci63] D. Barman, J. Chandrashekar, N. Taft, M. Faloutsos, L. Huang, and F. Giroire. Impact of it monoculture on behavioral end host intrusion detection. In *ACM SIGCOMM Workshop on Research on Enterprise Networking – WREN*, pages 27–36, Barcelona, Spain, August 2009. ACM.

[Ci64] J. Araujo, N. Cohen, F. Giroire, and F. Havet. Good edge-labelling of graphs. In *proceedings of the Latin-American Algorithms, Graphs and Optimization Symposium (LAGOS)*, volume 35 of *Electronic Notes in Discrete Mathematics*, pages 275–280, Gramado, Brazil, December 2009. Springer.

[Ci65] F. Giroire, J. Chandrashekar, G. Iannaccone, D. Papagiannaki, E. Schooler, and N. Taft. The cubicle vs. the coffee shop: Behavioral modes in enterprise end-users. In *Proceeding of the Passive and Active Monitoring conference (PAM)*, volume 4979 of *Lecture Notes in Computer Science*, pages 202–211. Springer, 2008.

[Ci66] Éric Fusy and Frécéric Giroire. Estimating the number of active flows in a data stream over a sliding window. In *2007 Proceedings of the Fourth Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*, pages 223–231. SIAM, 2007.

[Ci67] J-C. Bermond, F. Giroire, and S. Pérennes. Design of minimal fault tolerant on-board networks : Practical constructions. In *14th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 4474 of *Lecture Notes in Computer Sciences*, pages 261–273, Castiglioncello, Italy, June 2007.

[Ci68] F. Giroire. Order statistics and estimating cardinalities of massive data sets. In Conrado Martinez, editor, *2005 International Conference on Analysis of Algorithms*, volume AD of *DMTCS Proceedings*, pages 157–166. Discrete Mathematics and Theoretical Computer Science, 2005.

[Ci69] F. Giroire, A. Nucci, N. Taft, and C. Diot. Increasing the robustness of ip backbones in the absence of optical level protection. In *Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2003.

Nationales

[Cn70] Andrea Tomassilli, Frédéric Giroire, Nicolas Huin, and Stéphane Pérennes. Algorithmes d'approximation pour le placement de chaînes de fonctions de services avec des contraintes d'ordre. In *ALGOTEL 2018 - 20èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, Roscoff, France, 2018.

[Cn71] Nicolas Huin, Brigitte Jaumard, and Frédéric Giroire. Optimisation pour le provisionnement de chaînes de services réseau. In *ALGOTEL 2018 - 20èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, Roscoff, France, 2018. To appear.

[Cn72] Christelle Caillouet, Frederic Giroire, and Tahiry Razafindralambo. Déploiement efficace de drones pour la collecte de données de capteurs mobiles. In *CoRes*, Roscoff, France, 2018. To appear.

[Cn73] Myriana Rifai, Nicolas Huin, Christelle Caillouet, Frédéric Giroire, Joanna Moulierac, Lopez Pacheco, Dino, and Guillaume Urvoy-Keller. Minnie : enfin un monde sdn sans (trop de) règles. In *18es Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel) - court*, Bayonne, France, 2016.

[Cn74] Frédéric Giroire and Nicolas Huin. Etude d'un système distribué de diffusion de vidéo en direct. In *18es Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel) - court*, Bayonne, France, 2016.

[Cn75] Frédéric Giroire, Stéphane Pérennes, and Issam Tahiri. Sur la complexité du routage OSPF. In Nicolas Nisse, Franck Rousseau, and Yann Busnel, editors, *15èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel)*, pages 1–4, Pornic, France, 2013.

[Cn76] F. V. Fomin, F. Giroire, A. Jean-Marie, D. Mazauric, and N. Nisse. Satisfaire un internaute impatient est difficile. In *14es Rencontres Francophones sur les Aspects Algorithmiques de Télécommunications (AlgoTel)*, pages 79–82, 2012. Best paper award.

[Cn77] F. Giroire, D. Mazauric, and J. Moulierac. Routage efficace en énergie. In Ducourthial et Bertrand et Felber et Pascal, editor, *13es Rencontres Francophones sur les Aspects Algorithmiques de Télécommunications (AlgoTel)*, Cap Estérel, France, 2011.

[Cn78] S. Caron, F. Giroire, D. Mazauric, J. Monteiro, and S. Pérennes. P2P Storage Systems: Data Life Time for Different Placement Policies. In Maria Gradinariu Potop-Butucaru et Hervé Rivano, editor, *12èmes Rencontres Francophones sur les Aspects Algorithmiques de Télécommunications (AlgoTel)*, page 4p, Belle Dune France, 2010.

[Cn79] O. Dalle, F. Giroire, J. Monteiro, and S. Pérennes. Analyse des corrélations entre pannes dans les systèmes de stockage pair-à-pair. In Augustin Chaintreau and Clemence Magnien, editors, *11ème rencontres francophones sur les Aspects Algorithmiques des Télécommunications (Algotel'2009)*, page 4p, Carry-Le-Rouet France, 2009. Best Student Paper Award.

[Cn80] F. Giroire. Directions to use probabilistic algorithms for cardinality for DNA analysis. In *Journées Ouvertes Biologie Informatique Mathématiques (JOBIM 2006)*, pages 3–5, July 2006.

[Cn81] O. Amini, J-C. Bermond, F. Giroire, F. Huc, and S. Pérennes. Design of minimal fault tolerant networks: Asymptotic bounds. In *Huitièmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel'06)*, pages 37–40, Trégastel, France, May 2006.

Thèse

[T82] F. Giroire. *Réseaux, algorithmique et analyse combinatoire de grands ensembles.* PhD thesis, Université Paris VI, November 2006.

Rapports de recherche

[R83] Nicolas Huin, Myriana Rifai, Frédéric Giroire, Dino Lopez Pacheco, Guillaume Urvoy-Keller, and Joanna Moulierac. Bringing Energy Aware Routing closer to Reality with SDN Hybrid Networks. Research Report RR-9020, INRIA Sophia Antipolis - I3S, January 2017.

[R84] Frédéric Giroire, Nicolas Huin, and Andrea Tomassilli. The Structured Way of Dealing with Heterogeneous Live Streaming Systems. Research Report RR-9070, Inria Sophia Antipolis ; Cnrs ; Universite Cote d'Azur, June 2017.

[R85] Frédéric Giroire, Stéphane Pérennes, and Issam Tahiri. Graphs with optimal forwarding indices: What is the best throughput you can get with a given number of edges? Research Report RR-8752, INRIA Sophia Antipolis ; INRIA, June 2015.

[R86] F. Giroire, D. Mazauric, J. Moulierac, and B. Onfroy. Minimizing Routing Energy Consumption: from theoretical to practical results. Research Report inria-00464318, May 2010.

[R87] S. Caron, F. Giroire, D. Mazauric, J. Monteiro, and S. Pérennes. P2P Storage Systems: Data Life Time for Different Placement Policies. Research Report RR-7209, INRIA, February 2010.

[R88] F. Giroire, J. Monteiro, and S. Pérennes. P2P storage systems: How much locality can they tolerate? Research Report RR-7006, INRIA, July 2009.

[R89] D. Coudert, F. Giroire, and I. Sau. Circuit visiting 10 ordered vertices in infinite grids. Technical Report RR-6910, INRIA, 2009.

[R90] J. Araujo, N. Cohen, F. Giroire, and F. Havet. Good edge-labelling of graphs. Research Report 6934, INRIA, 2009.

[R91] O. Dalle, F. Giroire, J. Monteiro, and S. Pérennes. Analysis of failure correlation in peer-to-peer storage systems. Technical Report RR-6761, INRIA, December 2008.

[R92] F. Giroire, J. Chandrashekar, N. Taft, G. Iannaccone, T. Karagiannis, K. Papagiannaki, and E. Schooler. The case for personalizing end-host detectors. Technical report, Intel Research, 2007.

[R93] F. Giroire, J. Chandrashekar, G. Iannaccone, T. Karagiannis, K. Papagiannaki, E. Schooler, and N. Taft. Inside the forbidden city: A look at end-host traffic inside a modern enterprise. Technical report, Intel Research, 2007.

[R94] O. Amini, F. Giroire, F. Huc, and S. Pérennes. Minimal selectors and fault tolerant networks. Research report, INRIA Research Report HAL-00082015, July 2006.

Références

[95] http://www.riverbed.com/us/solutions/wan_optimization/.

[96] www-sop.inria.fr/mascotte/rapports_stages/KhoaPhan_internship-2011.pdf.

[97] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 1972.

[98] G.B. Dantzig. *Linear programming and extensions*. Princeton Univ Pr, 1963.

[99] www-01.ibm.com/software/integration/optimization/cplex-optimizer/.

[100] CISCO. *Best Practices in Core Network Capacity Planning*, 2013.

[101] V. Chvatal. *Linear Programming*. Freeman, 1983.

[102] "BlueCoat: WAN Optimization". http://www.bluecoat.com/.

[103] Frédéric Giroire, Nicolas Huin, Andrea Tomassilli, and Stéphane Pérennes. When network matters: Data center scheduling with network tasks. In *IEEE International Conference on Computer Communications (INFOCOM)*, Paris, France, 2019.

[104] Andrea Tomassilli, Nicolas Huin, Brigitte Jaumard, and Frédéric Giroire. Resource requirements for reliable service function chaining. In *IEEE International Conference on Communications (ICC)*, pages 1–7, Kansas City, Kansas, US, May 2018.

[105] Andrea Tomassilli, Nicolas Huin, Brigitte Jaumard, and Frédéric Giroire. Path protection in optical flexible networks with distance-adaptive modulation formats. In *International Conference on Optical Network Design and Modeling (ONDM)*, pages 30–35, Dublin, Ireland, May 2018.

[106] Andrea Tomassilli, Frédéric Giroire, Nicolas Huin, and Stéphane Pérennes. Provably efficient algorithms for placement of service function chains with ordering constraints. In *IEEE International Conference on Computer Communications (INFOCOM)*, Honolulu, Hawai, US, 2018.

[107] Andrea Tomassilli, Frédéric Giroire, Nicolas Huin, and Stéphane Pérennes. Algorithmes d'approximation pour le placement de chaînes de fonctions de services avec des contraintes d'ordre. In *ALGOTEL 2018 - 20èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, Roscoff, France, 2018.

[108] Oussama Soualah, Marouen Mechtri, Chaima Ghribi, and Djamal Zeghlache. A green vnfs placement and chaining algorithm. In *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2018.

[109] Nicolas Huin, Andrea Tomassilli, Frédéric Giroire, and Brigitte Jaumard. Energy-efficient service function chain provisioning. *Electronic Notes in Discrete Mathematics*, 64:265 – 274, 2018. 8th International Network Optimization Conference - INOC 2017.

[110] Nicolas Huin, Andrea Tomassilli, Frédéric Giroire, and Brigitte Jaumard. Energy-efficient service function chain provisioning. *IEEE/OSA Journal of Optical Communications and Networking*, 10(3):114–124, March 2018.

[111] Nicolas Huin, Myriana Rifai, Frédéric Giroire, Dino Lopez Pacheco, Guillaume Urvoy-Keller, and Joanna Moulierac. Bringing Energy Aware Routing closer to Reality with SDN Hybrid Networks. *IEEE Transactions on Green Communications and Networking*, 2018. To appear.

[112] Nicolas Huin, Brigitte Jaumard, and Frédéric Giroire. Optimization of network service chain provisioning. *IEEE/ACM Transactions on Networking (ToN)*, 26(3):1320–1333, June 2018.

[113] Nicolas Huin, Brigitte Jaumard, and Frédéric Giroire. Optimisation pour le provisionnement de chaînes de services réseau. In *ALGOTEL 2018 - 20èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, Roscoff, France, 2018. To appear.

[114] Frédéric Giroire, Frédéric Havet, and Joanna Moulierac. On the complexity of compressing two dimensional routing tables with order. *Algorithmica*, 80(1):209–233, January 2018.

[115] F. Giroire, N. Huin, J. Moulierac, and K. Phan. Energy-aware routing in software-defined network using compression. *The Computer Journal*, 2018. To appear.

[116] Christelle Caillouet, Frederic Giroire, and Tahiry Razafindralambo. Optimization of mobile sensor coverage with uavs. In *The 11th International Workshop on Wireless Sensor, Robot and UAV Networks - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Honolulu, Hawai, US, 2018. To appear.

[117] Christelle Caillouet, Frederic Giroire, and Tahiry Razafindralambo. Déploiement efficace de drones pour la collecte de données de capteurs mobiles. In *CoRes*, Roscoff, France, 2018. To appear.

[118] Haijun Zhang, Na Liu, Xiaoli Chu, Keping Long, Abdol-Hamid Aghvami, and Victor CM Leung. Network slicing based 5g and future mobile networks: mobility, resource management, and challenges. *IEEE Communications Magazine*, 55(8):138–145, 2017.

[119] Spyridon Vassilaras, Lazaros Gkatzikis, Nikolaos Liakopoulos, Ioannis N Stiakogiannakis, Meiyu Qi, Lei Shi, Liu Liu, Merouane Debbah, and Georgios S Paschos. The algorithmic aspects of network slicing. *IEEE Communications Magazine*, 55(8):112–119, 2017.

[120] Yu Sang, Bo Ji, Gagan R Gupta, Xiaojiang Du, and Lin Ye. Provably efficient algorithms for joint placement and allocation of virtual network functions. In *Proceedings of IEEE INFOCOM, 2017*, 2017.

[121] M. Rifai, N. Huin, C. Caillouet, F. Giroire, Dino Lopez, J. Moulierac, and G. Urvoy-Keller. Minnie: An sdn world with few compressed forwarding rules. *Computer Networks*, 121:185 – 207, 2017.

[122] Nicolas Huin, Myriana Rifai, Frédéric Giroire, Dino Lopez Pacheco, Guillaume Urvoy-Keller, and Joanna Moulierac. Bringing Energy Aware Routing closer to Reality with SDN Hybrid Networks. In *IEEE Global Communications Conference (GLOBECOM)*, Singapore, Dec 2017.

[123] Nicolas Huin, Brigitte Jaumard, and Frédéric Giroire. Optimization of network service chain provisioning. In *IEEE International Conference on Communications (ICC)*, Paris, France, 2017.

[124] Frédéric Giroire, Remigiusz Modrzejewski, Nicolas Nisse, and Stéphane Pérennes. Maintaining Balanced Trees For Structured Distributed Streaming Systems. *Discrete Applied Mathematics*, 2017.

[125] Frédéric Giroire, Nicolas Huin, and Andrea Tomassilli. The Structured Way of Dealing with Heterogeneous Live Streaming Systems. Research Report RR-9070, Inria Sophia Antipolis ; Cnrs ; Universite Cote d'Azur, June 2017.

[126] F. Giroire, S. Perennes, and I. Tahiri. Grid spanners with low forwarding index for energy efficient networks. *Discrete Applied Mathematics*, 2017.

[127] F. Giroire and J.-C. Maureira. Analysis of the failure tolerance of linear access networks. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–12, 2017.

[128] S. D'Oro, L. Galluccio, S. Palazzo, and G. Schembra. Exploiting congestion games to achieve distributed service chaining in nfv networks. *IEEE Journal on Selected Areas in Communications*, 35(2):407–420, Feb 2017.

[129] Anders Andrae. Total consumer power consumption forecast, 10 2017.

[130] Sdn and nfv transforming the network: where do we go from here? [online]. available: https://www.orange-business.com/en/blogs/connecting-technology/networks/sdn-and-nfv-transforming-the- 2017.

[131] Huawei releases sdn/nfv commercial and technological innovations. [on- line]. available: http://www.huawei.com/en/press-events/news/2017/10/Huawei-SDN-NFV-Commercial-Technological-Innovatio 2017.

[132] Cisco visual networking index: Forecast and methodology, 2016–2021, 2017.

[133] At&t's stephens: More than 40are virtualized. [online]. available: https://www.sdxcentral.com/articles/news/atts-stephens-47-network-functions-virtualized/2017/07/, 2017.

[134] Sachin Sharma, Dimitri Staessens, Didier Colle, Mario Pickavet, and Piet Demeester. In-band control, queuing, and failure recovery functionalities for openflow. *IEEE Network*, 30(1), 2016.

[135] Ori Rottenstreich, Isaac Keslassy, Avinatan Hassidim, Haim Kaplan, and Ely Porat. Optimal in/out tcam encodings of ranges. *IEEE/ACM Transactions on Networking*, 24(1):555–568, 2016.

[136] Myriana Rifai, Nicolas Huin, Christelle Caillouet, Frédéric Giroire, Joanna Moulierac, Lopez Pacheco, Dino, and Guillaume Urvoy-Keller. Minnie : enfin un monde sdn sans (trop de) règles. In *18es Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel) - court*, Bayonne, France, 2016.

[137] B. Ozbek, Y. Aydogmus, A. Ulaş, B. Gorkemli, and K. Ulusoy. Energy aware routing and traffic management for software defined networks. In *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, pages 73–77, June 2016.

[138] Mathis Obadia, Jean-Louis Rougier, Luigi Iannone, Vania Conan, and Mathieu Brouet. Revisiting nfv orchestration with routing games. In *Proceedings of IEEE NFV-SDN*, 2016.

[139] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba. Network function virtualization: State-of-the-art and research challenges. *IEEE Communications Surveys & Tutorials*, 18(1):236 –262, 2016.

[140] Marouen Mechtri, Chaima Ghribi, and Djamal Zeghlache. A scalable algorithm for the placement of service function chains. *IEEE Transactions on Network and Service Management*, 13(3):533–546, 2016.

[141] Kirill Kogan, Sergey I Nikolenko, Ori Rottenstreich, William Culhane, and Patrick Eugster. Exploiting order independence for scalable and expressive packet classification. *IEEE/ACM Transactions on Networking*, 24(2):1251–1264, 2016.

[142] Naga Katta, Omid Alipourfard, Jennifer Rexford, and David Walker. Cacheflow: Dependency-aware rule-caching for software-defined networks. In *Proc. ACM Symposium on SDN Research (SOSR)*, 2016.

[143] Filip Idzikowski, Luca Chiaraviglio, Antonio Cianfrani, Jorge López Vizcaíno, Marco Polverini, and Yabin Ye. A survey on energy-aware design and operation of core networks. *IEEE Communications Surveys & Tutorials*, 18(2), 2016.

[144] J.G. Herrera and J.F. Botero. Resource allocation in NFV: A comprehensive survey. *IEEE Transactions on Network and Service Management*, 13(3):32–40, March 2016.

[145] Frédéric Giroire and Nicolas Huin. Etude d'un système distribué de diffusion de vidéo en direct. In *18es Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel) - court*, Bayonne, France, 2016.

[146] Frédéric Giroire, Frédéric Havet, and Joanna Moulierac. On the complexity of compressing two-dimensional routing tables with order. *Algorithmica*, pages 1–25, 2016.

[147] F. Giroire and J.-C. Maureira. Analysis of the failure tolerance of linear access networks. In *IEEE Global Communications Conference (GLOBECOM)*, Washington, US, Dec 2016.

[148] A. Gember, P. Prabhu, Z. Ghadiyali, and A. Akella. Toward software-defined middlebox networking. In *ACM Workshop on Hot Topics in Networks (HotNets)*, pages 7–12, 2016.

[149] A. Fernandez-Fernandez, C. Cervello-Pastor, and L. Ochoa-Aday. Achieving energy efficiency: An energy-aware approach in sdn. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, Dec 2016.

[150] A. Dwaraki and T. Wolf. Adaptive service-chain routing for virtual network functions in software-defined networks. In *Workshop on Hot topics in Middleboxes and Network Function Virtualization (HotMIddlebox)*, pages 32–37, 2016.

[151] Md.F. Bari, S.R. Chowdhury, R. Ahmed, R. Boutaba, and O.C.M.B. Duarte. Orchestrating virtualized network functions. *IEEE Transactions on Network and Service Management*, PP:1–1, 2016.

[152] Mohamad Khattar Awad, Yousef Rafique, Sarah Alhadlaq, Dunya Hassoun, Asmaa Alabdulhadi, and Sheikha Thani. A greedy power-aware routing algorithm for software-defined networks. In *2016 IEEE International Symposium on Signal Processing and Information Technology, ISSPIT 2016, Limassol, Cyprus, December 12-14, 2016*, pages 268–273, 2016.

[153] Julio Araujo, Frédéric Giroire, Joanna Moulierac, Yaning Liu, and Modrzejewski Remigiusz. Energy efficient content distribution. *Computer Journal*, pages 1–18, 2016.

[154] Sandvine global internet phenomena. Technical report, 2016.

[155] Rahul Vaze. Random wireless networks, 2015.

[156] Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer, 2015.

[157] Marco Savi, Massimo Tornatore, and Giacomo Verticale. Impact of processing costs on service chain placement in network functions virtualization. In *Proceedings of IEEE NFV-SDN*, 2015.

[158] M. Savi, M. Tornatore, and G. Verticale. Impact of processing costs on service chain placement in network functions virtualization. In *IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)*, pages 191–197, Nov. 2015.

[159] Ori Rottenstreich et al. Lossy compression of packet classifiers. In *Proceedings of the Eleventh ACM/IEEE Symposium on Architectures for networking and communications systems*, pages 39–50. IEEE Computer Society, 2015.

[160] R. Riggio, A. Bradai, T. Rasheed, J. Schulz-Zander, S. Kuklinski, and T. Ahmed. Virtual network functions orchestration in wireless networks. In *International Conference on Network and Service Management (CNSM)*, pages 108–116, 2015.

[161] M. Rifai, N. Huin, C. Caillouet, F. Giroire, Dino Lopez, J. Moulierac, and G. Urvoy-Keller. Too many sdn rules? compress them with minnie. In *IEEE Global Communications Conference (GLOBECOM)*, San Diego, US, Dec 2015.

[162] Paul Quinn and Tom Nadeau. Problem statement for service function chaining. 2015.

[163] Xuan-Nam Nguyen, Damien Saucez, Chadi Barakat, and Thierry Turletti. OFFICER: A general Optimization Framework for OpenFlow Rule Allocation and Endpoint Policy Enforcement. In *INFOCOM*, pages 478–486. IEEE, April 2015.

[164] Index Cisco Visual Networking. Cisco visual networking index: Forecast and methodology 2015-2020. *White paper, CISCO)*, 2015.

[165] A. Mohammadkhan, S. Ghapani, G. Liu, W. Zhang, K.K. Ramakrishnan, and T. Wood. Virtual function placement and traffic steering in flexible and dynamic software defined networks. In *IEEE International Workshop on Local and Metropolitan Area Networks*, pages 1–6, 2015.

[166] Rashid Mijumbi. On the energy efficiency prospects of network function virtualization. *CoRR*, abs/1512.00215, 2015.

[167] Markus Meringer. Small cubic graphs, 2015. Flinders Univ projet.

[168] B. Martini, F. Paganelli, P. Cappanera, S. Turchi, and P. Castoldi. Latency-aware composition of virtual functions in 5G. In *IEEE Conference on Network Softwarization (NetSoft)*, pages 1–6, 2015.

[169] M.C. Luizelli, L.R. Bays, L.S. Buriol, M.P. Barcellos, and L.P. Gaspary. Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions. In *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 98–106, 2015.

[170] Y. Li and M. Cheng. Software-defined network function virtualization: A survey. *IEEE Access*, 3:2542 – 2553, 2015.

[171] Stanislav Lange, Steffen Gebert, Thomas Zinner, Phuoc Tran-Gia, David Hock, Michael Jarschel, and Marco Hoffmann. Heuristic approaches to the controller placement problem in large scale sdn networks. *IEEE Transactions on Network and Service Management*, 12(1):4–17, 2015.

[172] Dave Katz and David Ward. Bidirectional Forwarding Detection (BFD). RFC 5880, October 2015.

[173] Shuihai Hu, Kai Chen, Haitao Wu, Wei Bai, Chang Lan, Hao Wang, Hongze Zhao, and Chuanxiong Guo. Explicit path control in commodity data centers: Design and applications. In *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation*, NSDI'15, pages 15–28, Berkeley, CA, USA, 2015. USENIX Association.

[174] Bo Han, Vijay Gopalakrishnan, Lusheng Ji, and Seungjoon Lee. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine*, 53(2):90–97, 2015.

[175] A. Gupta, M.F. Habib, P. Chowdhury, M. Tornatore, and B. Mukherjee. Joint virtual network function placement and routing of traffic in operator networks. Technical report, UC Davis, Davis, CA, USA, 2015.

[176] Frédéric Giroire, Stéphane Pérennes, and Issam Tahiri. On the Hardness of Equal Shortest Path Routing. *Networks*, 65(4):344–352, 2015.

[177] Frédéric Giroire, Stéphane Pérennes, and Issam Tahiri. Graphs with optimal forwarding indices: What is the best throughput you can get with a given number of edges? Research Report RR-8752, INRIA Sophia Antipolis ; INRIA, June 2015.

[178] Frédéric Giroire, Joanna Moulierac, Truong Khoa Phan, and Frédéric Roudaut. Minimization of network power consumption with redundancy elimination. *Computer Communications*, 59:98–105, 2015.

[179] F. Giroire, S. Perennes, and I. Tahiri. How to design graphs with low forwarding index and limited number of edges. In *26th International Workshop on Combinatorial Algorithms (IWOCA)*, pages 221–234, Verona, Italy, Oct 2015. Springer.

[180] F. Giroire, S. Perennes, and I. Tahiri. Grid spanners with low forwarding index for energy efficient networks. In *International Network Optimization Conference (INOC)*, Warsaw, Poland, May 2015.

[181] F. Giroire, I. Lamprou, D. Mazauric, N. Nisse, S. Pérennes, and R. Soares. Connected surveillance game. *Theoretical Computer Science*, 584:131 – 143, 2015.

[182] F. Giroire and N. Huin. Study of repair protocols for live video streaming distributed systems. In *IEEE Global Communications Conference (GLOBECOM)*, San Diego, US, Dec 2015.

[183] F. Giroire, F. Havet, and J. Moulierac. Compressing two-dimensional routing tables with order. In *INOC*, pages 1–8, 2015.

[184] F. Giroire, F. Havet, and J. Moulierac. Compressing two-dimensional routing tables with order. In *International Network Optimization Conference (INOC)*, Warsaw, Poland, May 2015.

[185] Rami Cohen, Liane Lewin-Eytan, Joseph Seffi Naor, and Danny Raz. Near optimal placement of virtual network functions. In *Proceedings of IEEE INFOCOM*, 2015.

[186] Cing-Yu Chu, Kang Xi, Min Luo, and H Jonathan Chao. Congestion-aware single link failure recovery in hybrid sdn networks. In *IEEE INFOCOM*, 2015.

[187] Beakal Gizachew Assefa and Oznur Ozkasap. State-of-the-art energy efficiency approaches in software defined networking. *ICN 2015*, x:268, 2015.

[188] Anders SG Andrae and Tomas Edler. On global electricity usage of communication technology: trends to 2030. *Challenges*, 6(1):117–157, 2015.

[189] Service chaining in carrier networks. http://www.qosmos.com/wp-content/uploads/2015/02/Service-Chaining-in-Carrier-Networks_WP_Heavy-Reading_Qosmos_Feb2015.pdf, February 2015.

[190] R. Wang, Z. Jiang, S. Gao, W. Yang, Y. Xia, and M. Zhu. Energy-aware routing algorithms in software-defined networks. In *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, pages 1–6, June 2014.

[191] Stefano Vissicchio, Laurent Vanbever, and Olivier Bonaventure. Opportunities and research challenges of hybrid software defined networks. *ACM SIGCOMM Computer Communication Review*, 44(2), 2014.

[192] Ward Van Heddeghem, Sofie Lambert, Bart Lannoo, Didier Colle, Mario Pickavet, and Piet Demeester. Trends in worldwide ict electricity consumption from 2007 to 2012. *Computer Communications*, 50:64–76, 2014.

[193] Kevin Phemius, Mathieu Bouet, and Jeremie Leguay. Disco: Distributed multi-domain sdn controllers. In *Network Operations and Management Symposium (NOMS), 2014 IEEE*, pages 1–4. IEEE, 2014.

[194] Maciej Kuzniar, P Perešıni, and Dejan Kostic. What you need to know about sdn control and data planes. *EPFL, TR*, 199497, 2014.

[195] Anh Hoang, Phuc Do, and Benoît Iung. Integrating energy efficiency-based prognostic approaches into energy management systems of base stations. In *Advanced Technologies for Communications (ATC), 2014 International Conference on*, pages 220–225. IEEE, 2014.

[196] F. Giroire, J. Moulierac, and T.K. Phan. Optimizing rule placement in software-defined networks for energy-aware routing. In *GLOBECOM*, pages 1–6. IEEE, 2014.

[197] F. Giroire, J. Moulierac, and K. Phan. Optimizing rule placement in software-defined networks for energy-aware routing. In *IEEE Global Communications Conference (GLOBECOM)*, Austin, United States, December 2014.

[198] Fedor Fomin, Frédéric Giroire, Alain Jean-Marie, Dorian Mazauric, and Nicolas Nisse. To Satisfy Impatient Web surfers is Hard. *Theoretical Computer Science*, 526:1–17, 2014.

[199] Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing*, STOC '14, 2014.

[200] R. Cohen, L. Lewin-Eytan, J.S. Naor, and D. Raz. On the effect of forwarding table size on sdn network utilization. In *INFOCOM*, pages 1734–1742. IEEE, 2014.

[201] S. Caron, F. Giroire, D. Mazauric, J. Monteiro, and S. Pérennes. P2p storage systems: Study of different placement policies. *Peer-to-Peer Networking and Applications*, 7(4):427–443, December 2014.

[202] Niklas Carlsson, Derek L. Eager, Ajay Gopinathan, and Zongpeng Li. Caching and optimized request routing in cloud-based content delivery systems. *Perform. Eval.*, 79:38–55, 2014.

[203] W. Braun and M. Menth. Wildcard compression of inter-domain routing tables for openflow-based software-defined networking. In *Software Defined Networks (EWSDN), 2014 Third European Workshop on*, pages 25–30, Sept 2014.

[204] R. Bolla, C. Lombardo, R. Bruschi, and S. Mangialardi. DROPv2: energy efficiency through network function virtualization. *IEEE Network*, 28(2):26–32, 2014.

[205] S. Banerjee and K. Kannan. Tag-in-tag: Efficient flow table management in sdn switches. In *CNSM*, pages 109–117, 2014.

[206] Guoqiang Zhang, Yang Li, and Tao Lin. Caching in information centric networking: A survey. *Computer Network*, 57(16):3128–3141, November 2013.

[207] Soheil Hassas Yeganeh, Amin Tootoonchian, and Yashar Ganjali. On scalability of software-defined networking. *IEEE Communications Magazine*, 51(2):136–141, 2013.

[208] Ye Wang, Yueping Zhang, Vishal Singh, Cristian Lumezanu, and Guofei Jiang. Netfuse: Short-circuiting traffic surges in the cloud. In *IEEE ICC*, 2013.

[209] Alexander Shalimov, Dmitry Zuikov, Daria Zimarina, Vasily Pashkov, and Ruslan Smeliansky. Advanced study of SDN/openflow controllers. In *CEE-SECR*, pages 1:1–1:6. ACM, 2013.

[210] S. Sezer, S. Scott-Hayward, P.K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao. Are we ready for SDN? implementation challenges for software-defined networks. *IEEE Communications Magazine*, 51(7):36 – 43, July 2013.

[211] Esther Le Rouzic, Edoardo Bonetto, Luca Chiaraviglio, Frédéric Giroire, Filip Idzikowski, Felipe Jimenez, Christoph Lange, Julio Montalvo, Francesco Musumeci, Issam Tahiri, Alessandro Valenti, Ward Van Heddeghem, Yabin Ye, Andrea Bianco, and Achille Pattavina. Trend towards more energy-efficient optical networks. In *Conference on Optical Network Design and Modelling (ONDM)*, pages 211–216, Brest, France, 2013.

[212] Remigiusz Modrzejewski, Luca Chiaraviglio, Issam Tahiri, Frederic Giroire, Esther Le Rouzic, Edoardo Bonetto, Francesco Musumeci, Roberto Gonzalez, and Carmen Guerrero. Energy Efficient Content Distribution in an ISP Network. In *IEEE Global Communications Conference (GLOBECOM)*, Atlanta, United States, December 2013.

[213] Mirka Miller and Jozef Sirán. Moore graphs and beyond: A survey of the degree/diameter problem. *The Electronic Journal of Combinatorics*, 2013. Dynamic Survey DS14.

[214] Uttam Mandal, Pulak Chowdhury, Christoph Lange, Andreas Gladisch, and Biswanath Mukherjee. Energy-efficient networking for content distribution over telecom network infrastructure. *Optical Switching and Networking*, 10(4):393 – 405, 2013.

[215] K. Kannan and S. Banerjee. Compact TCAM: Flow Entry Compaction in TCAM for Power Aware SDN. In *ICDCN*, pages 439–444, 2013.

[216] Y. Kanizo, D. Hay, and I. Keslassy. Palette: Distributing tables in software-defined networks. In *INFOCOM, 2013 Proceedings IEEE*, pages 545–549, April 2013.

[217] Nanxi Kang, Zhenming Liu, Jennifer Rexford, and David Walker. Optimizing the "one big switch" abstraction in software-defined networks. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '13, pages 13–24, New York, NY, USA, 2013. ACM.

[218] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, et al. B4: Experience with a globally-deployed software defined wan. *ACM SIGCOMM Computer Communication Review*, 43(4):3–14, 2013.

[219] Frédéric Giroire, Stéphane Pérennes, and Issam Tahiri. Sur la complexité du routage OSPF. In Nicolas Nisse, Franck Rousseau, and Yann Busnel, editors, *15èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel)*, pages 1–4, Pornic, France, 2013.

[220] Frédéric Giroire, Stéphane Pérennes, and Issam Tahiri. On the Hardness of Equal Shortest Path Routing. In *International Network Optimization Conference (INOC)*, volume 41, pages 439–446, 2013.

[221] Frédéric Giroire, Remigiusz Modrzejewski, Nicolas Nisse, and Stéphane Pérennes. Maintaining Balanced Trees For Structured Distributed Streaming Systems. In *20th Colloquium on Structural Information and Communication Complexity (SIROCCO)*, pages 177–188, Ischia, Italy, 2013. Springer.

[222] Frédéric Giroire, Sandeep Kumar Gupta, Remigiusz Modrzejewski, Julian Monteiro, and Stéphane Pérennes. Repair time in distributed storage systems. In *Data Management in Cloud, Grid and P2P Systems (Globe)*, pages 99–110, Prague, Czech Republic, 2013. Springer.

[223] David Erickson. The beacon openflow controller. In *HotSDN*, pages 13–18. ACM, 2013.

[224] Christian Dannewitz, Dirk Kutscher, BöRje Ohlman, Stephen Farrell, Bengt Ahlgren, and Holger Karl. Network of information (netinf) - an information-centric networking architecture. *Comput. Commun.*, 36(7):721–735, April 2013.

[225] Luca Chiaraviglio, Antonio Cianfrani, Esther Le Rouzic, and Marco Polverini. Sleep modes effectiveness in backbone networks with limited configurations. *Computer Networks*, 57(15):2931 – 2948, 2013.

[226] Jaideep Chandrashekar, Eve M Schooler, Nina Taft, and Frederic Giroire. Method and system for detecting and reducing botnet activity, dec 2013. US Patent 8,612,579.

[227] Julio Araujo, Frédéric Giroire, Yaning Liu, Modrzejewski Remigiusz, and Joanna Moulierac. Energy Efficient Content Distribution. In *IEEE International Conference on Communications (ICC)*, pages 4233–4238, Budapest, Hungary, June 2013. IEEE.

[228] J. Araujo, V. Campos, F. Giroire, N. Nisse, L. Sampaio, and R. Soares. On the hull number of some graph classes. *Theoretical Computer Science*, 475:1–12, 2013.

[229] Sugam Agarwal, Murali Kodialam, and TV Lakshman. Traffic engineering in software defined networks. In *IEEE INFOCOM*, 2013.

[230] Sherali Zeadally, Samee Ullah Khan, and Naveen Chilamkurti. Energy-efficient networking: past, present, and future. *The Journal of Supercomputing*, 62(3):1093–1118, 2012.

[231] Jun-Ming Xu and Min Xu. The forwarding indices of graphs – a survey. *CoRR*, abs/1204.2604, 2012.

[232] Mengjun Xie, I. Widjaja, and Haining Wang. Enhancing cache robustness for content-centric networking. In *INFOCOM, 2012 Proceedings IEEE*, pages 2426–2434, Orlando, FL, March 2012. IEEE.

[233] Haiyong Xie, Guangyu Shi, and Pengwei Wang. Tecc: Towards collaborative in-network caching guided by traffic engineering. In *INFOCOM, 2012 Proceedings IEEE*, pages 2546–2550, Orlando,FL, March 2012. IEEE.

[234] Ward Van Heddeghem, Filip Idzikowski, Willem Vereecken, Didier Colle, Mario Pickavet, and Piet Demeester. Power consumption modeling in optical multilayer networks. *Photonic Network Communications*, 24(2):86–102, 2012.

[235] Ward Van Heddeghem, Filip Idzikowski, Esther Le Rouzic, Jean Yves Mazeas, Hubert Poignant, Suzanne Salaun, Bart Lannoo, and Didier Colle. Evaluation of power rating of core network equipment in practical deployments. In *Online Conference on Green Communications (GreenCom), 2012 IEEE*, pages 126–132. IEEE, 2012.

[236] Amin Tootoonchian, Sergey Gorbunov, Yashar Ganjali, Martin Casado, and Rob Sherwood. On controller performance in software-defined networks. *Hot-ICE*, 12:1–6, 2012.

[237] Justine Sherry, Shaddi Hasan, Colin Scott, Arvind Krishnamurthy, Sylvia Ratnasamy, and Vyas Sekar. Making middleboxes someone else's problem: network processing as a cloud service. *ACM SIGCOMM Computer Communication Review*, 42(4):13–24, 2012.

[238] Ioannis Psaras, Wei Koong Chai, and George Pavlou. Probabilistic in-network caching for information-centric networks. In *Proceedings of the Second Edition of the ICN Workshop on Information-centric Networking*, ICN '12, pages 55–60, New York, NY, USA, August 2012. ACM.

[239] Luca Niccolini, Gianluca Iannaccone, Sylvia Ratnasamy, Jaideep Chandrashekar, and Luigi Rizzo. Building a power-proportional software router. In *Presented as part of the 2012 USENIX Annual Technical Conference (USENIX ATC 12)*, pages 89–100, 2012.

[240] C. R. Meiners, A. X. Liu, and E. Torng. Bit Weaving: A Non-prefix Approach to Compressing Packet Classifiers in TCAMs. In *IEEE/ACM Transaction in Networking*, pages 488 – 500, 2012.

[241] V. Mathew, R.K. Sitaraman, and P. Shenoy. Energy-aware load balancing in content delivery networks. In *INFOCOM, 2012 Proceedings IEEE*, pages 954–962, Orlando, FL, March 2012. IEEE.

[242] J. Llorca, K. Guan, G. Atkinson, and D.C. Kilper. Energy efficient delivery of immersive video centric services. In *IEEE INFOCOM*, pages 1656–1664, Orlando, USA, 2012.

[243] Zhe Li, Gwendal Simon, and Annie Gravey. Caching policies for in-network caching. In *21st International Conference on Computer Communications and Networks (ICCCN)*, pages 1 –7, Munich, Germany, August 2012. IEEE.

[244] Min Xu Jun-Ming Xu. The forwarding indices of graphs – a survey. xarchiv, 2012.

[245] Brandon Heller, Rob Sherwood, and Nick McKeown. The controller placement problem. In *Proceedings of the first workshop on Hot topics in software defined networks*, pages 7–12. ACM, 2012.

[246] F. Giroire, J. Moulierac, T.K. Phan, and F. Roudaut. Minimization of network power consumption with redundancy elimination. In *IFIP Networking*, pages 247–258, Prague, Czech Republic, May 2012. Springer.

[247] Chang Ge, Ning Wang, and Zhili Sun. Optimizing server power consumption in cross-domain content distribution infrastructures. In *Communications (ICC), 2012 IEEE International Conference on*, pages 2628–2633, Ottawa, ON, June 2012. IEEE.

[248] F. V. Fomin, F. Giroire, A. Jean-Marie, D. Mazauric, and N. Nisse. To satisfy impatient web surfers is hard. In *6th International Conference on FUN with Algorithms (FUN)*, volume 7288, pages 166–176. Springer, LNCS, 2012.

[249] F. V. Fomin, F. Giroire, A. Jean-Marie, D. Mazauric, and N. Nisse. Satisfaire un internaute impatient est difficile. In *14es Rencontres Francophones sur les Aspects Algorithmiques de Télécommunications (AlgoTel)*, pages 79–82, 2012.

[250] Nakjung Choi, Kyle Guan, Daniel C Kilper, and Gary Atkinson. In-network caching effect on optimal energy consumption in content-centric networking. In *2012 IEEE International Conference on Communications*, 2012.

[251] Nakjung Choi, K. Guan, D.C. Kilper, and G. Atkinson. In-network caching effect on optimal energy consumption in content-centric networking. In *Communications (ICC), 2012 IEEE International Conference on*, pages 2889–2894, Ottawa, ON, June 2012. IEEE.

[252] Luca Chiaraviglio, Marco Mellia, and Fabio Neri. Minimizing isp network energy cost: Formulation and solutions. *IEEE/ACM Trans. Netw.*, 20(2):463–476, April 2012.

[253] L. Chiaraviglio, M. Mellia, and F. Neri. Minimizing isp network energy cost: formulation and solutions. *IEEE/ACM Transactions on Networking (TON)*, 20:463–476, April 2012.

[254] Aruna Prem Bianzino, Luca Chiaraviglio, Marco Mellia, and Jean-Louis Rougier. Grida: {GReen} distributed algorithm for energy-efficient {IP} backbone networks. *Computer Networks*, 56(14):3219 – 3232, 2012.

[255] Aruna Prem Bianzino, Claude Chaudet, Dario Rossi, Jean-Louis Rougier, et al. A survey of green networking research. *IEEE Communications Surveys & Tutorials*, 14(1):3–20, 2012.

[256] J. Araujo, N. Cohen, F. Giroire, and F. Havet. Good edge-labelling of graphs. *Discrete Applied Mathematics*, 160(18):2502 – 2513, 2012.

[257] J. Araujo, J-C. Bermond, F. Giroire, F. Havet, D. Mazauric, and R. Modrzejewski. Weighted improper colouring. *Journal of Discrete Algorithms*, 16:53–66, October 2012.

[258] Akamai, 2012.

[259] Yunlong Song, Min Liu, and Yuwei Wang. Power-aware traffic engineering with named data networking. In *Mobile Ad-hoc and Sensor Networks (MSN), 2011 Seventh International Conference on*, pages 289–296, Beijing, China, Dec 2011. IEEE.

[260] Ahmed Wasif Reza, Tan Kim Geok, and Kaharudin Dimyati. Tracking via square grid of rfid reader positioning and diffusion algorithm. *Wireless Personal Communications*, 61(1):227–250, 2011.

[261] Dorian Mazauric. *Optimisation discrète dans les réseaux de télécommunication : reconfiguration du routage, routage efficace en énergie, ordonnancement de liens et placement de données.* PhD thesis, 2011. Thèse de doctorat dirigée par Bermond, Jean-Claude et Nain, Philippe Informatique Nice 2011.

[262] U. Mandal, C. Lange, A. Gladisch, P. Chowdhury, and B. Mukherjee. Energy-efficient content distribution over telecom network infrastructure. In *13th International Conference on Transparent Optical Networks (ICTON)*, pages 1–4, Stockholm, Sweden, 2011.

[263] Richard TB Ma, Dah Ming Chiu, John CS Lui, Vishal Misra, and Dan Rubenstein. On cooperative settlement between content, transit, and eyeball internet service providers. *Networking, IEEE/ACM Transactions on*, 19(3):802–815, 2011.

[264] U. Lee, I. Rimac, D. Kilper, and V. Hilt. Toward energy-efficient content dissemination. *Network, IEEE*, 25(2):14–19, 2011.

[265] C. Lange, D. Kosiankowski, R. Weidmann, and A. Gladisch. Energy consumption of telecommunication networks and related improvement options. *Selected Topics in Quantum Electronics, IEEE Journal of*, 17(2):285–295, 2011.

[266] C. Jayasundara, A. Nirmalathas, E. Wong, and C.A. Chan. Energy efficient content distribution for VoD services. In *Optical Fiber Communication Conference*, pages 1–3, Los Angeles, USA, 2011.

[267] K. Guan, G. Atkinson, D.C. Kilper, and E. Gulsen. On the energy efficiency of content delivery architectures. In *Communications Workshops (ICC), 2011 IEEE International Conference on*, pages 1–6, Kyoto, Japan, June 2011. IEEE.

[268] F. Giroire, D. Mazauric, and J. Moulierac. Routage efficace en énergie. In Ducourthial et Bertrand et Felber et Pascal, editor, *13es Rencontres Francophones sur les Aspects Algorithmiques de Télécommunications (AlgoTel)*, Cap Estérel, France, 2011.

[269] D. Coudert, F. Giroire, and I. Sau. Circuits in graphs through a prescribed set of ordered vertices. *Journal of Interconnection Networks*, 11(3-4):121–141, 2011.

[270] Mosharaf Chowdhury, Matei Zaharia, Justin Ma, Michael I Jordan, and Ion Stoica. Managing data transfers in computer clusters with orchestra. In *ACM SIGCOMM Computer Communication Review*, volume 41, pages 98–109. ACM, 2011.

[271] L. Chiaraviglio and I. Matta. An energy-aware distributed approach for content and network management. In *2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 337–342, Shanghai, China, April 2011. IEEE.

[272] Raffaele Bolla, Franco Davoli, Roberto Bruschi, Ken Christensen, Flavio Cucchietti, and Suresh Singh. The potential impact of green technologies in next-generation wireline networks: Is there room for energy saving optimization? *IEEE Communications Magazine*, 49(8), 2011.

[273] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti. Energy efficiency in the future internet: a survey of existing approaches and trends in energy-aware fixed network infrastructures. *Communications Surveys & Tutorials, IEEE*, 13(2):223–244, 2011.

[274] J. Araujo, F. Giroire, and J. Monteiro. Hybrid approaches for distributed storage systems. In *Proceedings of Fourth International Conference on Data Management in Grid and P2P Systems (Globe'11)*, Toulouse, France, September 2011.

[275] J. Araujo, V. Campos, F. Giroire, L. Sampaio, and R. Soares. On the hull number of some graph classes. In *Proceedings of European Conference on Combinatorics, Graph Theory and Applications (EuroComb)*, volume 38 of *Electronic Notes in Discrete Mathematics*, pages 49–55, Budapest, Hungary, September 2011.

[276] J Araujo, J-C Bermond, F. Giroire, F. Havet, D. Mazauric, and R. Modrzejewski. Weighted improper colouring. In *Proceedings of International Workshop on Combinatorial Algorithms (IWOCA'11)*, volume 7056 of *Lecture Notes in Computer Science*, pages 1–18, Victoria, Canada, June 2011. Springer-Verlag.

[277] Jeffrey G Andrews, François Baccelli, and Radha Krishna Ganti. A tractable approach to coverage and rate in cellular networks. *IEEE Transactions on Communications*, 59(11):3122–3134, 2011.

[278] Mingui Zhang, Cheng Yi, Bin Liu, and Beichuan Zhang. Greente: Power-aware traffic engineering. In *Proceedings of the 18th IEEE International Conference on Network Protocols*, ICNP '10, pages 21–30, Washington, DC, USA, Oct 2010. IEEE Computer Society.

[279] Y. Song, K. Guo, and L. Gao. "Redundancy-Aware Routing with Limited Resources". In *ICCCN*, 2010.

[280] S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly. SNDlib 1.0–Survivable Network Design Library. *Networks*, 55(3):276–286, 2010.

[281] JUNIPER NETWORKS. Energy efficiency for network equipment: Two steps beyond greenwashing. Technical report, JUNIPER NETWORKS, 2010.

[282] C. R. Meiners, A. X. Liu, and E. Torng. TCAM Razor: A Systematic Approach Towards Minimizing Packet Classifiers in TCAMs. In *IEEE/ACM Transaction in Networking*, pages 490–500, 2010.

[283] U. Lee, I. Rimac, and V. Hilt. Greening the internet with content-centric networking. In *Proceedings of the 1st ACM International Conference on Energy-Efficient Computing and Networking*, pages 179–182, Passau, Germany, 2010.

[284] F. Idzikowski, S. Orlowski, C. Raack, H. Woesner, and A. Wolisz. Saving energy in IP-over-WDM networks by switching off line cards in low-demand scenarios. In *ONDM*, 2010.

[285] F. Giroire, J. Monteiro, and S. Pérennes. Peer-to-peer storage systems: a practical guideline to be lazy. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Miami, EUA, 12 2010.

[286] F. Giroire, D. Mazauric, J. Moulierac, and B. Onfroy. Minimizing routing energy consumption: from theoretical to practical results. In *IEEE/ACM International Conference on Green Computing and Communications (GreenCom'10)*, page 8, Hangzhou, China, 2010.

[287] F. Giroire, D. Mazauric, J. Moulierac, and B. Onfroy. Minimizing routing energy consumption: from theoretical to practical results. In *IEEE/ACM International Conference on Green Computing and Communications (GreenCom)*, page 8, Hangzhou, China, 2010.

[288] F. Giroire, D. Mazauric, J. Moulierac, and B. Onfroy. Minimizing Routing Energy Consumption: from theoretical to practical results. Research Report inria-00464318, May 2010.

[289] Anja Feldmann, Andreas Gladisch, Mario Kind, C Lange, G Smaragdakis, and F-J Westphal. Energy trade-offs among content delivery architectures. In *IEEE CTTE '10*, pages 1–6, 2010.

[290] D. Coudert, N. Nepomuceno, and H. Rivano. Power-efficient radio configuration in fixed broadband wireless networks. *Computer Communications*, 2010. To appear.

[291] Ken Christensen, Pedro Reviriego, Bruce Nordman, Michael Bennett, Mehrgan Mostowfi, and Juan Antonio Maestro. Ieee 802.3 az: the road to energy efficient ethernet. *IEEE Communications Magazine*, 48(11), 2010.

[292] Luca Chiaraviglio and Ibrahim Matta. Greencoop: Cooperative green routing with energy-efficient servers. In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, e-Energy '10, pages 191–194, New York, NY, USA, April 2010. ACM.

[293] L. Chiaraviglio and I. Matta. Greencoop: cooperative green routing with energy-efficient servers. In *Proceedings of the 1st ACM International Conference on Energy-Efficient Computing and Networking*, pages 191–194, Passau, Germany, 2010.

[294] S. Caron, F. Giroire, D. Mazauric, J. Monteiro, and S. Pérennes. P2P Storage Systems: Data Life Time for Different Placement Policies. Research Report RR-7209, INRIA, February 2010.

[295] S. Caron, F. Giroire, D. Mazauric, J. Monteiro, and S. Pérennes. P2P Storage Systems: Data Life Time for Different Placement Policies. In Maria Gradinariu Potop-Butucaru et Hervé Rivano, editor, *12èmes Rencontres Francophones sur les Aspects Algorithmiques de Télécommunications (AlgoTel)*, page 4p, Belle Dune France, 2010.

[296] S. Caron, F. Giroire, D. Mazauric, J. Monteiro, and S. Pérennes. Data life time for different placement policies in p2p storage systems. In *Proceedings of the 3rd Intl. Conference on Data Management in Grid and P2P Systems (Globe)*, volume 6265 of *Lecture Notes in Computer Science*, pages 75–88, Bilbao, Spain, September 2010.

[297] Sem Borst, Varun Gupta, and Anwar Walid. Distributed caching algorithms for content distribution networks. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9. IEEE, 2010.

[298] O. Amini, F. Giroire, F. Huc, and S. Pérennes. Minimal selectors and fault tolerant networks. *Networks*, 55(4):326–340, July 2010.

[299] Anke Van Zuylen. Linear programming based approximation algorithms for feedback set problems in bipartite tournaments. In *Theory and Applications of Models of Computation*, pages 370–379. Springer, 2009.

[300] Vytautas Valancius, Nikolaos Laoutaris, Laurent Massoulié, Christophe Diot, and Pablo Rodriguez. Greening the internet with nano data centers. In *ACM CoNEXT '09*, pages 37–48, Rome, Italy, 2009.

[301] L.B. Sofman and B. Krogfoss. Analytical model for hierarchical cache optimization in iptv network. *Broadcasting, IEEE Transactions on*, 55(1):62–70, 2009.

[302] E.J. Rosensweig and J. Kurose. Breadcrumbs: Efficient, best-effort content location in cache networks. In *INFOCOM 2009, IEEE*, pages 2631–2635, Rio de Janeiro, Brazil, April 2009. IEEE.

[303] JC Cardona Restrepo, Claus G Gruber, and C Mas Machuca. Energy profile aware routing. In *Communications Workshops, 2009. ICC Workshops 2009. IEEE International Conference on*, pages 1–5. IEEE, 2009.

[304] B. Puype, W. Vereecken, D. Colle, M. Pickavet, and P. Demeester. Power Reduction Techniques in Multilayer Traffic Engineering. In *ICTON*, 2009.

[305] M Ajmone Marsan, Luca Chiaraviglio, Delia Ciullo, and Michela Meo. Optimal energy savings in cellular access networks. In *Communications Workshops, 2009. ICC Workshops 2009. IEEE International Conference on*, pages 1–5. IEEE, 2009.

[306] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan. A Power Benchmarking Framework for Network Devices. In *IFIP NETWORKING 2009*, pages 795–808, may 2009.

[307] Adrian Kosowski. Forwarding and optical indices of a graph. *Discrete Applied Mathematics*, 157(2):321–329, 2009.

[308] Wenjie Jiang, Rui Zhang-Shen, Jennifer Rexford, and Mung Chiang. Cooperative content distribution and traffic engineering in an isp network. In *Proceedings of the Eleventh International Joint Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '09, pages 239–250, New York, NY, USA, June 2009. ACM.

[309] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. Networking named content. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '09, pages 1–12, New York, NY, USA, December 2009. ACM.

[310] Martin Haenggi, Jeffrey G Andrews, François Baccelli, Olivier Dousse, and Massimo Franceschetti. Stochastic geometry and random graphs for the analysis and design of wireless networks. *IEEE Journal on Selected Areas in Communications*, 27(7):1029–1046, 2009.

[311] A. Greenberg, J. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. Maltz, P. Patel, and S. Sengupta. VL2: a scalable and flexible data center network. In *SIGCOMM Comp. Commun. Rev.*, volume 39:4, pages 51–62. ACM, 2009.

[312] F. Giroire, J. Monteiro, and S. Pérennes. P2P storage systems: How much locality can they tolerate? Research Report RR-7006, INRIA, July 2009.

[313] F. Giroire, J. Monteiro, and S. Pérennes. P2P storage systems: How much locality can they tolerate? In *Proceedings of the 34th IEEE Conference on Local Computer Networks (LCN)*, pages 320–323, Oct 2009.

[314] F. Giroire, J. Chandrashekar, N. Taft, E. Schooler, and K. Papagiannaki. Exploiting temporal persistence to detect covert botnet channels. In Springer, editor, *12th International Symposium on Recent Advances in Intrusion Detection (RAID)*, volume 5758 of *Lecture Notes in Computer Science*, pages 326–345, Saint Malo, France, September 2009.

[315] F. Giroire. Order statistics and estimating cardinalities of massive data sets. *Discrete Applied Mathematics*, 157(2):406–427, 2009.

[316] Mitre C Dourado, John G Gimbel, Jan Kratochvíl, Fábio Protti, and Jayme L Szwarcfiter. On the computation of the hull number of a graph. *Discrete Mathematics*, 309(18):5668–5674, 2009.

[317] O. Dalle, F. Giroire, J. Monteiro, and S. Pérennes. Analysis of failure correlation impact on peer-to-peer storage systems. In *Proceedings of the 9th IEEE International Conference on Peer-to-Peer Computing (P2P)*, pages 184–193, sep 2009.

[318] D. Coudert, F. Giroire, and I. Sau. Edge-simple circuits through 10 ordered vertices in square grids. In J. Kratochvíl and M. Miller, editors, *20th International Workshop on Combinatorial Algorithms – IWOCA*, volume 5874 of *Lecture Notes in Computer Science*, pages 134–145, Hradec nad Moravicí, Czech Republic, June 2009. Springer.

[319] D. Coudert, F. Giroire, and I. Sau. Circuit visiting 10 ordered vertices in infinite grids. Technical Report RR-6910, INRIA, 2009.

[320] NM Mosharaf Kabir Chowdhury, Muntasir Raihan Rahman, and Raouf Boutaba. Virtual network embedding with coordinated node and link mapping. In *INFOCOM 2009, IEEE*, pages 783–791. IEEE, 2009.

[321] Luca Chiaraviglio, Marco Mellia, and Fabio Neri. Reducing power consumption in backbone networks. In *IEEE International Conference on Communications*, pages 1–6, 2009.

[322] L. Chiaraviglio, M. Mellia, and F. Neri. Reducing power consumption in backbone networks. In *ICC*, 2009.

[323] Vikram Chandrasekhar, Jeffrey G Andrews, Tarik Muharemovic, Zukang Shen, and Alan Gatherer. Power control in two-tier femtocell networks. *IEEE Transactions on Wireless Communications*, 8(8):4316–4328, 2009.

[324] Jean-Claude Bermond, Michel Cosnard, and Stéphane Pérennes. Directed acyclic graphs with the unique dipath property. Research Report RR-6932, 2009.

[325] D. Barman, J. Chandrashekar, N. Taft, M. Faloutsos, L. Huang, and F. Giroire. Impact of it monoculture on behavioral end host intrusion detection. In *ACM SIGCOMM Workshop on Research on Enterprise Networking – WREN*, pages 27–36, Barcelona, Spain, August 2009. ACM.

[326] J. Baliga, R. Ayre, K. Hinton, and R.S. Tucker. Architectures for energy-efficient iptv networks. In *Optical Fiber Communication Conference*, San Diego, California, 2009.

[327] J. Araujo, N. Cohen, F. Giroire, and F. Havet. Good edge-labelling of graphs. Research Report 6934, INRIA, 2009.

[328] J. Araujo, N. Cohen, F. Giroire, and F. Havet. Good edge-labelling of graphs. In *proceedings of the Latin-American Algorithms, Graphs and Optimization Symposium (LAGOS)*, volume 35 of *Electronic Notes in Discrete Mathematics*, pages 275–280, Gramado, Brazil, December 2009. Springer.

[329] A. Anand, V. Sekar, and A. Akella. "SmartRE: an Architecture for Coordinated Network-wide Redundancy Elimination". In *SIGCOMM*, 2009.

[330] A. Anand, C. Muthukrishnan, A. Akella, and R. Ramjee. "Redundancy in Network Traffic: Findings and Implications". In *SIGMETRICS*, 2009.

[331] Molly Webb et al. Smart 2020: Enabling the low carbon economy in the information age. *The Climate Group. London*, 1(1):1–1, 2008.

[332] R, J. Baliga, R. Ayre, K. Hinton, and W. Energy consumption in IP networks. In *Optical Communication, 2008. ECOC 2008. 34th European Conference on*, 2008.

[333] S. Paul, R. Yates, D. Raychaudhuri, and J. Kurose. The cache-and-forward network architecture for efficient mobile content delivery services in the future internet. In *Innovations in NGN: Future Network and Services, 2008. K-INGN 2008. First ITU-T Kaleidoscope Academic Conference*, pages 367–374, Geneva, Switzerland, May 2008. IEEE.

[334] White Paper. Energy efficiency for network equipment: Two steps beyond greenwashing. Technical report, Juniper Networks Inc., 2008.

[335] Sergiu Nedevschi, Lucian Popa, Gianluca Iannaccone, Sylvia Ratnasamy, and David Wetherall. Reducing network energy consumption via sleeping and rate-adaptation. In *NsDI*, volume 8, pages 323–336, 2008.

[336] L. Nedevschi, S.and Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall. Reducing Network Energy Consumption via Rate-Adaptation and Sleeping. In *NSDI*, 2008.

[337] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan. Energy Aware Network Operations. In *HP Labs*, 2008.

[338] F. Giroire, A. Nucci, N. Taft, and C. Diot. Method and systems for identifying optimal mapping in a network, November 2008. Patent number US 7,453,824 B1.

[339] F. Giroire, A. Nucci, N. Taft, and C. Diot. Method and systems for correlating practical constraints in a network, July 2008. Sprint, Patent number US 7,394,760 B1.

[340] F. Giroire, J. Chandrashekar, G. Iannaccone, D. Papagiannaki, E. Schooler, and N. Taft. The cubicle vs. the coffee shop: Behavioral modes in enterprise end-users. In *Proceeding of the Passive and Active Monitoring conference (PAM)*, volume 4979 of *Lecture Notes in Computer Science*, pages 202–211. Springer, 2008.

[341] P. Gill, M. Arlitt, Z. Li, and A. Mahanti. The flattening internet topology: Natural evolution, unsightly barnacles or contrived collapse? In *Passive and Active Network Measurement*, pages 1–10, Cleveland, USA, 2008. Springer.

[342] O. Dalle, F. Giroire, J. Monteiro, and S. Pérennes. Analysis of failure correlation in peer-to-peer storage systems. Technical Report RR-6761, INRIA, December 2008.

[343] Joseph Chabarek, Joel Sommers, Paul Barford, Cristian Estan, David Tsiang, and Steve Wright. Power awareness in network design and routing. In *IEEE 27th Conference on Computer Communications (INFOCOM)*, pages 457–465, april 2008.

[344] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright. Power awareness in network design and routing. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, April 2008.

[345] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright. Power awareness in network design and routing. In *IEEE INFOCOM*, pages 457–465, 2008.

[346] M. Cha, P. Rodriguez, J. Crowcroft, S. Moon, and X. Amatriain. Watching television over an IP network. In *ACM IMC '08*, pages 71–84, Vouliagmeni, Greece, 2008.

[347] Ivan Baev, Rajmohan Rajaraman, and Chaitanya Swamy. Approximation algorithms for data placement problems. *SIAM Journal on Computing*, 38(4):1411–1429, 2008.

[348] White paper, cisco, cisco integrated services router: Reduce power consumption through integrated services delivery, 2008.

[349] Jiong Guo, Falk Hüffner, and Hannes Moser. Feedback arc set in bipartite tournaments is np-complete. *Information processing letters*, 102(2):62–65, 2007.

[350] T. Jr. Grevers and J. Christner. "Application Acceleration and WAN Optimization Fundamentals". In *Cisco Press*, 2007.

[351] F. Giroire, J. Chandrashekar, N. Taft, G. Iannaccone, T. Karagiannis, K. Papagiannaki, and E. Schooler. The case for personalizing end-host detectors. Technical report, Intel Research, 2007.

[352] F. Giroire, J. Chandrashekar, G. Iannaccone, T. Karagiannis, K. Papagiannaki, E. Schooler, and N. Taft. Inside the forbidden city: A look at end-host traffic inside a modern enterprise. Technical report, Intel Research, 2007.

[353] Éric Fusy and Frécéric Giroire. Estimating the number of active flows in a data stream over a sliding window. In *2007 Proceedings of the Fourth Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*, pages 223–231. SIAM, 2007.

[354] Massimo Franceschetti, Olivier Dousse, NC David, and Patrick Thiran. Closing the gap in the capacity of wireless networks via percolation theory. *IEEE Transactions on Information Theory*, 53(3):1009–1018, 2007.

[355] J-C. Bermond, F. Giroire, and S. Pérennes. Design of minimal fault tolerant on-board networks : Practical constructions. In *14th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 4474 of *Lecture Notes in Computer Sciences*, pages 261–273, Castiglioncello, Italy, June 2007.

[356] D. L. Applegate, G. Calinescu, D. S. Johnson, H. Karloff, K. Ligett, and J. Wang. Compressing Rectilinear Pictures and Minimizing Access Control Lists. In *ACM-SIAM SODA*, pages 1066–1075, 2007.

[357] F. Giroire. Directions to use probabilistic algorithms for cardinality for DNA analysis. In *Journées Ouvertes Biologie Informatique Mathématiques (JOBIM 2006)*, pages 3–5, July 2006.

[358] Joseph Camp, Joshua Robinson, Christopher Steger, and Edward Knightly. Measurement driven deployment of a two-tier urban mesh access network. In *Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 96–109. ACM, 2006.

[359] O. Amini, F. Giroire, F. Huc, and S. Pérennes. Minimal selectors and fault tolerant networks. Research report, INRIA Research Report HAL-00082015, July 2006.

[360] O. Amini, J-C. Bermond, F. Giroire, F. Huc, and S. Pérennes. Design of minimal fault tolerant networks: Asymptotic bounds. In *Huitièmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel'06)*, pages 37–40, Trégastel, France, May 2006.

[361] David E Taylor. Survey and taxonomy of packet classification techniques. *ACM Computing Surveys (CSUR)*, 37(3):238–275, 2005.

[362] Anukool Lakhina, Mark Crovella, and Christophe Diot. Mining anomalies using traffic feature distributions. In *ACM Computer Communication Review*, volume 35, 2005.

[363] F. Giroire. Order statistics and estimating cardinalities of massive data sets. In Conrado Martinez, editor, *2005 International Conference on Analysis of Algorithms*, volume AD of *DMTCS Proceedings*, pages 157–166. Discrete Mathematics and Theoretical Computer Science, 2005.

[364] Claude Chaudet, Eric Fleury, Isabelle Guérin Lassous, Hervé Rivano, and Marie-Emilie Voge. Optimal positioning of active and passive monitoring devices. In *Proceedings of the 2005 ACM conference on Emerging network experiment and technology*, pages 71–82. ACM, 2005.

[365] Anukool Lakhina, Mark Crovella, and Christophe Diot. Characterization of network-wide anomalies in traffic flows. In *ACM IMC*, 2004.

[366] B. Claise. RFC 3954 - Cisco Systems NetFlow Services Export Version 9, 2004.

[367] Subhash Suri, Tuomas Sandholm, and Priyank Warkhede. Compressing two-dimensional routing tables. *Algorithmica*, 35(4):287–300, 2003.

[368] Yoav Sasson, David Cavin, and André Schiper. Probabilistic broadcast for flooding in wireless mobile ad hoc networks. In *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, volume 2, pages 1124–1130. IEEE, 2003.

[369] Sundar Iyer, Supratik Bhattacharyya, Nina Taft, and Christophe Diot. An approach to alleviate link overload as observed on an IP backbone. In *IEEE Annual Joint Conference of the IEEE Computer and Communications Societies - INFOCOM*, volume 1, pages 406–416, 2003.

[370] Maruti Gupta and Suresh Singh. Greening of the internet. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 19–26. ACM, 2003.

[371] M. Gupta and S. Singh. Greening of the internet. In *SIGCOMM*, 2003.

[372] F. Giroire, A. Nucci, N. Taft, and C. Diot. Increasing the robustness of ip backbones in the absence of optical level protection. In *Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2003.

[373] Chuck Fraleigh, Sue Moon, Bryan Lyles, Chase Cotton, Mujahid Khan, Deb Moll, Rob Rockell, Ted Seely, and SC Diot. Packet-level traffic measurements from the sprint ip backbone. *Network, IEEE*, 17(6):6–16, 2003.

[374] Ismail Ari, Bo Hong, Ethan L Miller, Scott A Brandt, and Darrell DE Long. Managing flash crowds on the internet. In *IEEE/ACM MASCOTS 2003*, 2003.

[375] Sangho Shim, Jozef Širáň, and Janez Žerovnik. Counterexamples to the uniform shortest path routing conjecture for vertex-transitive graphs. *Discrete applied mathematics*, 119(3):281–286, 2002.

[376] Matthew Roughan, Albert Greenberg, Charles Kalmanek, Michael Rumsewicz, Jennifer Yates, and Yin Zhang. Experience in measuring backbone traffic variability: Models, metrics, measurements and meaning. In *ACM IMW*, 2002.

[377] Seapahn Meguerdichian, Farinaz Koushanfar, Gang Qu, and Miodrag Potkonjak. Exposure in wireless ad-hoc sensor networks. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 139–150. ACM, 2001.

[378] Gianluca Iannaccone, Christophe Diot, Ian Graham, and Nick McKeown. Monitoring very high speed links. In *ACM IMW*, 2001.

[379] N. T. Spring and D. Wetherall. "A Protocol-Independent Technique for Eliminating Redundant Network Traffic". In *SIGCOMM*, 2000.

[380] Andrew Odlyzko. Data networks are mostly empty and for good reason. *IT professional*, 1(2):67–69, 1999.

[381] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *J. ACM*, 46(6):787–832, November 1999.

[382] V. Gabrel, A. Knippel, and M. Minoux. Exact solution of multicommodity network optimization problems with general step cost functions. *Operations Research Letters*, 25(1):15–23, 1999.

[383] Kapil Chawla, Xiaoxin Qiu, and Martin V Clark. Design of a wireless backhaul network for microcells. In *Wireless Communications and Networking Conference, 1999. WCNC. 1999 IEEE*, pages 428–432. IEEE, 1999.

[384] Ivan Stojmenovic. Honeycomb networks: Topological properties and communication algorithms. *IEEE Transactions on Parallel and Distributed Systems*, 8(10):1036–1042, 1997.

[385] Bojan Mohar. Some applications of laplace eigenvalues of graphs. In *Graph Symmetry*, volume 497 of *NATO ASI Series*, pages 225–275. Springer Netherlands, 1997.

[386] Michael Theobald, Steven M. Nowick, and Tao Wu. Espresso-hf: A heuristic hazard-free minimizer for two-level logic. In *Proceedings of the 33rd Annual Design Automation Conference*, DAC '96, pages 71–76, New York, NY, USA, 1996. ACM.

[387] Yannis Manoussakis and Zsolt Tuza. The forwarding index of directed networks. *Discrete Applied Mathematics*, 68(3):279 – 291, 1996.

[388] Patrick Solé. Expanding and forwarding. *Discrete Appl. Math.*, 58(1):67–78, 1995.

[389] Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:577–591, 1994.

[390] Hervé Brönnimann and Michael T Goodrich. Almost optimal set covers in finite vc-dimension. In *Proceedings of ACM SOCG*, pages 293–302, 1994.

[391] Abdelmadjid Bouabdallah and Dominique Sotteau. On the edge forwarding index problem for small graphs. *Networks*, 23(4):249–255, 1993.

[392] Leandros Tassiulas and Anthony Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *Automatic Control, IEEE Transactions on*, 37(12):1936–1948, Dec 1992.

[393] Alistair Sinclair. Improved bounds for mixing rates of markov chains and multicommodity flow. *Combinatorics, Probability and Computing*, 1:351–370, 1992.

[394] Viggo Kann. *On the approximability of NP-complete optimization problems.* PhD thesis, Royal Institute of Technology Stockholm, 1992.

[395] W. Fernandez de la Vega and L. Marquez Gordones. The forwarding indices of random graphs. *Random Structures and Algorithms*, 3(1):107–116, 1992.

[396] W. Fernandez de la Vega and Y. Manoussakis. The forwarding index of communication networks with given connectivity. *Discrete Appl. Math.*, 37-38:147–155, July 1992.

[397] Zicheng Guo, Rami G Melhem, Richard W Hall, Donald M Chiarulli, and Steven P Levitan. Pipelined communications in optically interconnected arrays. *Journal of Parallel and Distributed Computing*, 12(3):269–282, 1991.

[398] Farhad Shahrokhi and D. W. Matula. The maximum concurrent flow problem. *J. ACM*, 37(2):318–334, April 1990.

[399] Walter Schnyder. Embedding planar graphs on the grid. In *SoDA*, volume 90, pages 138–148, 1990.

[400] G.M. Guisewite and P.M. Pardalos. Minimum concave-cost network flow problems: Applications, complexity, and algorithms. *Annals of Operations Research*, 25(1):75–99, 1990.

[401] Marie Claude Heydemann, Jean Claude Meyer, and Dominique Sotteau. On forwarding indices of networks. *Discrete Applied Mathematics*, 23(2):103 – 123, 1989.

[402] Prabhakar Raghavan and ClarkD. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[403] Fan R. K. Chung, Edward G. Coffman Jr., Martin I. Reiman, and Burton Simon. The forwarding index of communication networks. *IEEE Transactions on Information Theory*, 33(2):224–232, 1987.

[404] Martin Farber and Robert E Jamison. Convexity in graphs and hypergraphs. *SIAM Journal on Algebraic Discrete Methods*, 7(3):433–444, 1986.

[405] Hristo Nicolov Djidjev. On the problem of partitioning planar graphs. *SIAM Journal on Algebraic Discrete Methods*, 3(2):229–240, 1982.

[406] Chuan-Lin Wu and Tse-Yun Feng. The universality of the shuffle-exchange network. *IEEE Transactions on Computers*, 100(5):324–332, 1981.

[407] Richard J Lipton and Robert Endre Tarjan. Applications of a planar separator theorem. In *Foundations of Computer Science, 1977., 18th Annual Symposium on*, pages 162–170. IEEE, 1977.

[408] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[409] S. Even, A. Itai, and A. Shamir. On the complexity of time table and multi-commodity flow problems. In *16th Annual Symposium on Foundations of Computer Science, 1975.*, pages 184–193, 1975.

[410] B. Yaged Jr. Minimum cost routing for static network models. *Networks*, 1(2):139–172, 1971.

[411] László Lovász. On decomposition of graphs. *Studia Sci. Math. Hungar*, 1(273):238, 1966.

[412] Kurt Spielberg. On the fixed charge transportation problem. In *Proceedings of the 1964 19th ACM national conference*, pages 11.101–11.1013, New York, NY, USA, 1964. ACM.

[413] Paul Erdős and Alfred Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungary. Acad. Sci.*, 5:17–61, 1960.

[414] Frédéric Giroire, Nicolas Huin, and Andrea Tomassilli. The structured way of dealing with heterogeneous live streaming systems.

[415] F. Giroire, S. Perennes, and I. Tahiri. How to design graphs with low forwarding index and limited number of edges.

[416] F. Giroire, F. Havet, and J. Moulierac. Compressing two-dimensional routing tables with order: Fixed parameter tractability.

[417] Twitch homepage, http://www.twitch.com/.

[418] Quagga. http://www.nongnu.org/quagga/.

[419] Openstack. http://www.openstack.org/.

[420] Openflow. https://www.opennetworking.org/.

[421] Opendaylight. https://www.opendaylight.org/.

[422] Open vswitch. http://openvswitch.org/.

[423] NEC univerge PF5240 and PF5820. http://www.openflow. org/wp/switch-nec/.

[424] Mininet. http://mininet.org/.

[425] International telecommunication union, world internet users statistics, https://www.internetworldstats.com/stats.htm.