Planification intelligente des flux entre centres de données

par

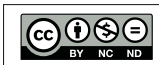
Meriem Amina Si Saber

MÉMOIRE PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
COMME EXIGENCE PARTIELLE À L'OBTENTION DE LA MAÎTRISE
AVEC MÉMOIRE EN GÉNIE CONCENTRATION RÉSEAUX DE
TÉLÉCOMMUNICATION
M. Sc. A.

MONTRÉAL, LE 13 DÉCEMBRE 2024

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC


Meriem Amina Si Saber, 2024



Cette licence Creative Commons signifie qu'il est permis de diffuser, d'imprimer ou de sauvegarder sur un autre support une partie ou la totalité de cette oeuvre à condition de mentionner l'auteur, que ces utilisations soient faites à des fins non commerciales et que le contenu de l'oeuvre n'ait pas été modifié.

PRÉSENTATION DU JURY

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE:

M. Mohamed Cheriet, directeur de mémoire
Département de génie de la production automatisée à l'École de technologie supérieure

Mme. Bassant Selim, présidente du jury
Département de génie des systèmes à l'École de technologie supérieure

M. Aris Leivadeas, membre du jury
Département de génie logiciel et TI à l'École de technologie supérieure

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 04 DÉCEMBRE 2024

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

REMERCIEMENTS

Je tiens à remercier toutes les personnes qui ont contribué de près ou de loin à ce travail.

Je voudrais, en premier lieu, adresser toute ma gratitude à M. Mohamed Cheriet, professeur titulaire à l'école de technologie supérieure et directeur du laboratoire Synchromedia, pour sa patience, sa disponibilité, mais aussi son précieux soutien qui m'a donné la force de mener à bien ce mémoire.

Je remercie aussi les membres du jury pour m'avoir fait l'honneur d'accepter d'évaluer mon travail.

Je souhaite remercier mes collègues et maintenant amis du laboratoire Synchromedia. Un immense merci à mon père (paix sur lui), ma mère et mes sœurs pour m'avoir aidé à persévérer.

Pour finir, je dédie mon travail à mon époux pour son soutien inconditionnel et aussi à mes deux petits monstres Amir et Selma qui, sans le savoir, ont été ma plus grande source d'inspiration.

Planification intelligente des flux entre centres de données

Meriem Amina Si Saber

RÉSUMÉ

La croissance importante des utilisateurs des centres de données et les échanges entre ces centres soulignent l'importance d'une planification efficace des flux.

Cependant, le problème dans les réseaux actuels interconnectant les centres de données, est que le scénario où les utilisateurs finaux précisent clairement les exigences pour leurs transferts, est irréaliste en raison de l'absence d'une interface permettant d'accomplir efficacement cette tâche. Par conséquent, et pour maintenir le niveau de performance à atteindre, la plupart des travaux existants supposent la disponibilité d'informations caractérisant les flux et se concentrent uniquement sur l'efficacité plutôt que sur la généralité, cet environnement est appelé agnostique à l'information.

L'idée principale de ce mémoire est de surmonter ce défi et de proposer une alternative efficace qui représente correctement les caractéristiques du trafic sans nécessiter une interface utilisateur évolutive et surtout onéreuse. Nous proposons ainsi une approche de planification basée sur une classification en ligne.

Tout d'abord, nous avons construit un nouveau module de classification basé sur la corrélation et combinant une approche sensible aux coûts avec un algorithme d'ensemble de forêts aléatoires, pour traiter le problème de déséquilibre des classes tout en répondant aux exigences de délais des différentes classes de trafic. Afin de calculer les corrélations entre flux qui représentent les poids de rééquilibrage, nous proposons un algorithme des k voisins les plus proches inversé.

Les résultats révèlent que l'algorithme proposé surpasse la plupart des approches dans les différents ensembles de données en termes de précision, rappel, mesure F1, AUC et Kappa. Les autres algorithmes ont donné une haute précision avec un faible rappel ou une faible précision et un haut rappel provoquant une congestion ou un sur-aprovisionnement des ressources.

Le résultat du module de classification représente des paramètres clé pour caractériser le trafic entrant dans le module de planification décrit par un problème d'optimisation, qui, en plus de garantir de meilleures performances de qualité mesurées par des délais d'acheminement optimaux, vise également à la minimisation des coûts en proposant une stratégie de provisionnent rentable.

Alors que d'autres approches obtiennent des taux de pertes élevés, notre approche préserve la qualité et la quantité du trafic échangé. De plus, notre approche surpasse les approches existantes, en particulier dans l'aspect généralisation, puisque c'est une méthode de planification en ligne, mais surtout multi-classe. La partie la plus longue de notre méthode est le réglage (qui est tout à fait négligeable).

Mots-clés: planification, optimisation, agnostique à l'information, classification, données non-balancées, algorithme de type ensemble

Information-Agnostic inter-data center traffic scheduling

Meriem Amina Si Saber

ABSTRACT

The important growth of data center users and inter-data center exchanges highlights the importance of efficient flow scheduling.

However, the problem in the current inter-DC WANs is that the scenario where end users clearly specify the requirements for their transfers is unrealistic because of the absence of an interface that accomplishes this task. Consequently, and to maintain the level of performance that must be met, most existing works assume the availability of flow information and focus only on efficiency rather than generality, this environment is referred to as information-agnostic.

The main idea behind this thesis is to overcome this challenge and propose an efficient alternative that understands traffic characteristics without requiring an expensive and non-scalable user interface. We thus propose an information-agnostic classification-based scheduling approach that differentiates inter-datacenter traffic classes in an online manner.

First, we built a novel correlation-based classification module combining a cost-sensitive approach with a Bagged Random Forest ensemble algorithm (BRF), to address the interclass imbalance problem while meeting the critical time requirements of the different traffic classes. In order to calculate interflow correlations representing the rebalancing weights, we propose Reverse k-Nearest Neighbors (RkNN), a new algorithm that outperforms several data level, algorithm level and cost-sensitive strategies on four real-world datasets.

The results reveal that the proposed algorithm outperforms most approaches in the different datasets in terms of precision, recall, F1 measure, AUC, and Kappa. The other algorithms resulted in either high precision with low recall or low precision and high recall causing congestion or resource over provisioning.

The outcome of the classification module represents key parameters to characterize the entering traffic in the scheduling module described through an optimization problem, which besides guaranteeing better QoS performances measured by optimal Flow Completion Times, also targets cost minimization by proposing a cost-effective resource provisioning strategy for inter-data center networks.

While other approaches result in high loss rates, our approach preserves the quality and the amount of exchanged traffic. Also, our approach outperforms existing approaches, particularly in the generalization aspect, since our scheduling approach is an online, multi-class scheduling method. The most time-consuming part of our method is tuning (which is still quite negligible).

Keywords: scheduling, optimization, information-agnostic, classification, data-imbalance, ensemble algorithms

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

QoS	Quality of Service
QoE	Quality of Experiance
CAPEX	Capital Expenditures
OPEX	Operating Expenditure
DSCP	Differentiated Services Code Point
TCAC	Taux de Croissance Annuel Composé
DC	Data Center
CD	Centre de données
CSP	Cloud Service Provider
FCT	Flox Completion Time
HTTP	HyperText Transfer Protocol
RkNN	Reverse k Nearest Neighbor
kNN	k Nearest Neighbor
MILP	Mixed Integer Linear Programming
WAN	Wide-Area Network
MLQ	Multiple Level Queuing
SJF	Shortest Job First
SRTF	Shortest Remaining Time First
ECN	Explicit Congestion Notification
PQ	Priority Queuing
ML	Machine Learning
SMOTE	Synthetic Minority Oversampling Technique
TCP	Transmission Control Protocol
CNN	Condensed Nearest neighbor
EV	Expending Vector

GMM	Gaussion Mixture Model
LSTM	Long Short-Term Memory
KDE	Kernel Dencity Estimation
BCA	Byte Classification Accuracy
DT	Decision Tree
TL-BOOST	Tomek-Link BOOST
ROS-BOOST	Random Over Sampling-BOOST
RUS	Random Under Sampling
SDN	Software Defined Network
NFV	Network functions virtualization
RF	Random Forest
OOB error	Out Of Band error
UDP	User Datagram Protocol
IP	Internet Protocol
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives
AUC	Area Under the Curve
ROC	Receiver Operating Characteristic
TPR	True Positive Rate
FPR	False Positive Rate
MINLP	Mixed Integer Nonlinear Programming
GAP	Generalized Assignement Problem
MCF	Muli Comodity Flow

INTRODUCTION

Ce chapitre présente la problématique de recherche de ce mémoire ainsi que les motivations qui nous ont poussés à étudier la planification entre centres de données dans un environnement agnostique à l'information.

0.1 Contexte général

0.1.1 Réseaux d'interconnexion de centres de données et chaine d'acheminement

Avec la prolifération du nuage informatique, de nombreux prestataires migrent leurs services vers des sites distribués causant ainsi l'augmentation du trafic échangé entre les centres de données. Ce trafic doit être sérieusement pris en charge afin de répondre aux exigences des demandeurs de services en termes de qualité de service (QoS) et qualité d'expérience (QoE) et par conséquent d'éviter la perte de clients, mais aussi des frais supplémentaires CAPEX ou même OPEX qui pourraient en résulter. Alors que les réseaux des centres de données (intra) ont reçu une attention significative pour la mise en place d'approches d'ingénierie du trafic efficaces, les performances d'allocation des ressources et d'interconnexion entre les centres de données géographiquement distribués, ont eux reçu moins d'intérêt. Cela est principalement dû au fait que les fournisseurs de services n'avaient pas prévu l'inefficacité des réseaux traditionnels à garantir les débits actuels et à affronter les diverses exigences de QoS.

De plus en plus, de nombreux travaux se penchent sur la question de routage/ingénierie de trafic visant l'optimisation des ressources disponibles, cependant à elle seule, cette approche demeure incomplète comme le montre la figure 0.1. Cette figure, représente "la chaîne d'acheminement des flux", et montre qu'afin d'être acheminés efficacement entre une source et une destination, les flux doivent transiter sur plusieurs modules combinant principalement routage et planification afin d'assurer la qualité requise.

Dès leurs réceptions, les flux passent par un premier module, qui dépendent de certains critères tels que la disponibilité des ressources, procède à l'acceptation ou au rejet de ces données. Lorsque les flux sont acceptés, ils sont transférés au module de classification qui à l'aide de certains paramètres définira leurs classes de service et ainsi leurs priorités. Par la suite, vient l'étape de maintient de l'ordre et marquage, qui consiste en premier lieu a vérifié le respect des limitations de débit sur les liens (tous comme un agent de police sur les routes), dans le cas contraire, les paquets sont rejetés ou bien reclassés, puis un marquage (au paravent DSCP) définissant la politique de qualité de service propre à la classe à laquelle appartiendrait le flux entrant. Une opération très importante suit celle du marquage est la planification, sujet de notre mémoire parallèlement à la classification. Finalement, la mise en forme du trafic consiste à mettre en file d'attente les flux lorsque la capacité des liens ne permet pas de les acheminer et de les transférer lorsque cela devient possible.

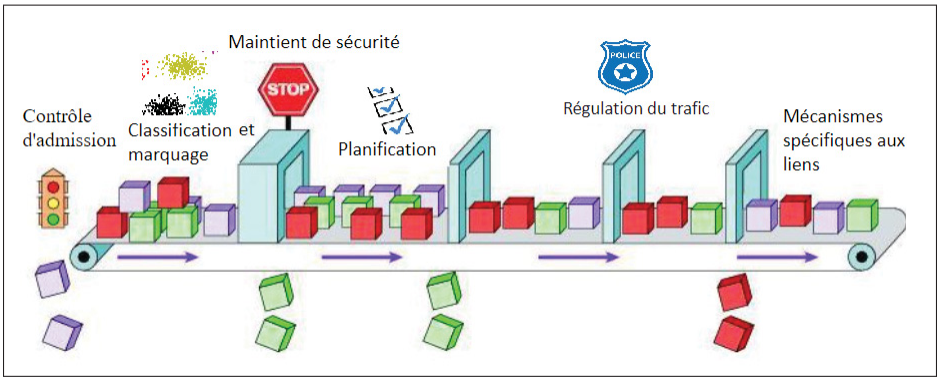


Figure 0.1 Chaîne d'acheminement de trafic

Toutes ces étapes précèdent l'élaboration des algorithmes de routage qui ont souvent était associé à un sur approvisionnement des capacités afin d'éviter les saturations et congestions des liens de transport et ainsi la dégradation de service qui l'accompagne. En effet, les réseaux de transport à large échelle, entre autres, les réseaux interconnectant les centres de données étaient au paravent connectés travers internet employant des liens souvent surdimensionnés à 30, voir 40 % (Wang, Wang, An & Zhang (2019)), ce qui devient économiquement inefficace. De plus, selon une étude

récente effectuée par Cisco, le trafic des centres de données a connu un taux de croissance annuel composé (TCAC) de 23,4% entre 2016 et 2021 atteignant ainsi pas moins de 14,7 zettabytes d'échanges par an, tel que représenté dans la figure 0.2. Cette augmentation entraine à son tours, une importante évolution du trafic entre les centres de données avec un TCAC de 32,7%, pour atteindre 2,8 zettabytes par an.

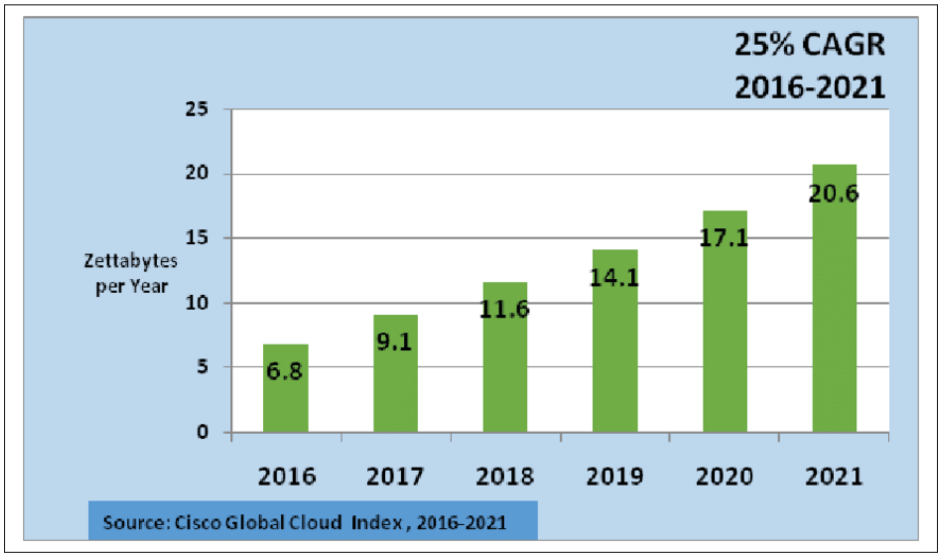


Figure 0.2 Évolution de trafic IP, études CISCO
Tirée de Bisht & Subrahmanyam (2021)

Selon Noormohammadpour & Raghavendra (2017), 45% (qui est en continuelle évolution) du trafic total d'un centre de données est en réalité du trafic provenant ou se dirigeant de/vers un autre ou d'autres centres de données. Cette augmentation de trafic peut entraîner une congestion, une utilisation inefficace des ressources, ainsi que de faibles performances de QoS reflétées par une latence élevée, par exemple. Il s'agit donc d'une urgence pour les fournisseurs de centres de données, qui sont actuellement confrontés à des coûts élevés pour la transmission des flux de trafic sur des centres de données géographiquement distribués, de comprendre et de mettre en place des méthodes efficaces de planification du trafic entre centres de données Hu *et al.* (2018).

Il est à noter que pour les transmissions entre CD (Centre de Données), deux scénarios sont possibles : Le premier scénario représente des réseaux de recherche/expérimentaux à petite échelle où les ressources sont détenues et gérées par des organisations comme le cas de SWAN de Microsoft et B4 de Google. Le deuxième et le plus courant est le scénario de tarification multiniveau figure 0.3 où les transmissions entre centres de données sont effectuées par l'intermédiaire de fournisseurs de services infonuagiques (cloud service providers CSP), facturant les utilisateurs du nuage en fonction du niveau de service requis. Les CSP garantissent la disponibilité et l'évolutivité des plateformes. Cependant, ces services ont un coût non négligeable qui doit être rentable pour les utilisateurs du cloud. Selon Dong *et al.* (2019), le trafic réseau peut être instable et même imprévisible, et les ressources ne peuvent pas être constamment disponibles. Habituellement, pour garantir la qualité de service, les utilisateurs du cloud choisissent la sécurité et sélectionnent le service le plus cher, même s'il n'est pas nécessairement le plus adéquoi, ce qui entraîne un surdimensionnement et donc des coûts de transmission élevés.

Il devient donc impératif de mettre en place des approches optimisant les ressources onéreuses disponibles en coordonnant les atouts de chacune des étapes dans la chaîne d'acheminement des flux (figure 0.1). À travers notre travail, nous supposons l'établissement préalable des règles de routage et nous nous concentrons principalement sur l'étape de la planification des flux.

0.2 Planification des flux échangés entre les centres de données : planification agnostique Vs planification consciente

Deux approches possibles existent pour la planification de trafic : l'approche consciente en opposition à l'approche agnostique. La plupart des travaux antérieurs sont décrits comme des approches de planification conscientes, ces dernières requièrent des informations détaillées sur les flux entrants, telles que la charge totale de trafic au début de la connexion, la taille des flux restant pendant un processus d'acheminement, la distribution du trafic, etc.

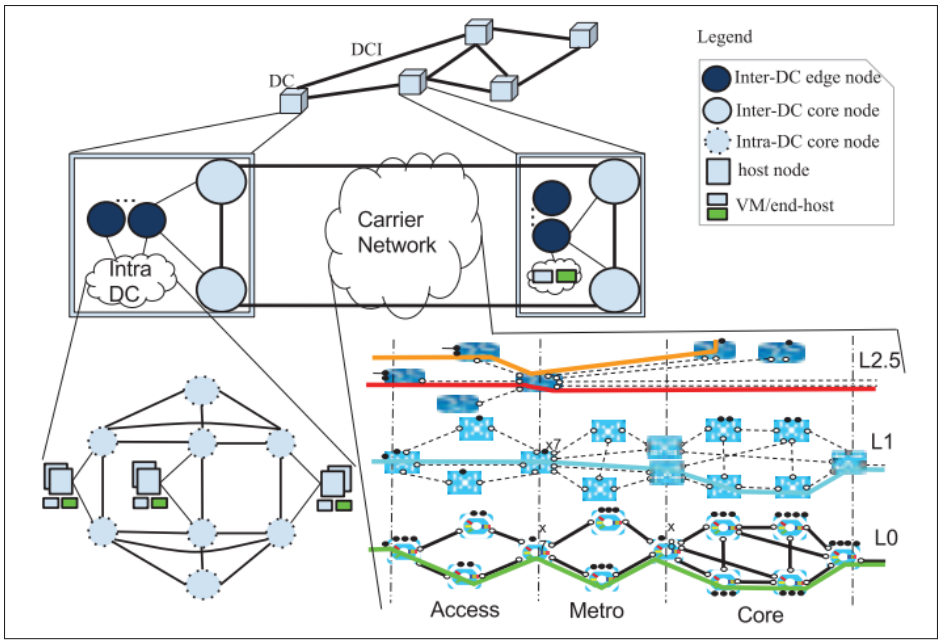


Figure 0.3 Interconnection de centre de données à travers un réseau intermédiaire et tarification multi-niveau
Tirée de Subedi (2018)

Cette hypothèse de connaissance préalable des informations caractérisant les flux entrants garantit de meilleures performances, nous citons par exemple des temps de transfert (FCT, Flow Completion Time) plus faibles Lei, Li, Xing, Jin & Wang (2018); Truong-Huu, Gurusamy & Girisankar (2017). Cependant, pour plusieurs raisons, comme celles énumérées ci-dessous, cette approche idéaliste est loin de la réalité des données et des plateformes actuelles, et laisse plutôt place à la planification agnostique des flux entre centres de données.

- En plus du fait que la taille des flux peut être inconnue pour plusieurs applications Dong, Munir, Tout & Ganjali (2021), les algorithmes d'ingénierie de trafic et la planification de flux en temps réel sont exécutés alors que les flux sont en cours de réception et ne peuvent donc pas attendre la réception de tous les flux afin de déterminer leurs tailles, car cela affecterait la qualité de service Saber, Ghorbani, Bayati, Nguyen & Cheriet (2020).

- Une solution serait de connaître à l'avance les débits des flux des utilisateurs finaux. Cependant, il n'existe actuellement aucune interface utilisateur appropriée à partir de laquelle les applications peuvent spécifier leurs besoins, et même si nous supposons la faisabilité de la mise en œuvre de cette interface, une transmission dynamique de ces informations serait extrêmement difficile et sera livrée avec un coût non négligeable, car il consiste à changer la structure des nœuds de fin et de transit Xie *et al.* (2017).
- De plus, si nous considérons que la prédiction de la taille du flux pourrait représenter la solution, les centres de données hébergent une variété d'applications avec un large spectre de tailles de flux, ce qui introduit un défi important pour les stratégies de prévision du trafic réseau, communément appelé "incertitude de la demande". Chen *et al.* (2016).

0.3 Problématique et questions de recherche

En fonction de son critère de sensibilité au temps, le trafic entre centres de données peut être classé en trois catégories : "le trafic interactif", hautement prioritaire, car il transporte de petits flux avec des contraintes de temps critiques, "un trafic élastique" qui a des exigences de retard plus souples, mais qui doit être transféré dans un délai imparti et pour finir "le trafic de fond" caractérisant des échanges de données de taille importante et généralement établie pendant les heures creuses, car ces flux peuvent avoir une large tolérance au délai long ou pas de contraintes de délai du tout.

La plupart des ouvrages existants se concentrent sur le trafic élastique, car il représente le pourcentage le plus élevé du trafic total entre centres de données en termes de nombre de flux Jalaparti, Bliznets, Kandula, Lucier & Menache (2016). En ce qui concerne les deux autres catégories, les méthodes de planification existantes réservent un pourcentage de bande passante au trafic critique, car il doit être transmis immédiatement et suppose que le trafic de fond n'a pas de contraintes de temps. Cependant, une fois de plus, les utilisateurs du cloud ne peuvent pas

se permettre une utilisation insuffisante de la bande passante, en particulier la congestion des liaisons qui se produit fréquemment dans les chemins les plus courts et qui peut dégrader les performances QoS et par conséquent la QoE des applications finales Sun, Zhao, Fang & Cui (2018).

Cette exploitation inefficace des ressources est la conséquence de l'hypothèse que les utilisateurs ont la capacité de communiquer préalablement les exigences de leurs transferts, en particulier les tailles de flux/demande ou les distributions, ce qui n'est plus réaliste dans les centres de données d'aujourd'hui Dong *et al.* (2019).

En fait, l'un des problèmes actuels dans les réseaux interconnectant les centres de données est que la plupart des approches existantes supposent ou même imposent que la couche de contrôle permet aux applications de spécifier leurs besoins Gerstel, Lopez & Siracusa (2015) et bien que ce scénario d'ordonnancement conscient de l'information soit théoriquement possible, il ne peut représenter un cas d'utilisation réel où pour de nombreuses applications, fournir cette information est difficile et parfois même impossible. Malheureusement, ces applications sont de plus en plus présentes dans les centres de données. Nous citons : les transferts HTTP par morceaux, le traitement de flux avec la canalisation où la taille du flux ne peut pas être obtenue au début de la transmissionXie *et al.* (2017); Wang, Xu & Liu (2017).

Plusieurs travaux Dong & Cai (2022) démontrent la variabilité accrue et la dynamité du trafic des centres de données et précisent que dans ces circonstances les informations caractérisant ces flux peuvent être difficile à obtenir voir impossible pour certaines applications, en particulier le volume de trafic, sa distribution ou même ses contraintes en terme délai dans certain cas. Afin de remédier à cette limitation, les utilisateurs du cloud se contentent de payer plus. En effet, afin de minimiser les risques de dégradation de QoS, les utilisateurs se trouvent face au dilemme de sélectionner un niveau de service sans information précise concernant la charge de trafic qui sera échangée. Ainsi, selon Dong *et al.* (2019), 64% des utilisateurs acceptent de payer un meilleur

service, même si le service n'est peut-être pas nécessaire et est certainement plus onéreux, juste afin de garantir une transmission dans des pénalités de retard, ce qui aggrave le problème de surdimensionnement.

Il s'agit d'une planification agnostique entre centre de données et donc indépendante de l'information, qui vise à répartir efficacement les transferts sans connaissance complète et préalable des caractéristiques du trafic.

De plus, alors que la plupart des travaux existants agnostiques aux caractéristiques des flux Dong *et al.* (2019); Zhang *et al.* (2015); Kandula, Menache, Schwartz & Babbula (2014); Dong, Li, Zhou, Li & Qi (2020), se concentrent sur la QoS, en particulier la minimisation des délais de transmissions des flux sensibles ; la planification rentable n'a pas été suffisamment étudiée pour ce scénario, encore moins en prenant en considération toutes les classes de trafic entre autres, et le trafic sensible et le trafic gourmand en bande passante.

Ce problème est d'autant plus critique que plusieurs travaux considèrent des contraintes de délais exacts uniquement. Ces délais expriment une faible tolérance de retard et dans ce cas le transfert est inutile après le délai défini, en opposition au délai souple pour qui le transfert de données se poursuit même après l'échéance, mais dans ce cas la valeur des informations échangées diminue dans le temps. Ce scénario idéal peut entraîner une dégradation sévère des performances, mais surtout être très coûteux, en particulier pour le scénario de planification agnostique de l'information.

À travers ce mémoire nous proposons de résoudre le problème de recherche suivant : Comment construire une solution de planification de trafic entre réseaux de centres de données qui est à la fois rentable (non coûteuse), mais qui garantit par la même occasion les exigences de QoS de flux appartenant à plusieurs classes de trafic et cela dans un environnement agnostique à l'information ?

Afin de résoudre ce problème et mettre en place notre méthodologie ainsi détaillée dans les prochains chapitres, nous répondons à travers notre travail aux sous-questions de recherches (QR) suivantes :

- QR 1. Avec la variété des trafics réseau actuels et sous la contrainte de non-disponibilité des informations représentant les flux entrants, comment caractériser ces flux de la façon la plus rapide, voire même en temps réel afin de répondre aux exigences critiques de QoS principalement les délais d'acheminement, mais aussi respecter les attentes des usagers en termes de QoE ?

  En fonction de l'approche de classification adoptée, le trafic des centres de données peut être identifié selon l'application source et des protocoles de communication, on décrit cette méthode comme la classification multiclasse, en opposition à la classification binaire qui considère principalement la taille des flux et qui identifie le trafic comme flux souris faisant référence à des flux de petite taille et flux éléphant représentant des flux de taille non négligeable. Le trafic des réseaux de centres de données pose un problème souvent non pris en charge (des hypothèses sont déployées à la place), qui est le déséquilibre des données qui apparaît comme un volume beaucoup plus important pour une ou plusieurs classes en comparaison aux autres classes. La classification du trafic devient plus difficile, car les algorithmes d'apprentissage automatiques se concentreront principalement sur les classes majoritaires, diminuant les performances de classification et provoquant ainsi une dégradation de la QoS.
- QR2. Comment mettre en place une approche de planification optimisant les délais de transmission tout en assurant un partage efficace des charges afin d'éviter la congestion des liens les plus courts particulièrement lorsque la charge et la distribution du trafic entrant sont non disponibles ?

  Un autre défi abordé par notre approche est l'évitement de la congestion. Il est important de maintenir l'équilibrage de charge entre les chemins multiples, surtout lorsqu'aucune

information sur les flux entrants dans le réseau n'est connue. La plupart des informations antérieures agnostiques déploient des approches de contrôle de la congestion tel que la notification explicite de la congestion, qui peut causer un retard supplémentaire et augmenter la perte de paquets si elle n'est pas prise en compte rapidement.

- QR3. Comment formuler et intégrer les coûts d'acheminement dans le modèle mathématique d'optimisation sans compromettre les performances de ce modèle en termes de délai ?
  Dans l'absence des caractéristiques de trafic, les utilisateurs de nuage informatique choisissent la sécurité et choisissent le service le plus cher, pour garantir la qualité de service, même s'il n'est pas obligatoirement nécessaire, ce qui entraîne une surconsommation et donc des coûts de transmission élevés.

0.4 Objectifs

L'objectif principal de ce mémoire est de construire un module de planification rentable s'adaptant aux besoins des différentes classes de trafic, dans un environnement agnostique à l'information. Cet objectif peut être divisé en trois sous objectifs, les suivants :

- O1. Proposer une méthode de classification en ligne et qui soit optimale en présence de données non balancées.
  Nous proposons une approche de classification sensible aux coûts et basée sur la corrélation, afin de faire face au problème de déséquilibre des données tout en assurant une classification en ligne du trafic.
  Nous proposons ainsi une stratégie de classification basée sur l'algorithme RkNN (Reverse k Nearest Neighbor : k plus proche voisin inversé) en construisant un algorithme de résolution basé sur une version modifiée du kNN. Cet algorithme intègre une nouvelle entité représentant la distance entre les nouveaux flux et le sac de flux entraînant l'algorithme d'ensemble.

- O2. Construire un modèle d'optimisation avec la fonction objective qui vise à minimiser les coûts opérationnels, sous la contrainte de délai dur pour les flux de souris et des délais doux exprimés par une meilleure tolérance de retard pour les flux d'éléphants.

  Afin de répondre aux questions de recherche, nous proposons de représenter le problème de planification par un modèle d'optimisation suivant une méthode de programmation non linéaire en nombres entiers mixtes (MILP Mixed Integer nonlinear programming). Notre modèle représentera les besoins en latence des différentes classes de trafic, mais aussi les coûts de traitement des flux dépendamment des chemins parcourus. Nous proposerons par la suite une méthode de résolution consistant à relaxer le problème en deux sous problème et à proposer un algorithme de résolution basé sur des solvers linéaires.
- O3. Au-delà de la garantie des exigences QoS et QoE pour toutes les classes, ce travail vise à construire une stratégie d' approvisionnement de ressources rentable entre les DC afin d'optimiser l'utilisation des ressources réseau tout en évitant le problème de congestion des liens.

  Afin de répondre à la question 3, nous proposons l'optimisation des coûts de traitement tout en évitant la congestion. Le défi réside dans l'incertitude apportée par l'aspect agnostique de l'information du problème. En d'autres termes, comment planifier les liens composant les chemins de routages avec des informations incomplètes sur le trafic entrant ? Pour ce faire, nous proposons un routage multichemin minimisant les coûts de transport, ainsi un des sous-problèmes que composera notre problème à relaxer (comme détaillé dans le sous-objectif précédent) est un problème d'assignation de flux avec minimisation de coût que nous proposons de résoudre dans notre l'algorithme.

0.5 Hypothèses

La catégorisation des flux de trafic en différentes classes grâce à la classification en ligne est en harmonie avec la contrainte d'incertitude du trafic. Ainsi, l'intégration d'un module de classification contribuera à l'amélioration des performances du module de planification.

De plus, bien que la plupart des travaux se concentrent sur les flux courts sensibles à la latence, optimiser également les performances des flux longs, permettra d'améliorer la qualité d'expérience des utilisateurs finaux.

0.6 Contributions

Les contributions que nous apportons à travers ce mémoire sont les suivantes :

- Le développement d'une méthode de classification en ligne s'adaptant à la dynamique du trafic entre centres de données, et abordant le problème de débalancement des données sous la contrainte de non-disponibilité des informations caractérisant les flux entrants.
- La modélisation du problème de planification sous les contraintes de qualité de toutes les classes de trafic échangées et ce dans un environnement agnostique aux informations des flux.
- Le développement et l'intégration d'un modèle caractérisant les coûts de transmission entre centres de données au modèle de planification précédent avec la contrainte de l'environnement agnostique.
- L'élaboration d'une approche de prévention de congestion en accord avec le modèle d'optimisation déployé sans disposer de certaines informations telle que la taille ou la distribution des flux entrants.

Les résultats obtenus ont fait l'objet de deux publications acceptées et d'une publication soumise.

- Article de journal : Saber, M. A. S., Ghorbani, M., Bayati, A., Nguyen, K. K., & Cheriet, M. (2020). Online data center traffic classification based on inter-flow correlations. IEEE Access, 8, 60401-60416.
- Article de conférence : Amina, SI SABER Meriem, Bayati Abdolkhalegh, Nguyen Kim Khoa, and Cheriet Mohamed. "Featuring real-time imbalanced network traffic classification." In 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pp. 840-846. IEEE, 2018.
- Article soumis à ICC 2025 - IEEE International Conference on Communications : Si Saber Meriem Amina, Mohamed Cheriet, A Deadline-Constrained and Cost-Minimized approach for inter-datacenter agnostic scheduling.

0.7 Plan du mémoire

Le premier chapitre représente le chapitre introductif. Il comprend une mise en contexte et détaille notre problématique de recherche ainsi que les questions de recherche auxquelles cette problématique se départage. On y présente aussi nos contributions et nous le clôturons par le plan de thèse.

Le second chapitre représente la revue de littérature. Le chapitre décrit en premier lieu le processus de planification des flux et définit la planification agnostique en opposition à la planification qui contient des flux. Les approches existantes pour la planification agnostique des flux échangés entre les centres de données sont par suite listées et comparées en fonction de ou bien des classes prises en considération, en fonction du critère d'optimisation ainsi que du système déployé. Un tableau comparatif clôture de chapitre.

Le chapitre 3, porte sur la classification des flux échangée entre les centres de données. Il définit pour commencer le processus de classification, particulièrement les caractéristiques des flux de souris et éléphants. Il expose le problème de déséquilibre des données ainsi que les solutions existantes pour le résoudre. À travers ce chapitre, nous présentons aussi la différence entre la classification en ligne et la classification hors ligne.

Le chapitre suivant présente la méthode de travail suivie pour la résolution de notre problématique de recherche. Il expose le système d'échange de flux déployé ainsi que ses caractéristiques. Par la suite, il cite les hypothèses sur lesquelles est basé notre modèle de planification ainsi que les objectifs visés par ce dernier. La méthodologie suivie est finalement détaillée avec la solution proposée.

Le chapitre 5, détaille le modude de classification construit afin d'apporter l'aspect dynamique au module de planification. Il commence par présenter quelques fondements théoriques employés dans l'algorithme de classification déployé. Par présenter les différents algorithmes existants types ensemble dans la section suivante. Par la suite, il présente la solution proposée au problème de classification en ligne des données non balancées échangées entre les centres de données. Dans la prochaine partie de ce chapitre, il cite le processus d'expérimentation et présente les résultats obtenus et analysés.

À travers le chapitre suivant, nous présentons les résultats obtenus lors de l'expérimentation du module de planification. Le chapitre détaille alors le protocole d'expérimentation suivi, les différents scénarios étudiés ainsi que les topologies de validation déployées.

Le mémoire est clôturé par un chapitre de discussion des possibles améliorations futures de notre travail.

CHAPITRE 1

REVUE DE LA LITTÉRATURE

Différentes des stratégies Intserv et Diffserv qui ont été déployées dans les WAN traditionnels, mais jamais à grande échelle en raison de certaines limites en matière de gestion et d'évolutivité ; plusieurs approches de planification entre CD ont été proposées. Ces stratégies peuvent être classées en deux catégories : la planification des tâches Joe-Wong, Kamitsos & Ha (2015) et la planification des flux Sun *et al.* (2018).

La planification des tâches ne considère pas l'approvisionnement des ressources réseau et se concentre plutôt sur le placement des requêtes émises par des utilisateurs finaux (tâches ou charges de travail) sur une machine virtuelle puis sur un serveur physique spécifique en fonction de ses performances. Elle concerne généralement le placement de machines virtuelles, la migration, la consolidation... En opposition à la planification des flux, se concentrant plutôt sur le transport efficace du trafic résultant d'une tâche à travers les ressources de réseau disponibles. Dans le présent travail, nous ciblons la planification des flux.

1.1 Définition du processus de planification de flux entre les centres de données

La planification des flux est définie comme la détermination des valeurs optimales d'une ou de plusieurs variables représentant les dimensions de planification, qui construisent la fonction objective à maximiser ou à minimiser afin d'optimiser l'utilisation des ressources.

Plusieurs dimensions de planification existent : l'espace, le temps, la capacité stockage, l'approvisionnement, le coût, le temps d'acheminement, etc., et la plupart des travaux récents formulent le problème de planification comme un problème d'optimisation maximisant le débit/goodput, le taux de réussite, le nombre de blocs complétés du flux avant le délai, le bénéfice récolte, ou encore minimisant le coût de transfert, le temps de transport... etc., Wang *et al.* (2019).

1.2 Planification agnostique et planification consciente des flux échangés entre les centres de données

Il existe deux approches possibles pour la planification de trafic : l'approche consciente en opposition à l'approche agnostique. La plupart des travaux antérieurs sont décrits comme des approches de planification conscientes, ces dernières requièrent des informations détaillées sur les flux entrants, telles que la charge totale de trafic au début de la connexion, la taille des flux restant pendant un processus d'acheminement, la distribution du trafic, etc.

Cette hypothèse de connaissance préalable des informations caractérisant les flux entrants garantit de meilleures performances, nous citons par exemple des temps de transfert (FCT, Flow Completion Time) plus faibles Lei *et al.* (2018); Truong-Huu *et al.* (2017). Cependant, pour plusieurs raisons, énumérées ci-dessous, cette approche idéaliste est loin de la réalité des données et plateformes actuelles, et laisse donc place à la planification agnostique des flux entre centres de données.

- En plus du fait que la taille des flux peut être inconnue pour certaines applications Dong *et al.* (2021), l'ingénierie du trafic et la planification de flux en temps réel opèrent pendant que les flux sont reçus et ne peuvent pas attendre la réception de tous les flux afin de déterminer leurs tailles, car cela affectera la qualité de service Saber *et al.* (2020).
- Une solution serait de connaître à l'avance les débits des flux des utilisateurs finaux. Cependant, il n'existe actuellement aucune interface utilisateur appropriée à partir de laquelle les applications peuvent spécifier leurs besoins, et même si nous supposons la faisabilité de la mise en œuvre de cette interface, une transmission dynamique de ces informations serait extrêmement difficile et sera livrée avec un coût non négligeable, car il consiste à changer la structure des nœuds de fin et de transit Xie *et al.* (2017).
- Aussi, si nous considérons que la prédiction de la taille du flux pourrait représenter la solution, les centres de données hébergent une variété d'applications avec un large spectre de tailles de flux, ce qui introduit un défi important pour les stratégies de prévision du trafic réseau, communément appelé incertitude de la demande. Chen *et al.* (2016).

1.3 Approches existantes pour la planification agnostique des flux échangés entre les centres de données

Les stratégies existantes d'ingénierie de trafic abordent le problème de la planification des flux de deux façons différentes, la première est d'exiger la disponibilité des données caractérisant le trafic entrant (consciente), alors que la seconde, ne requiert aucune ou peu d'information sur ce sujet (agnostique). La méthode consciente vise à maintenir des performances optimales sans tenir compte de la complexité de la méthode proposée qui est principalement causée par l'approche qu'elle suit pour saisir les informations sur les flux, ou en posant certaines hypothèses sur la taille des flux, le délai d'acheminement ou la distribution des flux... ce qui entraîne une perte de généralité ; la méthode agnostique est quant à elle plus réaliste, car elle admet la non-disponibilité de ces informations et peut donc couvrir un large éventail d'applications, mais ses performances peuvent être limitées.

Nous résumons ci-dessous quelques récents travaux portant sur la planification des flux entre les centres de données, identifions leurs limites et insistons également sur l'aspect agnostique/conscient de chaque stratégie en vers certains paramètres (délai, taille des flux). Nous classons ces travaux en deux catégories en fonction de leurs visions du problème de planification et par conséquent leurs objectifs ; la première catégorie considère les travaux qui approchent le problème de planification en fonction des classes de trafic considérées, tandis que la deuxième catégorie inclut les travaux qui abordent le même problème en fonction de la dimension de son modèle d'optimisation.

1.3.1 Première catégorie : Classification des travaux existant en fonction des classes de trafic considérées

En se basant sur le critère 'temps', le trafic entre centres de données peut être classé en trois classes : le trafic interactif qui devrait être hautement prioritaire, car il transporte des flux de petite taille avec des contraintes de temps critiques, le trafic élastique qui a des exigences moins strictes en vers les retards de transport, mais qui doit être transféré avant un délai précis et finalement, le

trafic de fond caractérisant des échanges de données très volumineux et généralement établi pendant les heures creuses, car ces flux peuvent ne pas exiger de délai ou poser un long délai.

Plusieurs travaux de recherches ont étaient effectués afin d'assurer une planification efficace de chacune de ces classes de trafic, même si plusieurs se concentrent principalement sur le trafic élastique en raison de sa forte représentation dans les échanges entre les centres de données Jalaparti *et al.* (2016).

1.3.1.1 Amoeba pour toutes les classes de trafic

L'un des premiers travaux, mais aussi le plus cité est Amoeba Zhang *et al.* (2016), qui cherche à garantir le délai pour autant de transferts que possible afin de maximiser l'utilisation de la bande passante et cela tout en assurant les exigences des trois classes de trafic. Amoeba offre comme illustré sur la figure 1.1 représentant le modèle de son système, une interface où les utilisateurs ont la possibilité de spécifier leurs exigences en termes de délai et propose une stratégie de planification flexible pour les fournisseurs visant à maximiser le nombre de demandes servies avant leur délai grâce à une stratégie de contrôle d'admission tout ou rien.

Cependant, ce concept du tout ou rien mis en place par Ameoba, et qui signifie que si un flux ne peut pas être transmis, il est rejeté, implique que toutes les données reçues après le délai maximum définit par l'utilisateur deviennent insignifiantes, ce qui augmente le pourcentage de perte de paquets et dégrade par conséquent la QoS. De plus, Ameoba exige aux utilisateurs finaux des spécifications claires concernant les exigences de chaque flux, ce qui peut être très onéreux, mais aussi dans certains cas impossible.

1.3.1.2 Trafic élastic

Parce qu'ils doivent être transférés d'une manière efficace, mais surtout rapide, plusieurs travaux sont axés sur les flux élastiques. DCRoute dans N.M, Raghavendra & Rao (2016a) est une approche de planification qui implémente la stratégie «Le plus tard possible» en retardant les flux entrants le plus près possible de leur échéance. Toutefois, en plus de l'aspect conscient du

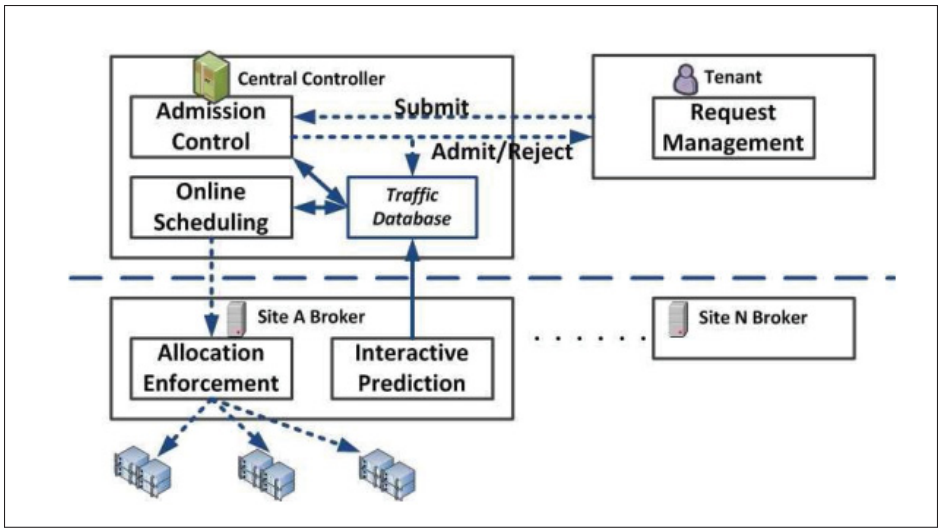


Figure 1.1 Modèle du système Amoeba
Tirée de Zhang *et al.* (2016)

délai et de la taille du flux de DCRoute, les requêtes ne sont acceptées que lorsque le système trouve suffisamment de ressources pour couvrir leur transmission. RCD dans N.M, Raghavendra, Rao & Madni (2016b), maximise l'utilisation des liens et le nombre de flux élastiques transférés avant leur échéance, il est alors lui aussi délai et taille conscient.

Les auteurs de Hu *et al.* (2018) de l'autre côté, formulent le problème de planification des flux entre les centres de données comme un problème d'optimisation visant la minimisation du temps de transfert, tout en considérant l'équité entre les transferts. Les auteurs affirment que ce modèle permettra au réseau de mieux comprendre les conséquences (en termes de délai de transfert) associées à l'attribution de la bande passante, en transformant ainsi le problème de minimisation du temps de transfert en un problème de maximisation de l'utilité. L'utilité est l'unité de mesure qui représente les avantages de l'achèvement d'une certaine tâche dans divers délais, ou au contraire l'inconvénient de ne pas être en mesure d'assurer le transfert. Il est à la fois délai et taille agnostique. Comme illustré dans la figure 1.2, quel que soit le type de la fonction (discret ou continue), plus le délai d'acheminement est réduit, plus la fonction est maximisée.

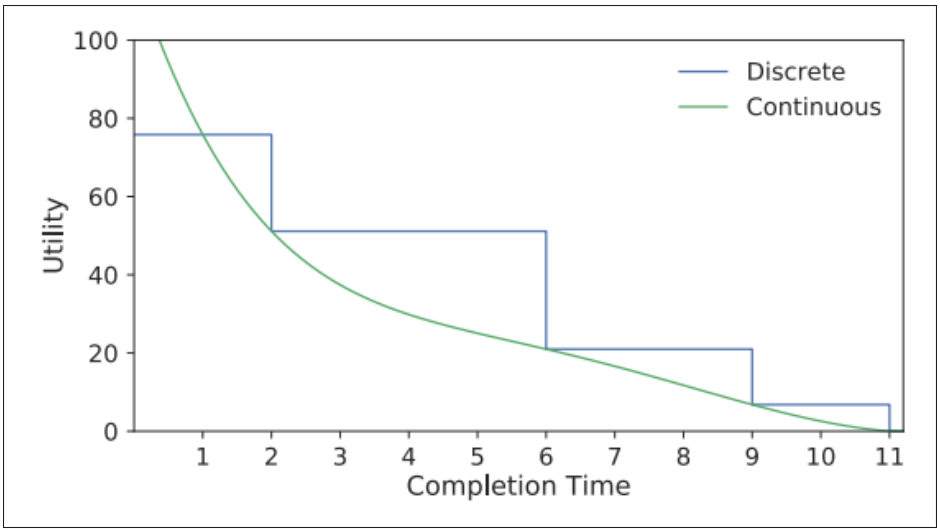


Figure 1.2 Évolution de la fonction utilité en fonction du délai d'acheminement
Tirée de Hu *et al.* (2018)

1.3.1.3 Gros trafic regroupant plusieurs flux de fond

Certains travaux ont porté sur les transferts en vrac (gros flux) entre les centres de données, qui peuvent être assez grands pour même submerger les liens optiques Wu *et al.* (2015b); Luo, Yu, Ye & Du (2018). Luo *et al.* (2018) propose un algorithme de planification temps réel qui vise à optimiser les revenus d'un fournisseur en créant un modèle construit aux tours de la méthode de maximisation des récompenses. La récompense représente les revenus tirés des flux (avec des délais mixtes) en respectant leurs délais. Il intègre également une pénalité pour les flux arrivant après leurs délais. Les auteurs résolvent le problème de planification en supposant en premier lieu la disponibilité de l'information caractérisant les flux dans un cadre non contraint par le temps. Une fois la solution trouvée, ils construisent, suivant la technique de dualité une solution finale temps réel. Wu *et al.* (2015b) se concentre sur les demandes de migration avec différents niveaux de criticité, mesurés en fonction de leur sensibilité aux délais. Ce travail vise à maximiser le nombre de demandes de migration traitées efficacement afin d'optimiser l'utilisation de la bande passante. Cette approche est consciente à la fois au délai, mais aussi la taille.

1.3.2 Seconde catégorie : Classification des travaux existants en fonction du critère d'optimisation

Dans la section précédente, nous avons constaté que de nombreux projets se concentrant sur les différentes catégories de trafic acheminées élaborent un modèle pour maximiser les bénéfices. Néanmoins, d'autres méthodes sont envisageables et sont répertoriées dans la section suivante.

1.3.2.1 Minimisation des coûts

Certains travaux sont portés sur les coûts de transmission du trafic entre centres de données. Comme dans Li, Zhou, Li, Qi & Guo (2018) où les auteurs proposent une approche de planification basée sur les créneaux horaires libres dans un modèle de charge de 95% centiles. Ce travail vise à exploiter les créneaux ignorés par les providers (5 %) car il ne représente que quelques pics du volume total de trafic entre centres de données et est donc considéré comme gratuit. L'idée principale derrière cet algorithme est de programmer autant de trafic que possible pendant les créneaux horaires libres tout en respectant les limites de bande passante et les délais, afin de minimiser le coût de transmission. Les auteurs insistent sur le fait qu'une connaissance approfondie des flux entrants est impossible en pratique, et proposent en conséquence une stratégie temps réel d'allocation qui est formulée en utilisant Lyapunov afin de représenter le compromis entre le coût de transmission et le délai.

Dans une autre approche, aussi agnostique à la taille des flux, les auteurs dans Dong *et al.* (2019) résolvent le problème de l'incertitude (taille des flux non disponible) au moyen d'un modèle de prédiction et sélectionne par la suite le niveau de service adéquat pour chaque demande à court terme afin de résoudre le problème d'optimisation qui vise à minimiser le coût unitaire à long terme. En effet, le problème de planification est représenté comme un problème d'optimisation avec pour objectif : mesurer le nombre optimal de demandes à court terme pour lequel chaque demande à long terme doit être divisée. La figure 1.3 illustre le modèle fractionnant la demande R en 3 sous demandes R1, R2 et R3 avec chacune de nouvelles caractéristiques incluant une nouvelle adresse source et destination, volume de trafic estime (v) et bande passante (b). Cette

méthode vise à minimiser le coût de transmission imposé par les fournisseurs de services aux utilisateurs de cloud suivant un schéma de tarification à plusieurs niveaux.

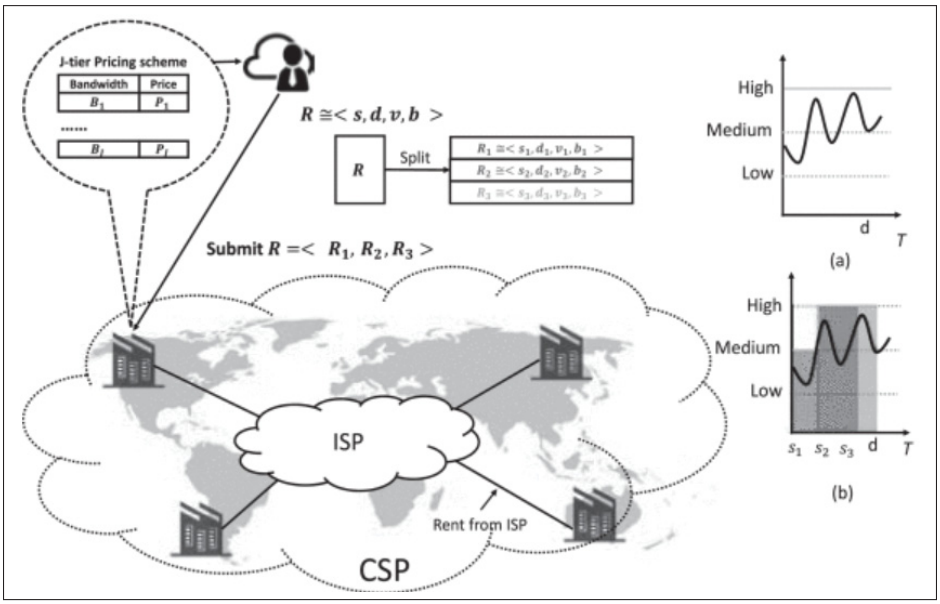


Figure 1.3 Modèle agnostique de partage de demandes
Tirée de Dong *et al.* (2019)

De l'autre côté, les auteurs dans Xu, Li, Qi, Wang & Li (2020) et Li *et al.* (2016), ne mentionnent nullement l'incertitude et fournissent une approche consciente de la taille/de l'échéance. Xu *et al.* (2020) vise à distribuer les demandes entre CD éloignés à des fins de traitement tout en minimisant le coût et le délai de transmission, et Li *et al.* (2016) vise à garantir la disponibilité de la même bande passante que pour la demande avec le coût minimum, les auteurs tentent également d'équilibrer le trafic entre les liens à faible coût et coûteux.

1.3.2.2 Maximisation des revenus

Les auteurs Gandhi *et al.* (2020) se sont concentrés sur les revenus des fournisseurs de services infonuagiques. Gandhi *et al.* (2020) planifie les flux entre centres de données, en maximisant les revenus tout en assurant l'équité entre les flux de diverses tailles connues. Dans un autre ouvrage Yang *et al.* (2019), les auteurs se concentrent sur le profit et insistent sur le fait que la maximisation du profit ne provient pas de la minimisation des coûts de transmission ni de la

maximisation des revenus, mais d'une relation entre les deux, l'approche proposée vise donc à sélectionner les demandes maximisant le profit au lieu de programmer toutes les demandes des utilisateurs. Cependant, des informations détaillées sur les flux sont demandées pour fournir des performances de programmation optimales.

1.3.2.3 Minimisation des délais de transfert

Les approches existantes peuvent être classées en deux catégories. La première est la méthode basée sur les files d'attente à plusieurs niveaux (Multiple Level Queue (MLQ)), a été élaborée pour surmonter les limitations de la méthode de la tache la plus courte (Shortest Job First (SJF)/ Short Remaining Processing Time) qui requiert une connaissance à priori des informations sur les flux entrants. La stratégie SJF privilégie les flux courts et repose sur le déploiement de plusieurs files d'attente avec chacune une priorité allant du niveau le plus haut au plus bas et dans lesquelles les flux transitent selon un certain paramètre. Par exemple, pour Bai *et al.* (2017) c'est le numéro d'octets reçus et appartenant à un flux spécifique. Toutefois, ces approches sont axées sur la conception et ne sont pas modélisées mathématiquement, ce qui limite les possibilités d'améliorer les performances de planification, d'où la deuxième catégorie Hu *et al.* (2016). Avant de présenter quelques travaux appartenant aux approches se basant sur des modèles mathématiques, nous allons en premier lieu aborder les approches basées sur MLQ.

Système MLQ

Afin d'éviter les limitations d'un système centralisé où un serveur collecte des informations sur les flux entrants, comme le numéro d'octet dans le cas de PIAS Bai *et al.* (2017), DIAS Lei *et al.* (2018) introduit une méthode distribuée où les nœuds réseau s'échangent des informations capturant la priorité des flux sous forme du temps d'attente des paquets. De l'autre côté, Wang *et al.* (2017) introduit une approche de priorité à deux niveaux basée sur l'urgence qui est représentée par : le rapport entre le temps de transmission écoulé et le temps restant avant l'échéance.

Abbasloo, Xu & Chao (2018) propose une stratégie hybride combinant à la fois la vision globale des réseaux centralisés et l'agilité des réseaux distribués. Cette approche exploite le routage multichemins dans les centres de données et met en place une stratégie d'équilibrage de charge dans le but de minimiser le temps de transfert. Suivant ce concept, en fonction du nombre de paquets, les flux dépassant un certain seuil (seuil de rétrogradation), et appartenant donc à la classe des flux d'éléphants (flux larges) sont programmés de manière centralisée en raison de leur capacité à tolérer des retards, tandis que les flux de souris (flux avec un nombre de paquets en dessous du seuil de rétrogradation) suivront le mode distribué.

Alors que la plupart des approches se concentrent sur l'élaboration de stratégies de plus en plus complexes et précises, les auteurs dans Abbasloo, Xu & Chao (2020) ont suggéré que suivre la simple stratégie du premier arrivé, premiers servis FCFS (First Come First Serve). L'approche proposée vise à trouver le seuil qui garantit le délai minimum pour les flux courts et est considéré comme l'approche optimale pour un chemin à une seule liaison.

Une caractéristique commune des approches précédentes est qu'elles suivent un mécanisme de file d'attente en réseau, ce qui signifie que l'approche de planification cible l'optimisation du système de file d'attente dans les nœuds réseau. Xie *et al.* (2018) propose une méthode de priorité basée au niveau l'hôte, ainsi la complexité est davantage poussée vers le bord tandis que les nœuds réseau restent simples et utilisent des files d'attente peu profondes du type premier arrivé premiers servis. Ainsi, la perte de paquets et la bande passante utilisée pour la retransmission peuvent être évitées et par conséquent le temps de transfert minimisé. Xie *et al.* (2018) introduit aussi L2DCT et Explicit Congestion Notification (ECN), pour garantir le contrôle des débits et la récupération des pertes.

Cependant, comme nous l'avons mentionné plus tôt, la limitation des approches fondées sur la méthode MLQ vient de son manque de modélisation mathématique. En outre, ces stratégies sont construites autour du seuil de rétrogradation, qui doit être dynamique aux changements de trafic et aussi bien défini afin de maintenir des performances satisfaisantes.

Systèmes basés sur des modèles mathématiques

Afin de faire face au problème d'incertitude, de nombreux travaux ont ciblé l'estimation de la taille des données en entrée ou bien la distribution associée à un paramètre caractérisant ces flux, nous citons par exemple la durée de service, la latence... Nous présentons dans ce qui suit les approches déployées pour intégrer ce module d'estimation dans le processus de planification des flux.

Les auteurs de Hu *et al.* (2016) proposent une méthode mathématique basée sur la stratégie de priorité en file d'attente (Priority Queuing PQ), où ils supposent que la distribution de la taille du flux peut être estimée et modélisent le système comme un modèle de file d'attente M/G/1. Les auteurs de Liu, Xing, Hu, Yu *et al.* (2016a) de leurs côté déploient également un modèle mathématique basé sur le PQ représentant chaque flux avec sa probabilité d'être acheminé dans une certaine période de temps, ainsi les flux avec des probabilités élevées sont affectés à des files d'attente hautement prioritaires. Cette probabilité est intégrée à un modèle mathématique où Liu *et al.* (2016a) suppose la possibilité d'estimer la taille des flux et cible la minimisation des temps d'acheminement pour les flux de souris et l'optimisation du débit pour les flux d'éléphants.

Dans Wang *et al.* (2020), les auteurs proposent quant à eux un algorithme de prédiction à deux étapes. Ainsi, un spectre représentant les tailles de flux est traduit en plusieurs groupes auquel est allouée une priorité. Une fois la priorité des flux établie, le problème est modélisé par un modèle d'optimisation visant à maximiser le pourcentage de flux courts qui doivent scarifier leurs performances afin de fournir une latence optimale.

L'incertitude quant à la durée de service, résultant des défis apportés par la commutation de circuits des réseaux optiques est explorée dans Truong-Huu *et al.* (2017). Les auteurs proposent un modèle probabiliste intégrant le processus de décision de Markov. La décision de planification est obtenue après la résolution d'un problème d'optimisation avec l'objectif de maximiser les revenus des fournisseurs de cloud. Ces revenus dépendent de la récompense obtenue en acceptant un flux entrant sur une certaine longueur d'onde à une plage horaire T et passant un certain temps écoulé sur le chemin lumineux.

1.4 Étude comparative entre les solutions de planification de flux existantes

Afin de mettre en évidence nos contributions dans les sections suivantes, nous résumons dans le tableau suivant les différentes méthodologies, leurs contributions ainsi que leurs limitations.

Comme indiqué dans le tableau récapitulatif (dans la partie concept, nous faisons référence à une méthode agnostique avec la lettre A et à une méthode consciente avec la lettre C), plusieurs travaux traitent de la planification du trafic entre les centres de données, certain ont introduit l'aspect agnostique en vers les informations des flux entrants, d'autre estiment que pour de bénéficier de meilleurs performances, malgré la perte de généralité, il serait préférable de ne pas poser ces contraintes. Ainsi, afin de garantir l'efficacité des approches sans avoir recours à tous les détails des flux entrants, certains travaux déploient des mécanismes de prédiction de la taille des flux ou d'autres paramètres les caractérisant, ce qui compte tenu de la variété des données échangées entre les CD, pourrait poser des problèmes d'efficacité des approches proposées. Aussi, bien que pour certains travaux, elle pourrait aboutir à des résultats appréciés, poser des hypothèses quant à la distribution des flux entrants ne représente pas l'aspect réel des données entre CD. Pour finir, peu de travaux ont ciblé le compromis entre FCT et coût, alors qu'aucun travail n'a abordé ce problème tout en différenciant les exigences pour les trois classes de centres de données.

Tableau 1.1 Étude comparative des approches de planifications de flux existantes

Article	Concept	Stratégie	Limitations
Zhang *et al.* (2016)	C	-Tout ou rien -Interface de spécification de délai	-Stratégie consciente -Perte d'information -Coût de transfert non considéré
N.M *et al.* (2016a)	C	-Le plus tard possible -Trafic élastique	-Stratégie consciente -Une classe de trafic uniquement -Coût de transfert non pris en considération
N.M *et al.* (2016b)	C	-Maximiser l'utilisation des liens -Maximiser le nombre de flux avant délai -Trafic élastique	-Stratégie consciente -Une seule classe de trafic -Coût de transfert absent dans le modèle
Hu *et al.* (2018)	A	-Minimisation du temps de transfert -Équité entre les transferts -Trafic élastique	-Trafic élastique uniquement -Coût de transfert non pris en considération
Luo *et al.* (2018)	C	-Optimisation des revenus -Temps réel -Trafic de fond	-Stratégie consciente -Une seule classe de trafic -Coût de transfert absent dans le modèle
Wu *et al.* (2015b)	C/A	-Maximisation du nombre de demandes de migration avant delai -Dualité des deux stratégies	-Trafic de fond uniquement -Coût de transfert non pris en considération
Li *et al.* (2018)	A	-Exploitation du modèle du 95 centile -Minimisation des coûts	Pas de différentiation entre classes Pas de compromit délai coût
Dong *et al.* (2019)	A	-Fractionner les demandes -Optimisation du nombre de sous-demandes	
Dong *et al.* (2019)	A	-Prédiction de la taille des flux -Minimisation du coût unitaire à long terme	
Dong *et al.* (2018)	A	-Prédiction de la taille des flux -Minimisation du coût unitaire à court terme	

Tableau 1.2 Étude comparative des approches de planifications de flux existantes (suite)

Article	Concept	Stratégie	Limitations
Xu *et al.* (2020)	C	-Minimisation des coûts et délai de transmission	-Stratégie consciente -Stratégie de prédiction inefficace de par le large spectre des tailles de flux
Li *et al.* (2016)	C	Équilibrage des trafics entre les liens	-Stratégie consciente -Stratégie de prédiction inefficace de par le large spectre des tailles de flux
Gandhi *et al.* (2020)	C	-Maximisation de revenus -Equité entre flux	-Le modèle n'intègre pas le délai -Approche consciente
Yang *et al.* (2019)	C	-Minimisation des coûts de transmission	
Lei *et al.* (2018)	A	-Minimisation des délais -Système de files d'attente avec seuil	-Temps d'attente important -Pas de modélisation, pas de possibilité d'amélioration -Seuil non dynamique aux variations de trafic -Difficulté de modélisation du seuil dans un environnement agnostique
Wang *et al.* (2017)	A	-Système de file d'attente -Seuil définit par le rapport de priorité	
Abbasloo *et al.* (2020)	A	-Système FIFO -Modélisation du seuil	
Xie *et al.* (2018)	A	-File d'attente niveau hôte -Mimimisation du délai	
Hu *et al.* (2016)	A	-Modèle de file M/G/1	-Pas de differentiation entre les classes -Impose la supposition/ estimation ou prédiction
Liu *et al.* (2016a)	A	-Système probabiliste -Estimation de la taille des flux	
Wang *et al.* (2020)	A	-Prédiction de la taille	
Truong-Huu *et al.* (2017)	A	-Décision de Markov	

CHAPITRE 2

CLASSIFICATION EN LIGNE DU TRAFIC DES CENTRES DE DONNÉES BASÉE SUR LES CORRÉLATIONS ENTRE FLUX

2.1 Introduction

Dans ce chapitre, nous allons aborder les concepts et théories derrière la classification de trafic. Nous présenterons la structure générale d'un système de classification afin d'examiner le module contrôlant l'aspect en ligne et hors ligne de l'identification des flux. Nous mettrons par la suite en revue des travaux connexes ayant abordé la classification entre centres de données, notamment ceux mettant en évidence le déséquilibre des données. Nous concluons ce chapitre en discutons des limitations des approches présentées afin de justifier la démarche que nous entreprenons dans la méthode de classification proposée dans les chapitres à venir.

2.2 Définition de la classification de trafic entre centres de données

La classification du trafic représente une étape clé contrôlant l'efficacité des mécanismes de qualité de service (QoS). En effet, les stratégies de contrôle de trafic qui gèrent les débits de transmission ainsi que les congestions du réseau, parallèlement aux approches de planification et d'ingénierie de trafic instaurant les politiques de priorités, dépendent principalement de l'efficacité de la méthode de classification, car elle fournit les outils nécessaires pour mieux comprendre les tendances de trafic et ainsi garantir les performances de qualité de service (QoS) requis, mais aussi résoudre et les problèmes d'évolutivité.

Le trafic des centres de données peut être classé selon deux différentes approches : la première, une classification multiclasses, consiste à identifier les applications ayant générées les paquets et déterminé les flux selon les protocoles de communication ; la seconde, le sujet de cette recherche, est binaire et considère particulièrement les tailles de flux, elle identifie le trafic comme *éléphant* ou *souris* Collell, Prelec & Patil (2018). Le choix de la granularité de la méthode de classification (binaire ou multiclasse) dépend de son niveau d'application sur le réseau (access, metro, core...) et par conséquent des informations disponibles pour caractériser le trafic entrant et la structure

le transportant. Puisque nous nous concentrons sur les réseaux interconnectant les centres de données et donc le niveau cœur, nous ciblons alors la classification binaire qui caractérise les trafics entrant en deux class : la classe des éléphants et la classe des souris. Bien que de nombreuses méthodes de classification ont été mises en place avec chacun des avantages mais aussi des limitations, nous citons par exemple la classification en fonction de l'adresse source, ou du port source et destination, la classification selon le protocole ..., l'approche qui a démontré son efficacité, dynamique et adaptabilité aux variations de trafic et environnement, demeure la classification construite aux tours des algorithmes machine learning. La figure 2.1 Hu *et al.* (2021) illustre une méthode de classification de trafic basée sur les réseaux de neurones.

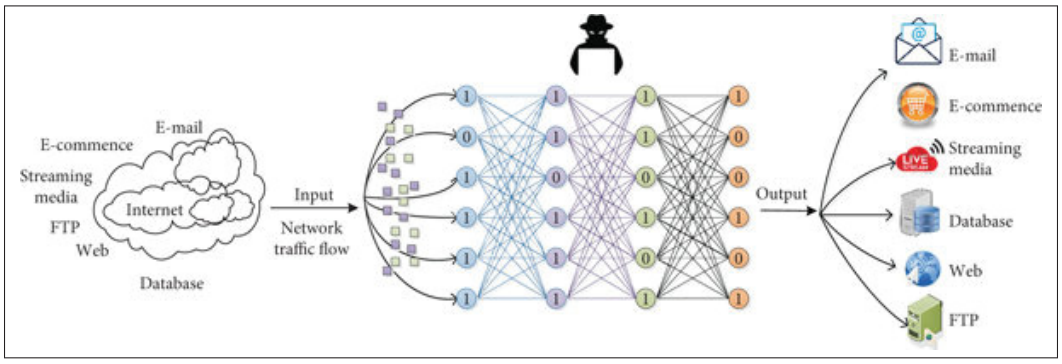


Figure 2.1 Modèle d'un système de classification basé sur les réseaux de neurone
Tirée de Hu *et al.* (2021)

2.2.1 La classe de trafic éléphant Vs la classe de trafic souris

Les flux éléphants sont caractérisés par une longue durée de vie (taille) et représentent par conséquent un pourcentage notable de tous les paquets transmis (80% du trafic total de DC), en opposition aux flux de souris, qui contribuent avec un pourcentage moins important, mais sont échangés plus fréquemment Tang, Li, Barolli & Tang (2017). Les flux éléphants peuvent causer la congestion des files d'attente et des liens de transport, retardant ainsi les flux souris sensibles à la latence et entraînant une dégradation non négligeable des performances du réseau. Les flux souris, d'autre part, contribuent à générer plus de requêtes vers le plan de contrôle, ce qui peut

entraîner une gigue importante, et également dégrader la qualité de service du réseau (QoS) Kiran & Chhabra (2019).

La performance optimale ne peut alors être garantie qu'en résolvant l'équation du compromis entre un débit élevé et une faible latence des deux classes. Ainsi, une compréhension plus approfondie des schémas de trafic peut considérablement améliorer le rendement du mécanisme de contrôle, ce qui se traduit par réseau adaptatif/dynamique et qui construit des CD plus efficaces en termes de consommation de ressources (énergie, bande passante, etc.)Noormohammadpour & Raghavendra (2017) et de sécurité (détection du trafic malveillant) Gómez, Martínez, Sánchez-Esguevillas & Callejo (2017b).

2.3 Déséquilibre des données

Un des problèmes majeurs qu'introduit la classification du trafic des centres de données est le déséquilibre des données. Un ensemble de données déséquilibré implique deux types de classes : la plupart des flux sont contenus dans la ou les classes majoritaires alors que seuls quelques échantillons sont inclus dans la ou les classes minoritaires Peng, Zhang, Yang & Chen (2014), figure 2.2. Dans notre cas, il apparaît entre le petit nombre de flux d'éléphants (donc, représentant la la classe minoritaire) et la quantité massive des flux de souris (comme la classe majoritaire). La classification de ce type de trafic devient alors plus difficile puisque les algorithmes d'apprentissage automatique se concentreront principalement sur les classes majoritaires, auront ainsi tendance à prédire ces classes et à ignorer les instances de classe minoritaire, biaisant par conséquent les résultats de classification Shi, Li, Zhang, Cheng & Cao (2018) et provoquant donc une dégradation de la qualité de service.

Afin de pallier à ce problème, des stratégies de classification efficaces fondées sur le rééquilibrage des données doivent être déployées afin d'éviter la congestion engendrée par les faux positifs (flux d'éléphants) prédits dans la classe des flux de souris) ou un approvisionnement excessif causé par faux négatifs (les flux de souris prédits comme des éléphants).

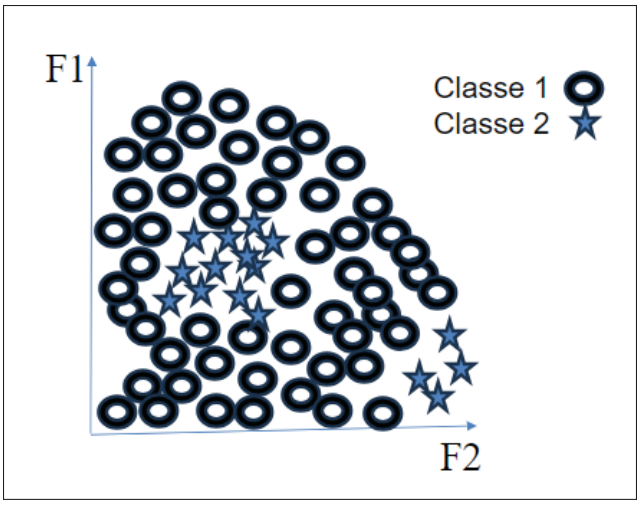


Figure 2.2 Déséquilibre des données

Bien que la plupart des travaux antérieurs aient porté sur l'efficacité des algorithmes d'apprentissage automatique (Machine Learning ML), peu d'études ont tenu compte de la classification en présence de problèmes de déséquilibre des données. La plupart des approches de classification supposent l'uniformité des distributions des flux alors qu'en réalité les trafics présentent naturellement des distributions déséquilibrées Peng, Zhang, Chen & Yang (2017); Gómez, Martínez, Sánchez-Esguevillas & Callejo (2017a).

2.4 Solutions existantes au problème du déséquilibre des charges lors de la classification de trafic

Les auteurs de Gómez, Hernández-Callejo, Martínez & Sánchez-Esguevillas (2019) soulignent l'impact du déséquilibre des données dans la classification du trafic réseau et insistent sur le fait que la base de données regroupant les travaux précédents qui résolvent ce problème n'est pas riche. En effet, la plupart des travaux antérieurs portant sur l'identification des flux souris et éléphant Ding, Liu, Qin & Li (2017); Kiran & Chhabra (2019) ne tiennent donc pas compte du problème du déséquilibre des données, mais se concentrent plutôt sur l'amélioration des performances de classification du trafic dans des scénarios idéaux ; d'autres ne classent que les classes majoritaires, et n'implémentent ainsi pas de stratégies de rééquilibrage et, par

conséquent, les classes minoritaires ne sont pas intégrées dans le processus de classification et de planification éventuellement.

Les solutions existantes, pour résoudre le problème du déséquilibre des données lors de la classification du trafic, relèvent de trois catégories.

- Au niveau de données, Rastegarfar *et al.* (2016); Hasibi, Shokri & Dehghan (2019), trois approches peuvent être utilisées : Le sous-échantillonnage qui consiste à rééquilibrer le rapport entre le nombre d'échantillons des classes en extrayant au hasard autant d'échantillons majoritaires que possible (possiblement des échantillons significatifs). Le suréchantillonnage reproduit les échantillons minoritaires existants (ou crée des échantillons synthétiques ; p. ex., la technique de suréchantillonnage synthétique des minorités Synthetic Minority Oversampling Technique , SMOTE), figure 2.3. L'approche hybride de son côté, commence par la génération de quelques échantillons minoritaires à la suite d'une méthode de suréchantillonnage, après quoi le sous-échantillonnage est introduit pour éliminer les échantillons des classes majoritaires ou des deux classes (majoritaires et minoritaires) Fernández, del Río, Chawla & Herrera (2017).

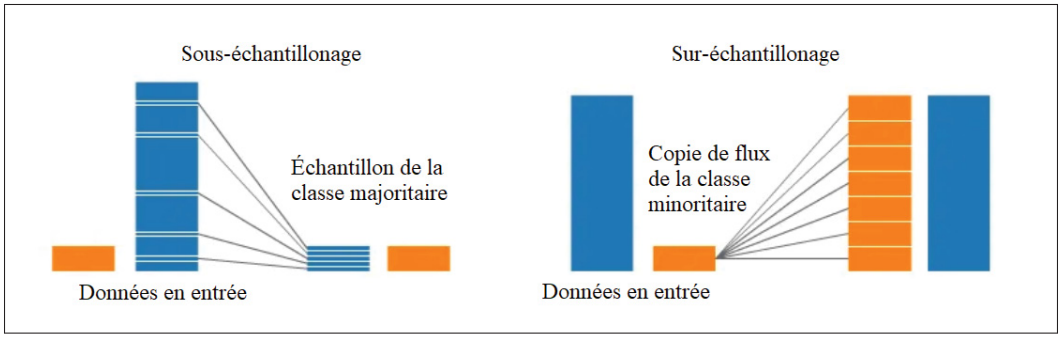


Figure 2.3 Sous-échantillonnage Vs Sur-échantillonnage

- Les solutions algorithmiques Liu, Wang & Tao (2016b); Gómez *et al.* (2019) : Elles sont plus complexes, car elles nécessitent une compréhension complète du raisonnement du classificateur afin de pouvoir envisager les options possibles d'amélioration de l'algorithme Sun, Kamel, Wong & Wang (2007). Récemment, les algorithmes d'ensemble sont devenus l'une des stratégies les plus déployées au niveau des solutions algorithmiques pour le

problème de déséquilibre des données. Même si déclaré comme le plus robuste contre le problème des données déséquilibrées, les algorithmes basés sur les ensembles commencent généralement par rééquilibrer les classes de trafic en utilisant les méthodes du niveau données Zhen & Qiong (2012).

- La dernière catégorie concerne les stratégies sensibles aux coûts. Dans ce cas, la décision de classification est prise en fonction de poids liés aux erreurs de classification et qui sont intégrés au processus Chao, Lin & Chen (2016). La difficulté se pose dans la construction de la matrice des coûts (les coûts liés aux erreurs de classification). Habituellement, les méthodes sensibles aux coûts supposent la disponibilité d'une matrice de coûts ou l'estiment puisque cette matrice devrait être fournie par des experts du domaine d'application Del Río, López, Benítez & Herrera (2014).

La complexité du problème augmente pour les trafics avec des besoins en temps réel, puisque les opérateurs doivent classer rapidement et efficacement les différents flux qui circulent dans leur réseau, à l'appui de leurs divers accords sur les niveaux de service (service level agreements (SLA)) Li, Abdin, Dann & Moore (2013); Tabatabaei, Karray & Kamel (2012).

2.5 Classification en ligne Vs classification Hors-ligne

Afin de définir la différence entre une approche de classification en ligne et une approche de classification hors-ligne, nous allons en premier lieu présenter les différentes étapes construisant un module de classification afin de déterminer l'étape qui contrôle cet aspect en ligne et de hors-ligne.

2.5.1 Système de classification

Comme le montre la figure ci-dessous, un système de classification se compose de plusieurs blocs :

Prétraitement : C'est une étape importante dans la classification des données, car elle vise l'amélioration des performances des algorithmes d'apprentissage automatique. Elle consiste à

éliminer les propriétés indésirables des flux de trafic en introduisant une stratégie de filtrage dans chaque nœud (source de données en ligne) ou encore en appliquant des filtres sur les traces capturées afin d'éliminer les échantillons inutiles.

Extraction des caractéristiques : Niveau d'extraction et directionnalité : Il existe trois niveaux d'extraction des caractéristiques.

- Le niveau le plus bas "caractéristiques du paquet" : ces caractéristiques peuvent être utilisées pour mesurer différents moments statistiques tels que la taille moyenne du paquet, et la variance de temps entre les arrivées.
- Le niveau supérieur, caractéristiques de niveaux flux. Selon la direction de transmission du flux, il existe trois types de flux : (i) Flux unidirectionnels (ii) Flux bidirectionnels et (iii) Flux complets. A ce niveau (flux complet), les caractéristiques représentent le niveau connexion, et impliquent des flux bidirectionnels dans une période allant de l'établissement de connexion jusqu'à sa fermeture (détaillé dans la section suivante). Plusieurs caractéristiques de niveau flux ont été utilisées dans les approches existantes. Ils impliquent des statistiques comportementales telles que la taille du flux et les moments de durée ou des statistiques liées aux informations d'en-tête, telles que le nombre de paquets avec des accusés de réception, avec synchronisation, statistiques liées aux bits de retransmission ou réinitialisation ou encore tout message d'initialisation de session du protocole de contrôle de transport (TCP).
- Variables de niveau connexion : Concerne les statistiques capturées pendant toute la session de transmission telle que la taille de la fenêtre TCP annoncée ou la distribution du débit Nguyen & Armitage (2008); Nguyen, Armitage, Branch & Zander (2012); Loo, Joseph & Marsono (2016).

Sélection de caractéristiques : Le module de sélection se charge de sélectionner les caractéristiques importantes principalement pour réduire le temps d'apprentissage, mais aussi améliorer les performances en éliminant les caractéristiques peu utiles Huang, Li & Qiang (2016). Une des approches existantes est l'emploi du score d'importance. Ainsi, les caractéristiques avec des scores d'importance inférieure peuvent être supprimées. L'algorithme Forêt aléatoire

peut par exemple être utilisé pour tracer les importances des caractéristiques en exploitant sa structure en arbre.

En effet, bien qu'il ait été largement utilisé dans la classification, grâce à sa stratégie basée sur les arbres, l'algorithme Forêt aléatoire a prouvé son efficacité dans la sélection des caractéristiques puisqu'il les classe automatiquement en fonction de leurs importances, et ce en optimisant au niveau de chaque nœud de l'arbre une mesure appelée : impureté de Gini (Gini impurity). Ce paramètre est mesuré par l'équation ci-dessous, ou pour chaque caractéristique θ, $\Delta_\theta(\tau, T)$ représente la réduction de Gini impureté au nœud de l'arbre .

$$I_G(\theta) = \sum_T \sum_\tau \Delta_\theta(\tau, T) \tag{2.1}$$

Classification basée sur les algorithmes machine learning : C'est le processus d'application d'un algorithme afin de définir la classe à laquelle appartient le flux entrant. Il existe plusieurs catégories de classification, les trois plus importantes sont les suivantes :

- Classification supervisée : Les algorithmes disposent d'une base de données labélisée, ou chaque échantillon est associé à une étiquette. Cette base de données sera employée pour l'apprentissage de l'algorithme et pour la classification de nouveaux échantillons entrants.
- Classification non supervisée : Pour ce type d'approche, l'algorithme ne dispose pas de base de données labélisée. Les flux sont donc classés par regroupement, et cela en employant différents critères dépendamment de l'algorithme de classification déployé. Par exemple : l'algorithme CNN emploie les distances entre les échantillons pour la classification des flux. Figure 2.4
- Classification semi-supervisé : Dans ce cas, l'apprentissage regroupe les deux approches précédentes et se fait donc en employant des échantillons labélisés et non labélisés.

Analyse et validation des résultats : Habituellement, la métrique la plus utilisée pour la validation des résultats de classification est l'exactitude, cependant, en présence de classes

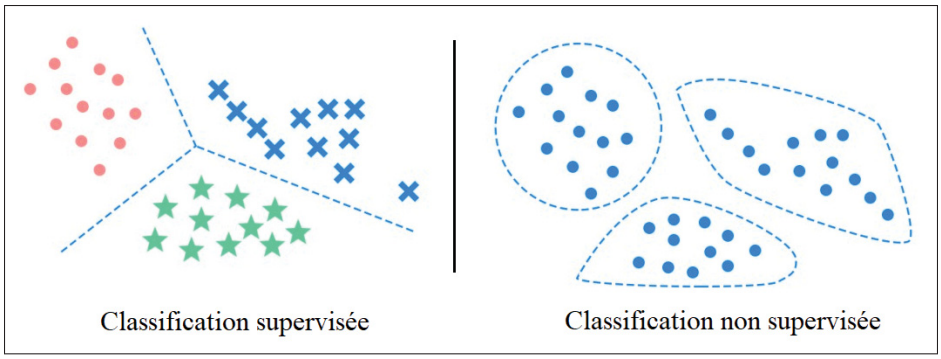


Figure 2.4 Classification supervisée vs Classification non supervisée

déséquilibrées, ce paramètre devient inefficace et d'autres critères de validation doivent être instaurés. Dépendamment du problème de classification, il existe de nombreux critères de validation, plus de détails seront émis dans le chapitre de méthodologie, puisque dans ce chapitre nous nous concentrons sur l'introduction de l'aspect en ligne vs hors-ligne.

2.5.2 Classification en ligne

Lorsqu'il est question de planification du trafic réseau en temps réel, la classification en ligne est nécessaire plutôt que la classification hors ligne, aussi appelée classification tardive. C'est l'étape d'extraction des caractéristiques qui détermine la sensibilité au temps de l'approche de classification et qui définit ainsi l'aspect en ligne ou hors-ligne du mécanisme de planification.

Contrairement à la classification de flux hors ligne, qui extrait les caractéristiques à la fin de la réception de tous les paquets d'un flux, la classification en ligne exécute l'algorithme à partir des premiers paquets seulement et fournit une prévision de flux pendant que le flux est en cours.

Malheureusement, de nombreux algorithmes Liu *et al.* (2016b), ne peuvent pas fournir une classification précoce et construisent ainsi des méthodes hors ligne puisque leurs algorithmes ne garantissent pas une classification efficace avec seulement quelques échantillons. Par conséquent, le problème le plus critique dans la classification du trafic de DC c'est la prise en charge des données en temps réel, avec un trafic fortement déséquilibré. Le challenge vient du fait

que la perte de paquets résultant de l'utilisation de quelques paquets dans le processus de classification en ligne peut affecter les performances des classificateurs en réseau Ding *et al.* (2017) et ainsi entraîner l'inefficacité des ressources en approvisionnement (congestion ou sur approvisionnement).

2.6 Revue de littérature : classification de trafic entre centres de données

Certaines des solutions existantes ne ciblent que la détection des flux éléphants, car ces derniers sont beaucoup plus gourmands en termes de consommation de ressources que les flux souris Zhang *et al.* (2019); Tang *et al.* (2017). D'autres travaux s'intéressent au contraire à la classification des des deux classes Ding *et al.* (2017); Kiran & Chhabra (2019).

Parmi les différentes approches de classification se focalisant sur les flux d'éléphants uniquement, la plus couramment citée est l'échantillonnage. L'échantillonnage ne nécessite pas de statistiques pour le flux au complet, mais fournit plutôt des mesures du réseau en retirant certains paquets pour lesquels des caractéristiques de classification sont mesurées. Un standard d'échantillonnage bien connu est sFlow, qui extrait un paquet à partir de k paquets entrant dans le commutateur. Dans Tang *et al.* (2017) et Zhang *et al.* (2019), les auteurs estiment les périodes d'échantillonnage afin d'éviter des délais supplémentaires. Les résultats montrent que des performances élevées de classification nécessitent un grand nombre de paquets au cours de la période d'émission, ce qui n'est pas efficace pour une identification précoce.

Pour l'identification des deux classes (flux souris et flux éléphants), plusieurs travaux ont exploité la corrélation entre flux qui prenait différentes définitions. Dans Ding *et al.* (2017), les auteurs présentent la corrélation entre flux comme une nouvelle entité nommée Expanding Vector (EV) composée de 7 relations entre un flux entrant et ses voisins. De l'autre cote, les auteurs Kiran & Chhabra (2019) ont exprimé une corrélation au moyen d'agrégation ou cluster formés à l'aide d'une approche semi-supervisée de l'algorithme Gaussian Mixture Model (GMM).

Cependant, au-delà des contraintes de temps dans le premier cas et des erreurs de classification fréquentes des flux de souris dans le second, l'une des limitations communes des travaux présentés

précédemment est l'absence d'un mécanisme pour s'attaquer au problème du déséquilibre des données. En effet, le déséquilibre des données doit être sérieusement considéré étant donné l'impact qu'une approche de classification non efficace/non adapté pourrait avoir sur les stratégies de contrôle de la circulation Gómez *et al.* (2019), entre les coûts élevés de CAPEX et d'OPEX résultant d'un approvisionnement excessif ou encore la congestion et la sous-utilisation des ressources onéreuses.

Nous allons présentés dans ce qui suit quelques articles qui couvrent ce problème en nous concentrons aussi sur l'aspect en ligne ou hors ligne de la méthode proposée.

2.6.1 Travaux existants ciblant la classification de trafic avec des données déséquilibrées

Il convient de mentionner que certains chercheurs ont accordé plus d'attention à cette contrainte (déséquilibre des données) non pas dans le contexte des centres de données, mais plutôt dans différents environnements, par exemple, sur Internet. Les algorithmes de classification développés pour ces réseaux sont toujours applicables dans le contexte des centres de données puisque la seule différence est généralement la dimension du problème de classification. Nous allons présenter les solutions existantes suivant la taxonomie élaborée dans le paragraphe 2.4

2.6.1.1 Niveau données

Dans Rastegarfar *et al.* (2016), les auteurs explorent la classification des flux TCP dans les DC interconnectés par des liaisons optiques. Il utilisent le sous-échantillonnage des caractéristiques extraites des 30 premiers paquets TCP de chaque flux, afin d'atténuer le problème de déséquilibre des données. Cette approche inclut un classificateur par hachage, représentant une mémoire de classification qui réduit la latence ; même si les auteurs réduisent le nombre de paquets impliqués dans le processus d'échantillonnage à 30 paquets, cela reste encore lent pour une classification en ligne. Dans Hasibi *et al.* (2019), les auteurs proposent de traiter le déséquilibre des données par l'augmentation des données (suréchantillonnage). Cette méthode vise à modéliser la distribution de probabilité des caractéristiques afin de pouvoir générer de nouveaux échantillons, de la

même manière que SMOTE. En plus des caractéristiques de niveau flux, ce travail introduit des caractéristiques séquentielles représentant les directions des paquets. L'algorithme de mémoire à court terme (LSTM) est utilisé pour la génération de caractéristiques séquentielles, et pour ce qui concerne les caractéristiques numériques, les auteurs ont appliqué l'estimation de densité du noyau (Kernel Density Estimation KDE). Cependant, les auteurs ne tiennent pas compte du temps requis pour l'augmentation du trafic et extraient les fonctionnalités de 20 paquets de chaque flux.

2.6.1.2 Les solutions algorithmiques

Au niveau algorithmique, Liu *et al.* (2016b) a utilisé SMOTE pour réduire les déséquilibres entre les classes minoritaires et majoritaires, combiné avec une stratégie de type boosting dans le but d'améliorer l'exactitude de la classification en octets (Byte Classification Accuracy : BCA). En plus des taux d'erreur de classification, cette solution calcule pour chaque échantillon un terme de pénalité exprimant la diversité de l'algorithme d'ensemble (faible degré de désaccord entre les classificateurs). Bien que l'algorithme proposé affiche d'excellents résultats pour la classification tardive, il ne parvient pas à maintenir sa stabilité lorsqu'il est utilisé pour l'identification précoce.

Dans un autre article, Gómez *et al.* (2017a), met l'accent sur les algorithmes d'ensemble. Les auteurs ont souligné le fait qu'il n'existe pas d'approche globale claire pour la classification du trafic réseau en temps réel compte tenu du déséquilibre des données et ont construit une nouvelle méthode basée sur une hiérarchie en cascade des classificateurs de l'arbre de décision (DT). Les résultats ont montré que l'approche proposée maintient les mêmes performances de classification par rapport aux autres méthodes d'ensemble en termes de précision, mais surpasse toutes les méthodes d'ensemble et de DT pour les temps de formation et de test prouvant son efficacité pour la classification du trafic réseau en ligne.

Gómez *et al.* (2017a) a contribué à l'élaboration d'une méthode améliorant les temps de calcul et s'approchant ainsi de la classification du trafic en temps réel tout en tenant compte des ensembles de données déséquilibrés. Toutefois, il est important de mentionner que l'établissement de (n-1)

époques d'apprentissage (n étant le nombre d'applications) peut surcharger la mémoire du système, lorsque le nombre des applications est important.

2.6.1.3 Les stratégies sensibles aux coûts

Les auteurs de Chao *et al.* (2016) proposent une stratégie de classification en ligne basée sur l'extraction de flux (stream mining), pour l'identification des souris et des flux éléphants, dans un environnement SDN. L'algorithme proposé est déployé en deux niveaux : 1. MetaCost, constitue la première phase de classification. Il s'agit d'un algorithme de sensible aux coûts fondé sur les arbres de décision. 2. La seconde phase de classification comprend un algorithme d'extraction de flux basé sur l'arbre de Hoeffding qui est implémenté au niveau du contrôleur. Il effectue un apprentissage progressif sur les cinq premiers paquets de chaque flux, ce qui garantit un délai de classification court ; bien que ce travail couvre à la fois le déséquilibre des données et les problèmes de classification en ligne contrairement à la plupart de l'approche de classification du trafic réseau existante, nous pouvons citer quelques limitations. La classification de la phase 2 dépend entièrement des résultats de la phase 1 qui utilise uniquement le numéro de port de destination et le protocole comme caractéristiques de classification, ce qui peut fausser les performances de classification. De plus, le document ne traite pas de l'effet de la première phase de la classification sur l'aspect temps réel de la classification.

Les auteurs dans Gómez *et al.* (2019) ont proposé une étude comparative approfondie entre des approches appartenant à la classe "données", "solutions algorithmiques" ainsi que les "solutions sensibles au coût" avec deux nouveaux algorithmes d'ensemble qui sont la combinaison d'un algorithme d'amplification (boosting) avec une stratégie pour traiter le déséquilibre des données TL-boost et ROSboost (Tomek Links : TL et ROS : Random Over Sampling). Les expériences ont donné plusieurs résultats. En ce qui concerne les stratégies d'échantillonnage, le sous-échantillonnage est plus efficace contre le déséquilibre des données que les stratégies de suréchantillonnage et hybrides. Certains algorithmes d'ensemble se sont avérés plus efficaces, surtout lorsqu'ils sont combinés à des stratégies de sous-échantillonnage, en particulier TL-boost.

Quant à l'algorithme sensible aux coûts, les auteurs ont noté l'importance et la difficulté de produire une matrice efficace des coûts.

2.7 Discussion

Plusieurs des articles présentés font état du déséquilibre des données et de l'importance de la classification du trafic réseau en ligne ; cependant, très peu d'algorithmes peuvent résoudre efficacement les deux problèmes en même temps. En outre, nous remarquons que la plupart des travaux ont choisi de cibler les méthodes de niveau données ou les stratégies algorithmiques, principalement les combinaisons d'ensemble avec l'échantillonnage en raison de la difficulté de construire une matrice de coûts efficace, même si cela pourrait fournir des résultats plus précis Gómez *et al.* (2019).

De plus, nous croyons qu'il est insuffisant de se concentrer uniquement sur la classification pour surmonter ces contraintes, et que l'élaboration d'une approche qui intègre des informations sur la corrélation entre les flux de trafic pourrait améliorer les performances de la classification en caractérisant mieux le déséquilibre des données. Notre approche intègre ces caractéristiques importantes afin de déployer une nouvelle approche de classification du trafic DC basée sur les corrélations entre flux. Plus précisément : Nous proposons une nouvelle approche de classification du trafic en ligne qui résoudra la contrainte du déséquilibre des données au moyen d'une stratégie axée sur les coûts et qui exploitera la corrélation entre les flux afin de remplacer les coûts de classification erronés (matrice des coûts). Plus de détails sur la méthode de calcul des coefficients de corrélation et de l'algorithme de classification dans le chapitre suivant.

CHAPITRE 3

MODULE DE CLASSIFICATION DE TRAFIC ENTRE CENTRES DE DONNÉES

3.1 Introduction

Dans ce chapitre, nous allons étudier les fondements théoriques de la classification de trafic déséquilibré. Nous aborderons ainsi les algorithmes de type ensemble et proposons une approche combinant les corrélations entre les flux dans un concept de solution sensible aux coûts, cela avec une méthode améliorée de l'algorithme RkNN intégrant le bagging. Nous présenterons les BDD sur lesquels sera testée notre approche ainsi que les critères de validation employés.

3.2 Fondements théoriques

Le problème du déséquilibre des données est au centre des préoccupations depuis plusieurs années Del Río *et al.* (2014); Fernández *et al.* (2017). Il attire un nombre important de communautés de recherche dans de nombreux domaines, en particulier les TIs. Plus précisément, avec l'expansion de la virtualisation, des concepts SDN et NFV, les centres de données gagnent du terrain à un rythme exponentiel et en raison de la nature extrêmement variable des données des DC, la classification du trafic devient de plus en plus difficile.

À ces fins, nous construisons une solution de classification basée sur un algorithme ensemble du type bagging, qui appartient à la classe des approches sensible aux coûts. Dans cette solution, nous mesurons les corrélations entre les flux afin de remplacer les poids liés aux erreurs de classification présentes dans une approche du type sensible aux coûts. Nous expliquerons cela plus en détail dans les prochaines sections, mais avant, nous allons passer en revue certaines définitions concernant les algorithmes d'ensemble, plus particulièrement les algorithmes d'ensemble du type bagging.

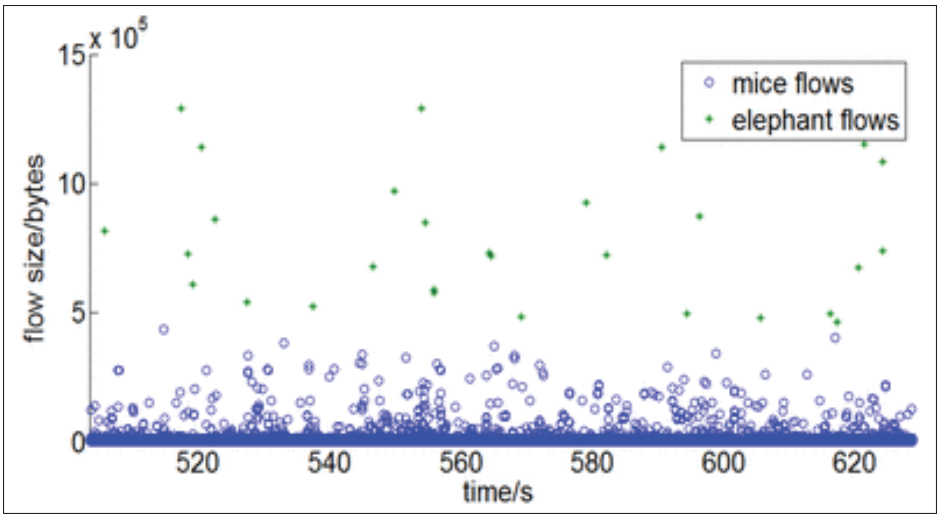


Figure 3.1 Présence de trafic souris Vs éléphant dans les réseaux à large échelle
Tirée de Zhang *et al.* (2019)

3.2.1 Algorithme ensemble, type bagging, boosting

Lors de la classification du trafic, chaque algorithme est entraîné et par la suite ses performances sont testées sur une base de données spécifique. Toutefois, les performances en matière de classification peuvent varier d'une base de données à une autre, ce qui explique pourquoi nous trouvons certains documents qui comparent l'efficacité de leurs algorithmes sur plusieurs ensembles de données. Un algorithme efficace se doit d'avoir un pouvoir de généralité face aux données, dans le cas contraire un changement imminant doit être effectué à chaque changement de données, et dans certains cas, l'algorithme peut même ne pas être applicable. Statistiquement parlant, la généralisation peut être exprimée en termes de biais mesurant la sensibilité du classificateur envers les données d'entraînement ou de variance exprimant la sensibilité envers les ensembles de tests.

Suivant une certaine approche, les algorithmes ensemble construisent des classificateurs plus efficaces à partir d'estimateurs de base dans le but d'améliorer leurs capacités de généralisation Gómez *et al.* (2017b); Sun *et al.* (2007). Les algorithmes d'ensemble visent généralement à réduire la variance et ainsi minimiser le surajustement et augmenter la robustesse du modèle

Gómez *et al.* (2017b). Les 2 approches d'ensemble les plus utilisées sont le bagging (méthode parallèle) et le boosting (méthode séquentielle).

- Bagging est en fait l'abréviation de Bootstrapping Aggregation. Le bootstrapping consiste à échantillonner au hasard la base de données originale (avec ou sans remplacement), afin de construire plusieurs autres populations qui peuvent avoir la même taille que l'ensemble de données original ou non. Ces populations sont appelées sacs. Les modèles sont alors entraînés sur ces sacs et la décision finale est construite en agrégeant, suivant une approche spécifique (peut-être le vote, la pondération. . .), les résultats des différents algorithmes. Cette approche est dite parallèle, car les algorithmes sont appliqués simultanément aux différents 'sacs' pour ensuite obtenir le résultat final de la classification par agrégation.
- Boosting, d'autre part est un algorithme d'ensemble qui entraîne séquentiellement un nouveau modèle selon les résultats du précédent. Bien que la principale motivation derrière cette approche soit de créer des algorithmes plus forts à partir des faibles, des erreurs de classification peuvent se produire chez les algorithmes de bas niveau entraînant une dégradation des performances de l'algorithme final.

Dans notre approche de classification du trafic déséquilibré échangé entre centres de données et caractérisé par des besoins différents en temps, nous avons décidé de déployer des sacs de forêts aléatoires (Random Forest, RF), c'est-à-dire de combiner les RF avec l'algorithme ensemble de type bagging, car ils ont tous deux montré leurs preuves pour traiter avec succès la réduction de la variance Sun *et al.* (2007). De plus, RF est considéré comme l'un des algorithmes les plus appropriés pour la classification du trafic en ligne Gómez *et al.* (2017b) et le bagging pour remédier à la contrainte du déséquilibre de trafic Collell *et al.* (2018) .

La forêt aléatoire (Random forest RF) est un algorithme d'ensemble de types bagging avec pour estimateur de base des arbres de décision (Decision Trees DT) de type CART. À l'origine, le vote était utilisé pour l'agrégation des prédictions des DT. Cependant, la littérature présente plusieurs approches pour améliorer les performances de ce classificateur. Certains se sont concentrés sur la transformation du vote majoritaire en un système plus efficace tel que la pondération et

donc directement amélioré les performances du classificateur global. D'autres méthodes visaient à améliorer la performance de chaque arbre en examinant par exemple la mesure de division utilisée lors de la construction de l'arbre (ration de gain, indice de gini, gain d'information, chi square, ReliefF. . .) Kulkarni, Sinha & Petare (2016). Dans notre approche proposée, nous utilisons l'estimation de l'erreur en dehors du sac (Out-Of-Bag error (OOB)) dans l'agrégation des arbres au lieu du vote classique. OOB représente l'erreur résultant des échantillons exclus lors de l'échantillonnage des ensembles de données pour l'étape Boostrap.

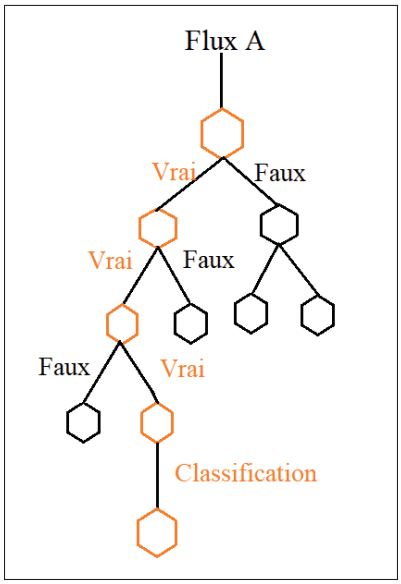


Figure 3.2 Démonstration du concept des forêts aléatoires

3.3 Solution proposée au problème de classification en ligne des données non balancées échangées entre les centres de données

Dans cette section, nous présentons une nouvelle approche pour la classification en ligne du trafic déséquilibré des centres de données, combinant la puissance des algorithmes d'ensemble de type bagging avec l'efficacité de l'utilisation des corrélations entre les flux entrants dans la matrice des coûts. Selon notre connaissance, il n'existe pas de telle combinaison pour surmonter les défis de l'aspect déséquilibre du trafic tout en assurant l'identification en ligne des flux d'éléphants et de souris. Cette approche exploite la corrélation entre les flux de test et les flux

d'entraînement au niveau de chaque sac (pour les labels rares, ce poids est augmenté par le rapport de déséquilibre mesurant la variabilité entre les classes majoritaires et minoritaires). Ces coefficients vont principalement agir comme un poids de rééquilibrage pour le trafic des centres de données occupé par les flux de souris. La mesure de la corrélation est calculée à l'aide d'un algorithme d'apprentissage automatique modifié de l'algorithme du k plus proche voisin . Avant de présenter notre système de classification au complet, nous allons en premier lieu présenter l'algorithme de base pour calculer les coefficients : (Reverse k Nearest Neighbour (RkNN).

3.3.1 Fondements conceptuels de l'algorithme RkNN

En raison de ses nombreuses applications, l'algorithme RkNN est devenu un outil de traitement spatial de données essentiel. Il trouve dans une base de données, tous les points pour lesquels un point de requête fait partie de leurs ensembles de voisins les k plus proches. Particulièrement, RNN trouve les points dans une base de données pour lesquels le point de requête est leur voisin le plus proche Angiulli (2017). Si la requête est du même type que les données, RkNN est dit monochromatique ; sinon, il est bichromatique Yan, Zhao & Ng (2012). Dans ce travail, nous nous concentrons sur un RkNN monochromatique puisque nous cherchons à trouver les RkNNs pour les flux de test parmi les sacs de flux d'apprentissage.

En détail, nous dénotons l'ensemble de données multidimensionnelles $S = s$, où s est un objet de données représenté par m caractéristiques ; trouver les RkNNs d'un point requête $q \notin S$ consiste à trouver tous les points $s \in S$ pour lesquels cette condition est assurée :

$$RkNN(q) = \{s \in S | d(s, q) < d(s, s^{'})\}, \tag{3.1}$$

où s' est l'un des k voisins les plus proches, mais le plus éloignés de s et $d(a, b)$ est une métrique de distance (par exemple la distance euclidienne Tao, Papadias, Lian & Xiao (2007)).

Il est important de mentionner que la relation RNN et NN n'est pas symétrique. En d'autres termes, $s \in kNN(q)$ n'implique pas automatiquement $s \in RkNN(q)$, où $RkNN(q)$ représente

les k voisins les plus proches inverses de q et $kNN(q)$ ses k voisins les plus proches Angiulli (2017); Tao *et al.* (2007).

Résoudre le problème RkNN peut être réalisé en recherchant les points pour lesquels la requête q est l'un de leurs voisins les plus proches Angiulli (2017). Ceci consiste à déterminer les voisins les plus proches de chaque point de l'ensemble de données et de les parcourir par la ensuite un à un afin de retrouver q. Le problème est qu'avec des ensembles de données croissants, il augmenterait drastiquement la consommation en termes de ressources, mais aussi de temps. Une autre approche possible, cartographie les points dans un espace et emploie la méthode de recherche, séparation et évaluation (branch and bound search). En utilisant cette méthode, nous comparons le point de requête à chaque branche d'une structure arborescente et supprimons les branches qui ne correspondent pas aux critères de recherche Dawar, Goyal & Bera (2015).

Pour expliquer plus clairement l'algorithme RkNN, nous avons choisi une valeur fixe pour k (par exemple $k = 3$). Comme mentionné précédemment, la première étape consiste à calculer les NNs (p) pour chaque point de données ($s \in S$). Par la suite, un cercle d'un rayon calculé comme la distance entre s et NN(s) est dessiné (cercle en noir sur la figure ci-dessous : c_3, c_4, c_5 et c_6), et son rectangle limite minimum (Minimum Bounding Rectangle MBR) est représenté. Par exemple, les points : p_1, p_2 et p_3 forment une MBR et sont représentés par le cercle c_3 ; de même pour les points p_{10} et p_{11} représentés eux aussi par une MBR et par le cercle c_1. De la même manière, le cercle c_2 est construit à partir des autres points. Ainsi, les MBR sont indexés par un arbre de type R en fonction des distances entre les échantillons. Chaque feuille est un point de données, tandis qu'une branche (nœud intermédiaire) est le MBR de son nœud enfant. Une fois que nous avons construit l'arbre, les RkNN d'une requête peuvent être obtenus en mesurant la distance entre le MBR de chaque branche et la requête et en parcourant les différentes branches de l'arbre Tao *et al.* (2007).

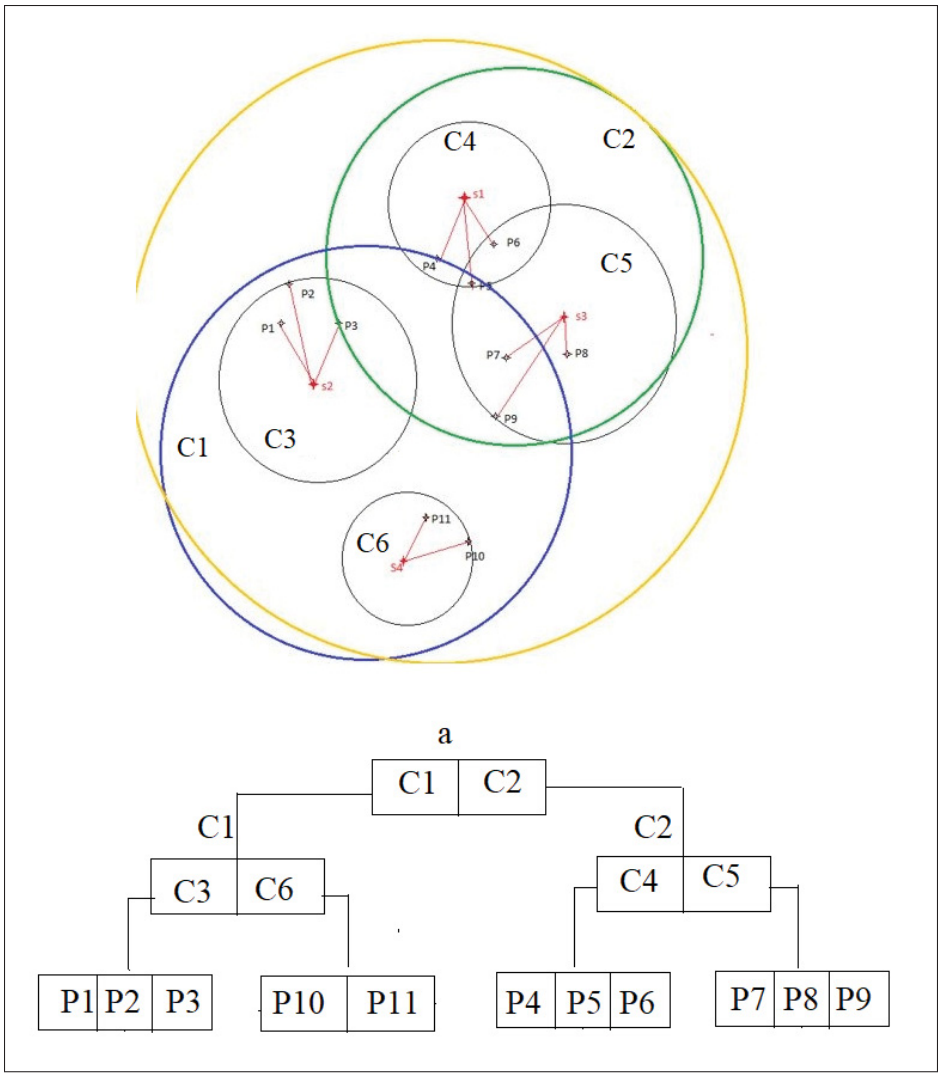


Figure 3.3 Concept général de l'algorithme RkNN

3.3.2 Approche proposée : Description du système

L'utilisation d'une approche de classification sensible aux coûts pour résoudre le problème du déséquilibre des données nécessite une matrice des coûts afin de coder les erreurs de classification et de construire la matrice des poids de classification erronée. Dans notre approche, les coûts de la matrice des coûts sont représentés par les corrélations entre les échantillons de test et le modèle d'ensemble (corrélations entre flux). Notre méthode comprend plusieurs modules présentés à la figure 3.4.

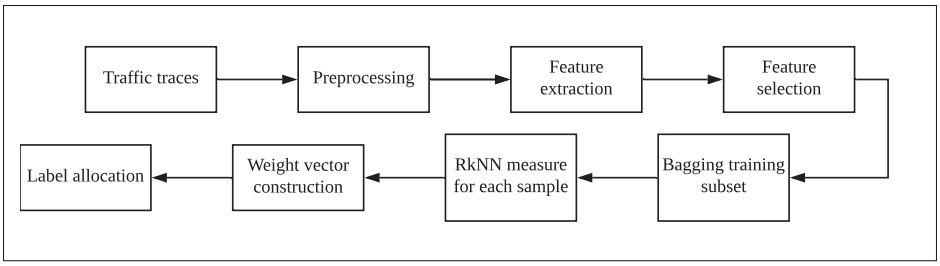


Figure 3.4 Modèle d'un système de classification

- Module de prétraitement : Cette étape consiste à supprimer les échantillons non significatifs, incomplets ou inutiles (par exemple, les paquets vides). Pour les bases de données que nous allons employer, nous allons par exemple nous concentrer principalement sur les flux TCP, car les paquets UDP n'ont pas une contribution significative au trafic total, ils seront donc filtrés par ce module.
- Extraction des caractéristiques : Comme indiqué dans Amina, Abdolkhalegh, Khoa & Mohamed (2018), les meilleures performances de classification ont été observées avec des flux de 8 paquets. Nous commençons donc par construire des flux à partir de paquets avec la même adresse IP source et IP de destination ainsi que des numéros de port, puis nous construisons la matrice des caractéristiques à partir de 26 fonctionnalités (niveau de paquet et fonctionnalités de niveau de flux), y compris l'intervalle de temps entre les arrivées, la longueur des paquets, les statistiques de longueur et autres informations de synchronisation TCP, telles que le nombre d'accusés de réception, de paquets d'envois, de paquets de synchronisation, de retransmissions et d'accusés de réception sélectifs, comme dans Amina *et al.* (2018).
- Sélection de caractéristiques : Nous construisons notre matrice de caractéristiques finale après avoir appliqué RF pour la sélection des fonctionnalités significatives. Des recherches antérieures ont démontré les limites de la méthode RkNN dans une espace de dimension importante Angiulli (2017); Casanova *et al.* (2017); Tao *et al.* (2007). Cette étape est donc très importante, car elle réduit la dimension de l'espace pour la génération de candidats RkNN. De plus, en plus de réduire le temps d'entraînement, le rejet de caractéristiques inutiles permet d'économiser sur des ressources mémoires limitées.

- Bagging des échantillons d'entraînement : Dans cette étape, nous construisons notre algorithme d'ensemble. Nous commençons par échantillonner au hasard et avec remplacement le sous-ensemble d'entraînement pour construire plusieurs populations, à savoir les sacs de l'algorithme bagging (généralement, ces sacs peuvent être de la même taille que l'ensemble d'entraînement ou pas). Plus tard, nous capturerons les performances de notre algorithme en fonction du nombre de sacs déployés afin de tester la variabilité de la base de données et la possibilité de représenter efficacement les données déséquilibrées en coefficients de corrélation. La décision finale de classification est prise selon une approche pondérée des résultats des algorithmes au niveau de chaque sac, où les poids sont les coefficients de corrélations.
- RkNN : Au cours de cette étape, nous mesurons la corrélation entre chaque échantillon de test et les sacs de l'algorithme bagging construits lors de l'étape précédente. Cette corrélation aide à répondre à la question suivante : dans quelle mesure ce flux est-il proche de la prédiction d'un sac ? Les coefficients de corrélation sont, en fait, le nombre de RkNN d'un échantillon de test. Ainsi, pour chaque échantillon de test X, nous mesurons Rk la corrélation avec le sac i. Nous allons présenter plus en détail ce module dans la section suivante.
- Construction du vecteur poids : À partir du résultat des algorithmes RkNN ou, en d'autres termes, des coefficients de corrélation entre flux, nous construisons la matrice des poids comme suit :

$$W_i = \begin{cases} Rk(X,i), if\ l(X) = 0 \\ Rk(X,i) * R, \text{if l(X)=1,} \end{cases} \tag{3.2}$$

où R est le rapport de déséquilibre calculé comme le rapport entre le nombre de flux majoritaire par flux minoritaire (dans le cas de labels rares, la corrélation est augmentée par le rapport de déséquilibre mesurant la variabilité entre les classes majoritaires et minoritaires comme le montre l'équation ci-dessous), et l le label de classification obtenue avec le sac i. Enfin, nous construisons le vecteur de poids de classification :

$$W = \{w_i\}, \forall i \in [1, 2, ..., m], \tag{3.3}$$

où m est le nombre de sacs.

- Allocation des labels de classification : Dépendamment de l'étiquette obtenue par chaque sac et de la distance entre l'échantillon d'essai et le sac (les coefficients de corrélation), nous choisissons l'étiquette w_i associé à la somme maximale de RkNNs, comme suit.

$$w_i = \begin{cases} 0, if \sum_i w_{i|l=0} > \sum_i w_{i|l=1} \\ 1, otherwise, \end{cases} \tag{3.4}$$

où $\Sigma_i w_{i|l=0}$ représente la somme du nombre de RkNN entre l'échantillon d'essai X et tous les sacs de classification pour lesquels l'étiquette obtenue est égale à 0 et $\Sigma_i w_{i|l=1}$ est la somme du nombre de RkNNs entre l'échantillon d'essai X et tous les sacs pour lesquels le label obtenu est égal à 1.

La totalité du système est résumée par la figure suivante.

3.3.2.1 Algorithme proposé pour le calcul des coefficients de corrélation basé sur le RkNN

Dans cette section, nous présentons l'approche utilisée pour calculer le nombre de RkNN de chaque flux de test. Notre algorithme est une version modifiée du kNN inverse ; nous choisissons cet algorithme, car il est connu pour être efficace contre le déséquilibre des données Wu *et al.* (2015a).

Dans cette section, nous présentons l'approche utilisée pour calculer le nombre de RkNN de chaque flux de test. Les auteurs de Angiulli (2017) ont affirmé que lorsque les techniques requête de spectre (Range Query techniques) basé sur le concept des MBRs sont déployées telqu'expliqué dans la section 3.1.1, la recherche des points de données spécifiques peut être coûteuse en termes de ressource, mais aussi en temps. Cependant, notre approche est basée sur la technique requête de rayon (radius query technique), ou les calculs de distance sont plus rapides et moins onéreux.

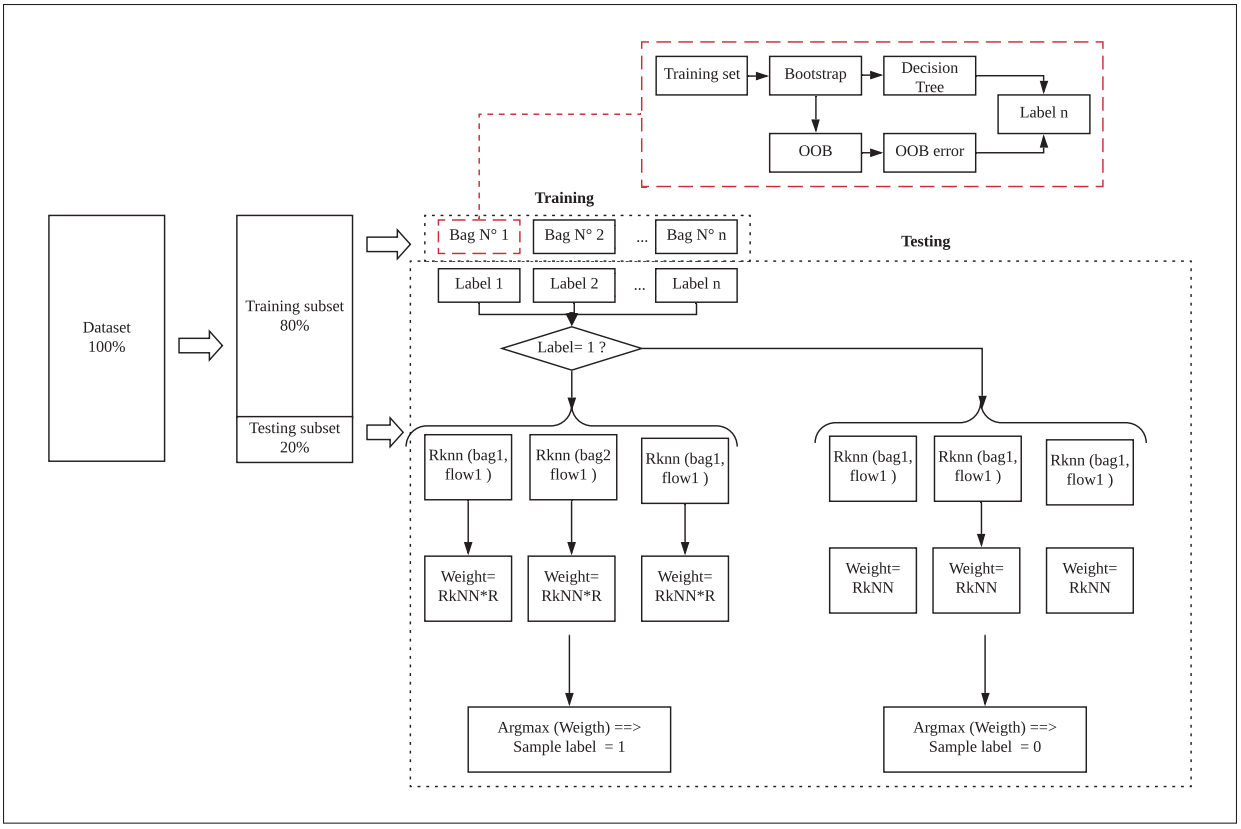


Figure 3.5 Approche de classification proposée, basée sur la corrélation entre échantillons

Ainsi, comparativement à la méthode classique déployant des arbres de type R (R tree), nous employons dans notre approche, une autre structure d'indexation célèbre, l'arbre K-d (Kd tree) tree Barewar, Radke & Deshpande (2014), qui est similaire à R-tree mais plus facile à déployer et plus efficace en termes de temps de construction. La seule différence entre les arbres R et Kd est que si le premier utilise les MBR pour regrouper les points de données, le second calcule récursivement les points médians et les divise en deux moitiés autour d'eux, réduisant ainsi le temps de construction.

De plus, la dimensionnalité du vecteur de caractéristiques doit être considérée puisqu'une dimensionnalité plus élevée entraîne une capacité de sélection moins efficace. Ceci est causé principalement par le chevauchement élevé entre les cercles, qui peut pousser l'algorithme à ne pas les choisir. Notre solution à cette contrainte est d'utiliser la sélection des caractéristiques,

car elle représente un moyen efficace pour réduire la dimensionnalité des bases de données en supprimant uniquement les caractéristiques non significatives.

L'algorithme ci-dessous décrit l'approche proposée. Nous mesurons la corrélation entre les flux entrants et le modèle d'entraînement sur plusieurs sacs à travers deux sous-fonctions (représentant à la fois les stratégies d'entraînement et de test de l'algorithme d'apprentissage automatique).

Dans la première partie de notre algorithme : apprentissage (étapes 3 à 11), nous construisons un arbre K-d en utilisant des échantillons d'entraînement de chaque sac s_{ij}. Au cours de cette étape, nous commençons par calculer pour chaque point de données s_{ij} (à partir du sac i), ses kNN afin de concentrer plus tard la recherche de RkNNs dans le sous-ensemble de k points de données se rapprochant le plus de s_{ij}. Par exemple, pour le point s_2, ses NNs sont : p_1, p_2 et p_3. Cette opération de filtrage (étapes 5 à 8) est suivie de la construction de la structure d'index (étapes 9 à 10). Nous dessinons des cercles avec le rayon correspondant à la distance entre l'échantillon et son voisin le plus éloigné afin d'inclure tous les kNN calculés. Ainsi, comme le montre la figure 3.3, les points p_1, p_2 et p_3, voisins de s_2 sont représentés par le cercle c_3. De même pour les autres points restants. Après cela, nous groupons les cercles deux par deux jusqu'à atteindre la racine de l'arbre ; cela dépend de la distance entre les centres des cercles. En d'autres termes, les cercles c_3 et c_6 sont regroupés par le cercle c_1, et les cercles c_4 et c_5 par le cercle c_2. Nous calculons ainsi le voisin le plus proche du centre d'un cercle, et si le cercle de ce voisin le plus proche n'a pas déjà été groupé à un autre cercle, il est choisi. Sinon, nous choisissons le voisin le plus proche.

Pour la phase de test, en considérant un ensemble de requêtes $Q = q$ représentant des échantillons de test (flux) dans un espace multidimensionnel (caractéristiques), en suivant la deuxième sous-fonction de notre algorithme (étapes 13 à 26), nous parcourons l'arbre K-d construit (de la racine aux feuilles), afin de construire RkNN, un vecteur contenant le nombre de RkNN de chaque échantillon. La procédure commence par mesurer la distance entre la requête et le centre de chaque cercle (représentant une branche ou une feuille dans l'arbre). Nous comparons cette

Algorithme 3.1 Algorithme de mesure de la corrélation entre flux basé sur la méthode RkNN

```
1  Algorithme : Approche de mesure de la corrélation entre flux basée sur la méthode
     RkNN
   Input : k, Q = {q}, all = {S_i}, S_i = {s_i,1, s_i,2, ..., s_i,j}
   Output : R_k
2  Phase d'apprentissage : Construire les arbres de type K-d pour chaque sac
3  repeat
4      foreach echantillon s in S do
5          Mesurer R_k{s}
6          Tracer un cercle avec le rayon égal à la distance avec le NN (Nearest Neighnor :
             voisin le plus proche) le plus loin
7      end foreach
8      Grouper les cercles deux par deux afin de construire les branches del'arbre.
9      Répéter jusqu'au top de l'arbre
10 until all est vide;
11 Phase de test : Calculer le RkNN pour chaque échantillon de test dans chaque sac
12 repeat
13     foreach cercle (C) composant l'arbre du type K-d do
14         if d(q,center(C : une branche ou la feuille))<
15         rayon C then
16             if C est une feuille then
17                 R_k = R_k + 1
18             end if
19             else
20                 aller aux sous branches de cette branche
21                 Aller à la ligne 14
22             end if
23         end if
24     end foreach
25     Return R_k
26 until All est vide est ce pour chaque requête q dans Q;
```

distance au rayon du cercle afin de choisir les branches respectant la condition de l'étape 15. Une fois que nous arrivons au bas de l'arbre, nous incrémentons le numéro RkNN de l'échantillon concerné par 1 (étapes 17-18). Sinon, nous continuons à parcourir l'arbre (étapes 20 et 21).

Nous avons testé les performances de notre approche sur plusieurs bases de donnés et défini aussi plusieurs matrices de validations que nous discuterons dans le chapitre d'analyse de résultats.

3.4 Expérimentation

Afin de valider notre approche de classification, nous avons mené des tests sur plusieurs bases de données. Un ensemble de centres de données infonuagiques (CAIDA CAIDA (2011)), deux bases de données de centres universitaires (Wisconsin Madison, UNI1 et UNI2 bases de données Benson (2010))) et un ensemble de données de trafic Internet (UNIBs UNIBS (2009)).

3.4.1 Bases de données

Le site CAIDA contient une série de bases de données capturées dans différentes directions, en liaison montante (direction A) ou descendante (direction B) à différentes périodes ; nous avons utilisé les traces de trafic produites le 17 février. 2011, et dans la direction A représenté par la figure 3.6. Nous avons décidé d'utiliser ces traces de trafic de CAIDA car elle a été récemment utilisée dans une approche de prévision du trafic en temps réel par Iqbal, Zahid, Habib & John (2019). Aussi, nous choisissons la direction A au lieu de la direction B car la dernière contient un pourcentage important de perte de paquets selon CAIDA (2011). Les bases de données sont très déséquilibrées CAIDA (2011); Benson (2010), et contiennent certains paquets TCP et UDP inconnus, ce qui les rend plus appropriés pour la classification souris/éléphant plutôt que l'identification de l'application.

Pour UNI1 et UNI2 Benson, Akella & Maltz (2010), les traces de trafic ont été collectées sur un site universitaire dans l'ouest des États-Unis pendant 12 heures sur plusieurs jours. Après avoir filtré les paquets non significatifs, nous avons combiné plusieurs fichiers de traces de trafic dans l'ordre chronologique pour construire un fichier avec suffisamment d'instances.

UNIBs, est par contre une base de données de trafic composé de 99% de paquets TCP échangés entre plusieurs machines connectées à Internet via un routeur périphérique et une liaison à haute capacité.

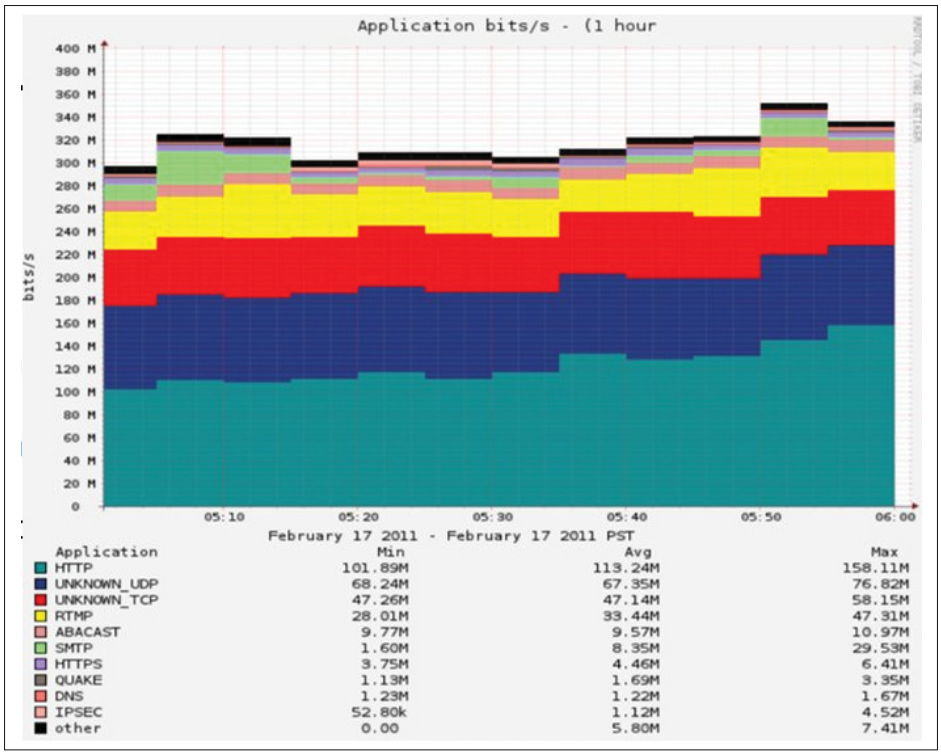


Figure 3.6 Base de données CAIDA
Tirée de CAIDA (2011)

Chacune des bases de données précédemment citées est représentée par une matrice de caractéristiques, leurs tailles respectives sont les suivantes : La base de données de Caida forme une matrice de taille (20000x26), UNI1 une matrice de (21013x26), UNI2 une matrice de (2586x26) et UNIBs une matrice de (19349x26), où les lignes représentent des paquets de trafic regroupés en flux par 4 paramètres : adresses et numéros de port source et destination, le type de protocole n'est pas inclus puisque nous avons appliqué un filtre à l'étape de prétraitement pour sélectionner uniquement les paquets TCP), les colonnes quant à elle représentent le vecteur de caractéristiques tel que décrit dans Amina *et al.* (2018).

Il est important de mentionner que puisque nous n'avons pas été en mesure de trouver une base de données préalablement labélisée de flux de souris et éléphants (ground truth), nous avons construit notre propre ground truth en posant cette hypothèse : si le nombre de paquets dans le flux dépasse 100 paquets, c'est un flux éléphant ; autrement, c'est un flux de souris. Notre

hypothèse est basée sur le fait que si la taille de trame Ethernet est limitée à 1500 octets (trames géantes exclues), 100 trames génèrent 1,2 Mbit de données. Nous supposons que cette taille pourrait représenter un flux d'éléphant Benson *et al.* (2010). Cependant, comme mentionné dans Benson *et al.* (2010) pour l'ensemble de données UNI2 et comme observé à partir de nos expériences pour l'ensemble de données UNIBs, les flux sont courts et détiennent donc un plus petit nombre de paquets. Ainsi, selon les observations faites à partir de Benson *et al.* (2010) et de nos propres expériences, nous avons fixé le seuil à 15 paquets pour UNI2 et 30 paquets pour UNIBs.

3.4.2 Évaluation des performances

Dans la présente section, nous testons les performances de notre approche en faisant varier ses hyperparamètres : le nombre et la taille des sacs (3, 5 et 10 de 20, 40, 60 et 80% du set d'entraînement) et l'ordre RkNN (k=2, 5, 10 et 20). Nous comparons également les meilleures performances de classification que nous obtenons avec notre algorithme fondé sur la corrélation, à 16 algorithmes existants pour la classification du trafic déséquilibré, allant du sous-échantillonnage, suréchantillonnage, des approches hybrides, des stratégies algorithmiques aux méthodes sensibles aux coûts.

3.4.2.1 Métriques d'évaluation de performance

3.4.2.1.1 Mesure F1, précision et rappel

Alors que l'exactitude (accuracy) est habituellement la métrique de validation des performances d'un algorithme de classification ; avec un ensemble de données très déséquilibré, cette mesure biaise l'efficacité de tout algorithme, puisqu'elle se concentre principalement sur les classes majoritaires. Par conséquent, afin de valider efficacement notre approche proposée, nous avons utilisé plusieurs autres métriques de performance : la précision, le rappel, la mesure-F1, la matrice de confusion et le temps d'exécution. Bien que la mesure-F1 (combinaison de la précision et du rappel) soit la mesure la plus couramment utilisée avec des classes déséquilibrées,

nous cherchons à trouver un bon équilibre entre les cinq mesures. En effet, comme le montrent les différents scénarios couverts pour ce test de performance, dans certains cas, nous pouvons trouver une mesure F1 plus élevée, qui est principalement obtenue à partir d'une plus grande précision ou d'un rappel plus élevé, et non d'un équilibre entre les deux.

Il est important de mentionner que les graphiques que nous présentons sont liés à la classe minoritaire (flux d'éléphants) et que la classe majoritaire obtient entre 98% et 99% de précision, de rappel et de mesure F1 dans la plupart des scénarios. Ainsi, comme notre algorithme est aussi efficace que les autres pour la classe majoritaire, nous avons décidé de nous concentrer sur les échantillons rares dans cette analyse.

La mesure-F1 ($F1$) est calculée à l'aide de l'équation suivante :

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3.5}$$

La précision (precision) exprime la pertinence des résultats, par exemple, sur 100 échantillons, combien ont été correctement classés dans la classe 1 ou 2 (vrais positifs, TP) parmi le nombre total d'échantillons prédits comme positifs (vrais positifs + faux positifs, TP+FP), ce qui équivaut à l'équation suivante :

$$Precision = \frac{TP}{TP + FP} \tag{3.6}$$

Le rappel (recall), d'autre part, exprime la probabilité de détection, en d'autres termes, le rapport entre les prédictions positives (TP) et le total des échantillons positifs (vrais positifs + faux négatifs, TP+FN),

$$Recall = \frac{TP}{TP + FN} \tag{3.7}$$

La haute précision et le faible rappel sont équivalents à un FN élevé et à un FP faible (un nombre important de flux de souris sont classés comme des flux d'éléphants), tandis qu'un rappel élevé

et une faible précision signifient un FP élevé et un FN faible (un nombre important de flux d'éléphants sont classés comme des souris). En suivant notre stratégie d'équilibre entre précision et rappel, nous essayons d'éviter la congestion et le surapprovisionnement en même temps, tout en maintenant un temps de formation efficace pour les contraintes de temps.

3.4.2.1.2 AUC et Cohen Kappa

Lorsque nous comparons les résultats obtenus par notre approche à d'autres travaux connexes, nous utilisons deux autres mesures avec la précision, le rappel, la mesure F1 et le temps de fonctionnement ; il s'agit de la métrique Cohen Kappa ainsi que la circonférence sous la courbe des caractéristiques opérationnelles du récepteur (Area Under the Receiver Operating Characteristic (ROC)) : AUC, figure 3.7 Martinez-Ríos *et al.* (2021).

La mesure AUC est obtenue à partir de la courbe ROC qui trace le rapport entre le taux des vrais positif (True Positive Rate :TPR) et le taux des faux positives (Fause Positive Rate :FPR) afin de représenter les performances du modèle. Le TPR et le FPR sont calculés par l'évaluation d'un modèle de régression logistique pour différents seuils. Parce qu'il ne nécessite aucune information sur les distributions de trafic, AUC est fortement utilisé pour l'évaluation des performances en présence de données déséquilibrées Zhang, Lu, Qassrawi, Zhang & Yu (2012). AUC calcule la surface sous la courbe entre les valeurs TPR et FPR à la fois égales à 0 et 1 (0,0) et (1,1) en utilisant le calcul intégral. Plus la valeur d'AUC est élevée, plus la classe est séparable et donc meilleure est l'efficacité de l'algorithme Doroud *et al.* (2018).

Quant à la métrique Kappa Cohen, elle mesure l'accord entre deux algorithmes également appelés évaluateurs, lorsqu'ils sont appliqués au même ensemble de données. L'objectif est de mesurer la fiabilité des évaluateurs observée. Habituellement, les valeurs de Kappa varient entre 0 et 1, où 0 représente aucun accord entre les classificateurs ou un accord par hasard, et 1 est le synonyme d'un accord idéal. Pour calculer kappa (k), nous construisons une matrice incluant les probabilités que les deux algorithmes obtiennent les mêmes labels (P_{11}, P_{22}) et la probabilité qu'ils ne soient pas d'accord sur l'attribution des étiquettes (P_{12}, P_{21}). $\left[\begin{smallmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{smallmatrix}\right]$

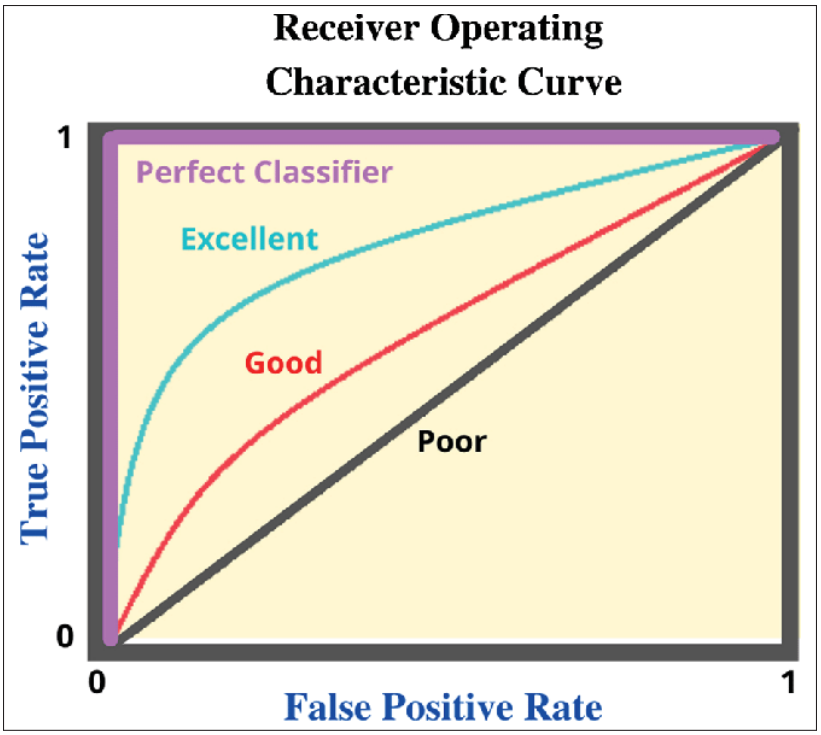


Figure 3.7 Courbe AUC
Tirée de Martinez-Ríos *et al.* (2021)

Nous mesurons alors $_{P0}$ à partir de P_{11} et P_{22} exprimant le niveau d'accord et Pe à partir de P_{12} et P_{21} exprimant l'attribution aléatoire d'étiquettes à partir des algorithmes. Enfin, la valeur Kappa est obtenue à partir de P_0 et P_e suivant cette équation McHugh (2012).

$$\kappa = \frac{P_0 - P_e}{1 - P_e} \tag{3.8}$$

3.4.2.2 Analyse des résultats

Notre algorithme est basé sur le calcul des corrélations entre chaque flux de test et les échantillons d'entraînement composant chaque sac de l'ensemble bagging. Ainsi, il est important de paramétrer efficacement l'algorithme en termes de nombre de sacs et de leurs tailles respectives. Sur cette base, nous avons établi une comparaison entre plusieurs scénarios pour 3, 5, 10 et 20 sacs en

utilisant respectivement 20%, 40%, 60% et 80% de l'ensemble des échantillons d'entraînement dans chaque sac. Parmi tous les scénarios testés que nous présentons dans cette section, nous choisissons pour chaque base de données, celui qui présente les meilleures performances de classification afin de le comparer à d'autres algorithmes de classification déséquilibrés. Pour la première base de données (CAIDA), nous analyserons minutieusement les résultats et présenterons l'approche qui mène au choix d'un seul scénario optimal. Pour le reste des bases de données, pour plus de simplification, nous allons directement présenter le meilleur scénario. Étant donné que le temps d'exécution pour les scénarios de déploiement de 20 sacs de toute taille confondue et pour n'importe quelle base de données est très élevé et ne convient donc pas aux contraintes en temps réel, nous avons décidé de ne pas inclure leurs résultats dans l'analyse des résultats expérimentaux suivante.

3.4.2.2.1 Sélection du nombre de sacs et de leurs tailles

- CAIDA :

  Pour la base de données de CAIDA, nous remarquons qu'à l'exception du scénario de déploiement de 10 sacs de 20% de l'ensemble d'entraînement, l'augmentation du nombre de sacs et de leur densité améliore les performances de classification en termes de précision, de rappel, de mesure F1 et d'AUC. Cependant, comme les temps d'entraînement et de test augmentent de façon exponentielle en fonction du nombre et taille des sacs (3.8 pour le temps d'entraînement), nous devons trouver le meilleur compromis en nous aidons de la matrice de confusion afin de sélectionner le scénario optimal. On sélectionne à chaque fois le meilleur scénario parmi les quatre représentant le même nombre de sacs et différentes tailles, puis on compare ces 4 scénarios (un avec le nombre de sacs=3, un autre avec 5 et puis 10) et on sélectionne le meilleur compromis en termes de mesures de classification (précision, rappel, mesure F1, ainsi que la matrice de confusion et AUC) et de consommation de temps (temps de fonctionnement).

  Dans la première étape, nous sélectionnons (figure 3.10) le scénario déployant 20% de l'ensemble d'entraînement avec 3 sacs comme le meilleur scénario. Ce scénario correspondant à 65% de précision, 61% de rappel, 63% de mesure F1, 80% d'ASC et de matrice de confusion

$\begin{bmatrix} 3807 & 48 \\ 56 & 89 \end{bmatrix}$ pour 22 secondes d'entraînement ; Avec 5 sacs, le scénario optimal correspondant à une précision de 65%, un rappel de 61%, une mesure F1 de 63%, un AUC de 80% et une matrice de confusion $\begin{bmatrix} 3808 & 47 \\ 56 & 89 \end{bmatrix}$, est obtenu avec 20% de l'ensemble d'entraînement dans chaque sac, pendant 36 secondes d'entraînement. Ensuite, avec 59% de précision, 72% de rappel, 65% de mesure F1, 85% d'AUC et une matrice de confusion $\begin{bmatrix} 3781 & 74 \\ 40 & 105 \end{bmatrix}$, le meilleur compromis est obtenu avec 40% d'entraînement dans chacun des 10 sacs en 147 secondes d'entraînement.

Enfin, pendant 22 secondes d'entraînement, le scénario utilisant trois sacs de 20% de l'ensemble d'entraînement original représente le compromis optimal avec un ordre RkNN égal à 10.

- UNI1 :

  Pour la seconde base de données (UNI1), lors de la recherche d'un scénario avec matrice de confusion équilibrée (nombre de FN proche de FP) afin d'éviter le surapprovisionnement et la congestion dégradant les performances QoS, et tout en considérant le compromis entre la précision, le rappel, la mesure F1 et l'AUC, nous trouvons que le meilleur scénario est obtenu avec 5 sacs de 40% de l'ensemble d'entraînement. Les résultats comprennent 55% de précision, 58% de rappel, 57% de mesure F1, près de 80% d'AUC et une matrice de confusion $\begin{bmatrix} 3847 & 114 \\ 101 & 140 \end{bmatrix}$ pour 99 secondes d'entraînement.

- UNI2 :

  Contrairement aux autres bases de données, le meilleur scénario pour UNI2 est obtenu en 3,24 secondes avec 3 sacs de 60% de l'ensemble d'entraînement dans chaque sac. 82% de précision, 100% de rappel, 90% de mesure F1, 99% d'AUC et une matrice de confusion $\begin{bmatrix} 478 & 7 \\ 0 & 33 \end{bmatrix}$ résultent de cette base de données. Il convient de mentionner que UNI2 est également plus courte que les autres bases, ce qui peut affecter les performances de classification.

- UNIBS :

  Pour l'ensemble de données UNIBS, nous remarquons dans nos expériences qu'il est également caractérisé par de petits flux, nous ajustons le seuil et le fixons à 30 paquets par flux et obtenons les résultats optimaux avec 5 sacs de 20% de l'ensemble d'entraînement. Les performances de classification incluent une précision de 65%, un rappel de 78%, une

mesure F1 de 71%, près de 80% d'AUC et une matrice de confusion égale à $\left[\begin{smallmatrix} 2119 & 521 \\ 270 & 960 \end{smallmatrix}\right]$ pour 54 secondes d'entraînement. D'autres scénarios présentent de meilleures performances de classification, mais leur temps de formation est trop élevé pour être considéré avec des contraintes temps réel.

3.4.2.2.2 Variation de l'ordre du RkNN

Pour les prochaines expériences, nous nous concentrons sur l'amélioration des performances de notre classifieur en ajustant l'ordre de l'algorithme RkNN. En d'autres termes, puisque le RkNN calcule le nombre d'échantillons de sacs qui ont un échantillon d'essai (requête) parmi leur ensemble de voisins k les plus proches, nous voulons saisir comment le nombre de voisins «k» les plus proches que nous avons établi pour chaque sac influence le rendement de l'algorithme de classification. Avant de passer en revue les scénarios, il convient de noter à la figure 3.9 que l'ordre RkNN n'affecte que légèrement (presque pas) le temps d'entraînement, puisqu'il est impliqué dans la génération des cercles (choisir le k NN le plus éloigné pour déterminer le rayon), qui influe sur le rendement de la classification en termes de précision, de rappel, de mesure F1 et de matrice de confusion, mais non sur les temps de formation et d'essai. Contrairement au nombre et à la taille des sacs, cela affecte le nombre de cercles impliqués dans le processus de formation et donc dans les périodes de formation, comme le montre la figure 3.8.

Comme les scénarios utilisant 3 et 5 sacs parviennent à trouver de bons compromis entre la précision, le rappel, la mesure-F1 et la matrice de confusion, ainsi que les temps de formation et de test par rapport aux scénarios avec 10 sacs, nous avons décidé de limiter nos tests pour faire varier le RkNN à ces deux cas de figure seulement.

- CAIDA :

  Pour la base de données CAIDA, en premier lieu, il convient d'observer aux figures 3.11 et 3.12 que l'augmentation de l'ordre RkNN augmente la stabilité de la classification. Nous suivons la même approche que dans la section précédente, pour sélectionner le scénario optimal. Avec 5 sacs (comme le montre la figure 3.11), le meilleur compromis est obtenu

avec un ordre RkNN égal à 10, en utilisant 20% de l'ensemble d'entraînement dans chaque sac. Ce scénario fournit une précision de 65%, un rappel de 61%, une mesure F1 de 63%, une matrice de confusion égale à $\left[\begin{smallmatrix}3808 & 47\\ 56 & 89\end{smallmatrix}\right]$ en 36 secondes d'entraînement. Avec 3 sacs (figure 3.12), bien que le scénario avec un ordre RkNN fixé à 10 pour 40% d'entraînement dans chaque sac surpasse tous les autres scénarios avec 68% de précision, 63% de rappel, 66% de mesure F1 et une matrice de confusion de $\left[\begin{smallmatrix}3812 & 43\\ 53 & 92\end{smallmatrix}\right]$, ces résultats sont obtenus en 42 secondes tandis que le scénario utilisant l'ordre RkNN égal à 20 avec des tailles de sac de 20% de l'ensemble de données de formation fournit 62% de précision, 64% de rappel, 63% F1 mesure et une matrice de confusion égale à $\left[\begin{smallmatrix}3798 & 57\\ 52 & 93\end{smallmatrix}\right]$ en 22 secondes seulement. Enfin, nous considérons le dernier scénario comme optimal pour l'ensemble de données CAIDA.

- UNI1 :

  Pour le deuxième ensemble de données (UNI1, fig. 3.13, 3.14 et 3.15), avec 5 sacs et kNN=10, le meilleur scénario est toujours à 55% de précision, 58% de rappel, 57% de mesure F1, 80% d'AUC et une matrice de confusion bien équilibrée, il est obtenu avec des sacs de 40% du set d'entraînement. Ce scénario est surpassé par un autre avec 3 sacs de 40% et kNN=20, qui se traduit par 61% de précision, 53% de rappel, 57% de mesure F1, 76% d'AUC et une matrice de confusion $\left[\begin{smallmatrix}3877 & 84\\ 113 & 129\end{smallmatrix}\right]$ en 63 secondes.

- UNI2 :

  Pour l'ensemble de données UNI2 (3.16, 3.17 et 3.18), le scénario optimal reste le même, c'est-à-dire 3 sacs de 60% avec kNN=10, ce qui donne 82% de précision, 100% de rappel, 90% de mesure F1, 99% d'AUC et une matrice de confusion équilibrée pendant 3,24 secondes.

- UNIBS :

  Pour la base de données UNIBs (3.19, 3.20 et 3.21), nous améliorons le scénario optimal représenté par 5 sacs de 20% et kNN=10, par le scénario avec kNN=5 obtenant 69% de précision, 71% de rappel, 70% de mesure F1, 78% d'AUC et une matrice de confusion égale à$\left[\begin{smallmatrix}2291 & 349\\ 403 & 827\end{smallmatrix}\right]$ pendant 53,31 secondes d'entraînement. Ce scénario peut être encore plus amélioré spécialement en termes de matrice de confusion en utilisant 3 sacs de 40% et un ordre kNN égal à 5. Ainsi, le scénario optimal pour l'ensemble de données UNBS

obtient 71% de précision, 73% de rappel, 72% de mesure F1, 80% d'AUC et une matrice de confusion égale à $\begin{bmatrix} 2272 & 368 \\ 335 & 895 \end{bmatrix}$ en 68,31 secondes d'entrainement.

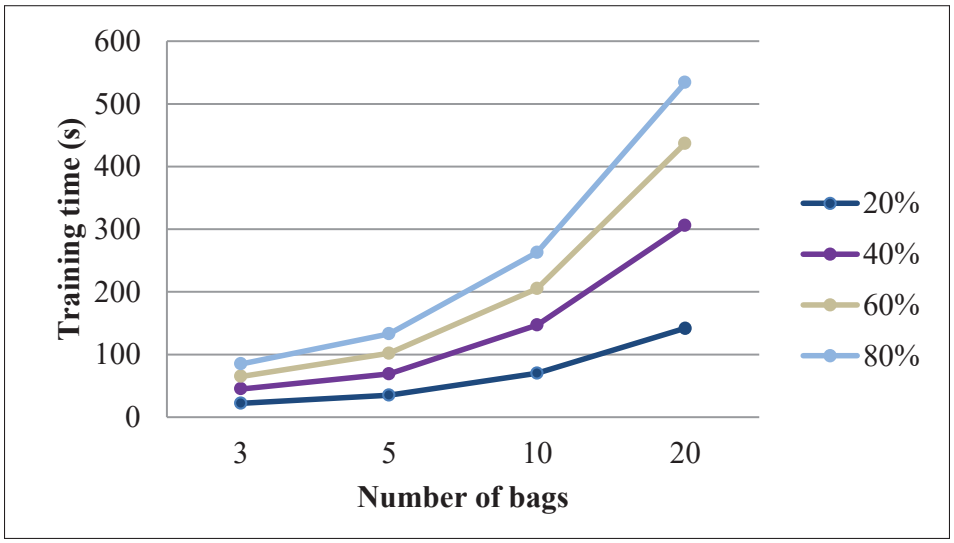


Figure 3.8 Évolution du temps d'entraînement en fonction du nombre et de la taille des sacs (CAIDA)

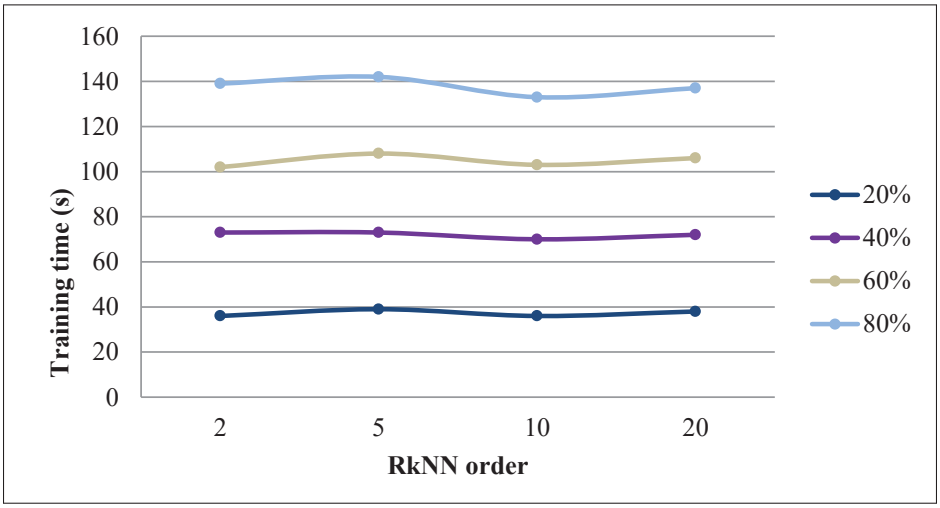


Figure 3.9 Évolution du temps d'entraînement en fonction du de l'ordre du RkNN (CAIDA)

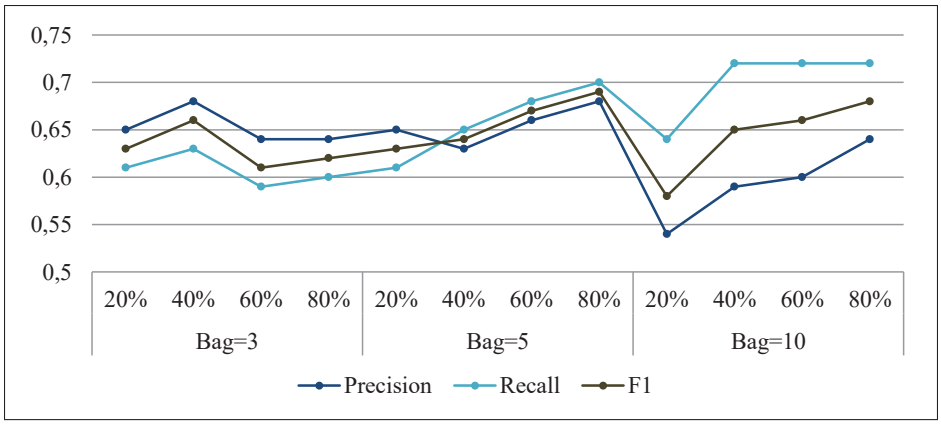


Figure 3.10 Performance de l'approche de classification proposée en utilisant différents nombres et tailles de sacs (CAIDA)

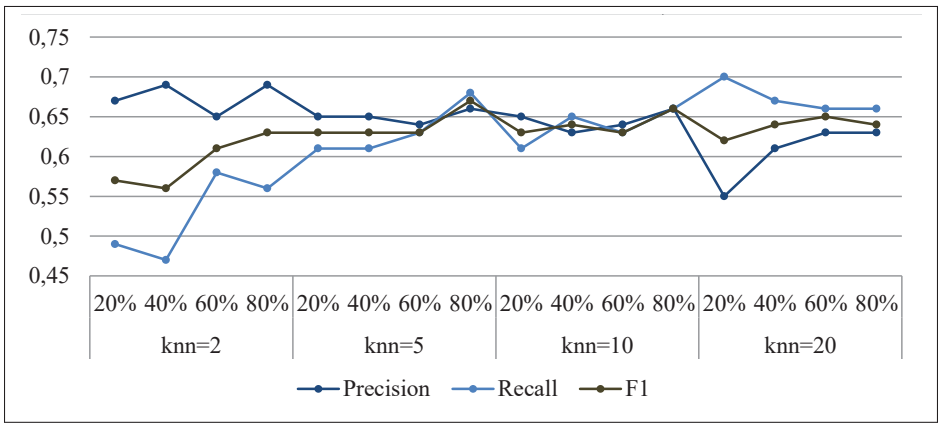


Figure 3.11 Performance de l'approche de classification proposée en utilisant 5 sacs de différentes tailles (CAIDA)

3.4.2.2.3 Comparaison des performances obtenues par notre approche avec des travaux connexes

Étant donné que la plupart des méthodes de classification avec des données déséquilibrées reposent sur des approches de niveau données ou sur des algorithmes d'ensemble Fernández *et al.* (2017), nous proposons de comparer nos performances à cinq algorithmes de sous-échantillonnage (voisin le plus proche modifié, voisin le plus proche répété, tout kNN, RUS, TomekLink), quatre algorithmes de suréchantillonnage (ADASYN, KmeansSMOTE, SMOTE,

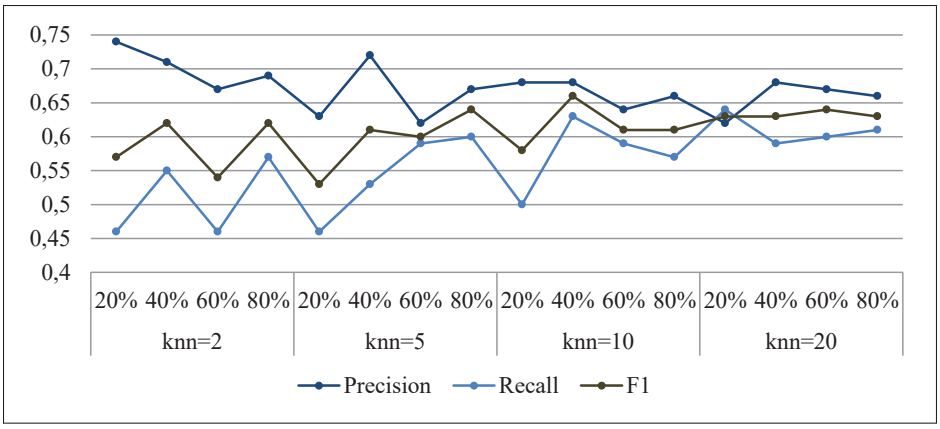


Figure 3.12 Performance de l'approche de classification proposée en utilisant 3 sacs de différentes tailles (CAIDA)

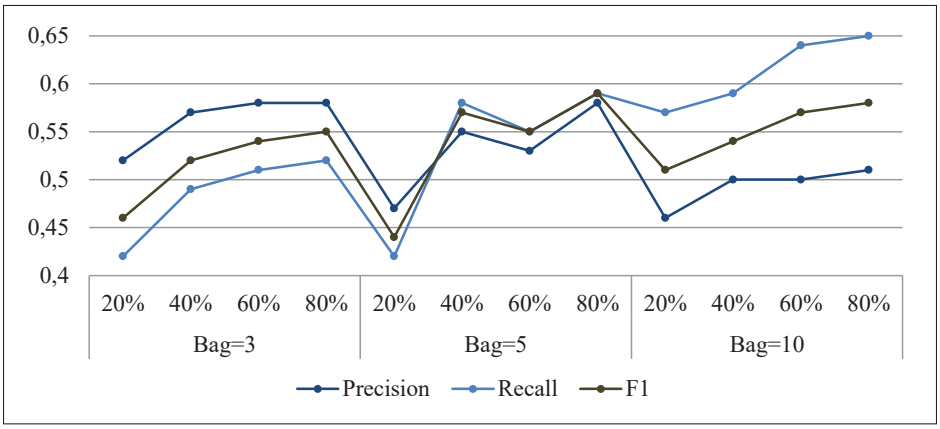


Figure 3.13 Performance de l'approche de classification proposée en utilisant différents nombres et tailles de sacs (UNI1)

ROS), un hybride (SMOTE avec TomekLink) et deux stratégies de niveau algorithmique (un bagging aléatoire fondé sur la forêt équilibré au niveau des données, RUSBoost). Des détails sur ces algorithmes peuvent être trouvés dans Gómez *et al.* (2019). De plus, nous comparons dans cette section notre stratégie de corrélation sensible aux coûts pour la classification du trafic DC déséquilibré en ligne à trois autres algorithmes sensibles aux coûts avec respectivement la méthode J48, la stratégie bagging et Forêt aléatoire comme apprenants de base.

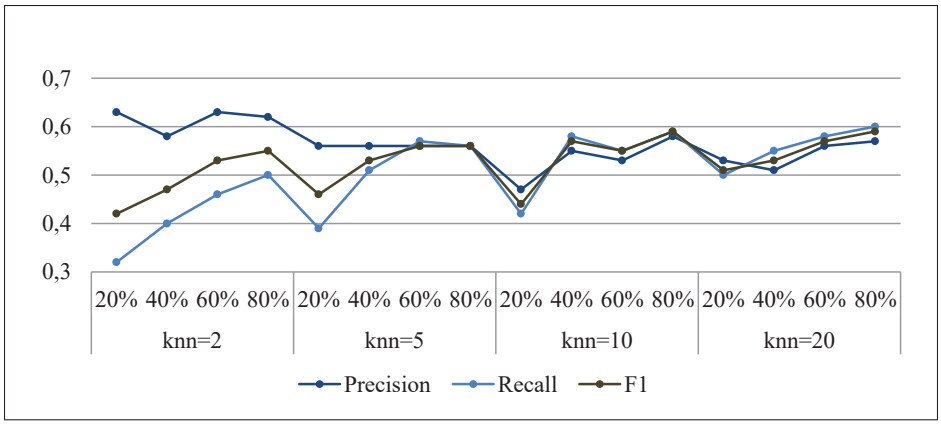


Figure 3.14 Performance de l'approche de classification proposée en utilisant 5 sacs de différentes tailles (UNI1)

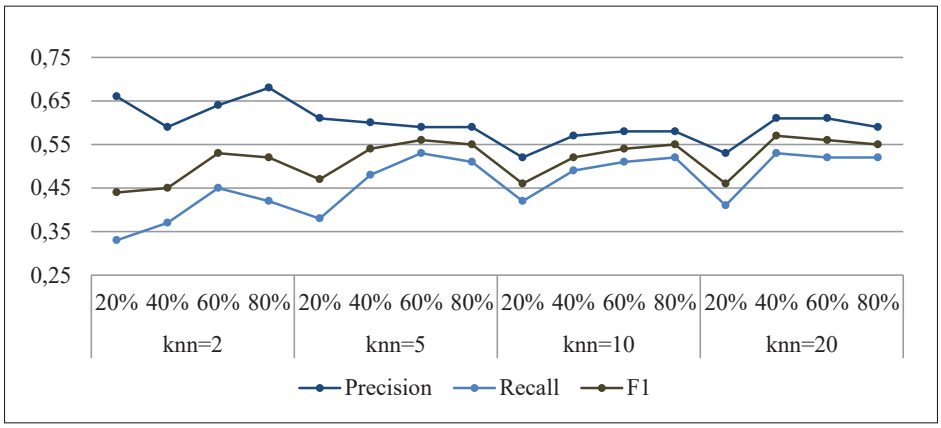


Figure 3.15 Performance de l'approche de classification proposée en utilisant 3 sacs de différentes tailles (UNI1)

Il convient de mentionner que l'algorithme sensible aux coûts implémentant bagging en tant qu'apprenant de base ressemble à MetaCost, un algorithme souvent cité lors de l'introduction de la classification sensible aux coûts Gómez *et al.* (2019); Jain, Kotsampasakou & Ecker (2018). Meta dans MetaCost désigne le méta-apprentissage défini dans Shilbayeh *et al.* (2015) comme le processus d'apprentissage à partir de diverses expériences cumulatives. Les algorithmes d'ensemble appartiennent alors à la famille des méta-apprenants. Nous avons calculé la matrice des coûts pour les algorithmes sensibles aux coûts comme dans Gómez *et al.* (2019), en suivant

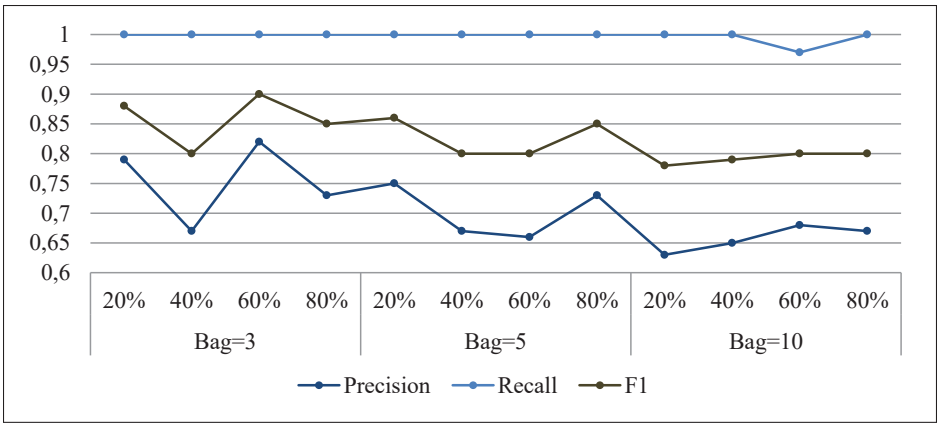


Figure 3.16 Performance de l'approche de classification proposée en utilisant différents nombres et tailles de sacs (UNI2)

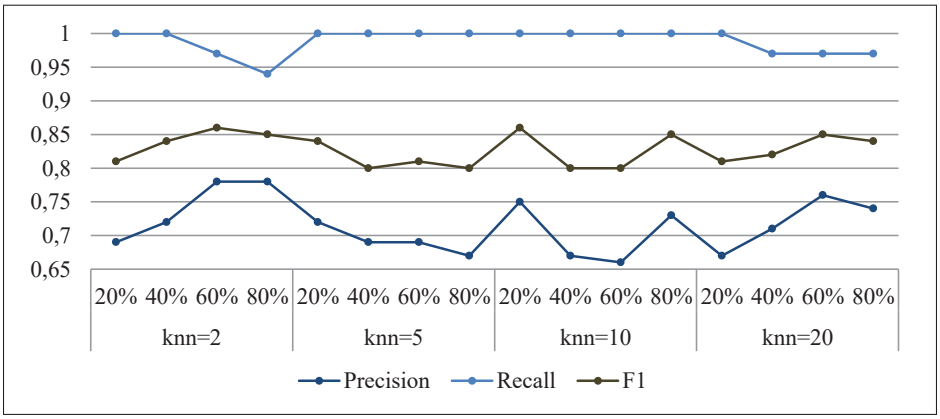


Figure 3.17 Performance de l'approche de classification proposée en utilisant 5 sacs de différentes tailles (UNI2)

l'équation ci-dessous. En d'autres termes, le coût d'erreur de classification de la classe i en j, C_{ij} dépend du nombre de flux dans chaque classe M_i ou M_j.

$$C_{ij} = \begin{cases} log_{10}(M_i)/log_{10}(M_j), if\ i \neq j \\ 0, otherwise, \end{cases} \tag{3.9}$$

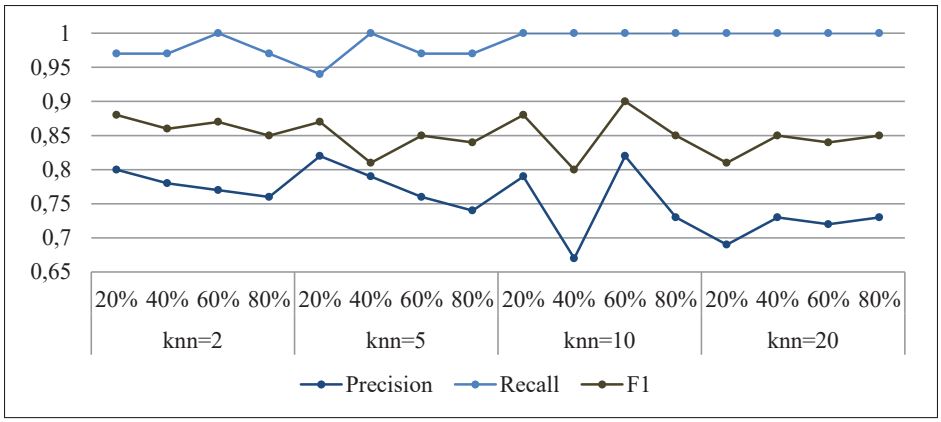


Figure 3.18 Performance de l'approche de classification proposée en utilisant 3 sacs de différentes tailles (UNI2)

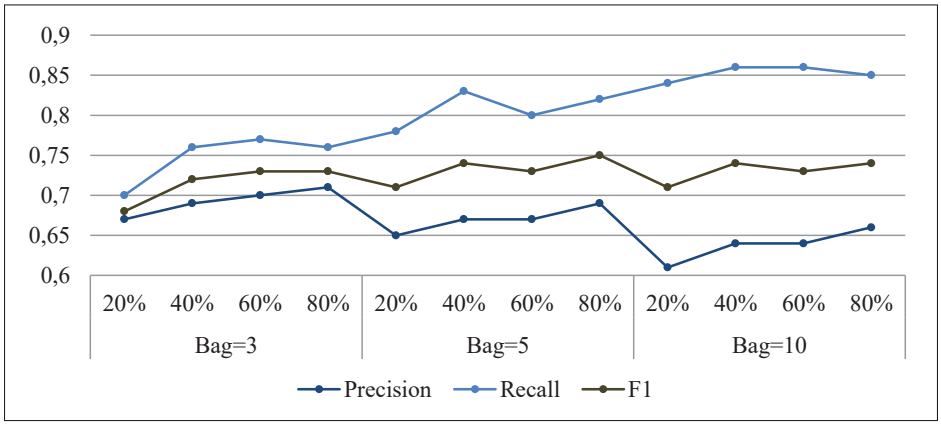


Figure 3.19 Performance de l'approche de classification proposée en utilisant différents nombres et tailles de sacs (UNIBs)

Des expériences ont été menées sur quatre bases de données et sont résumées aux tableaux ci-dessous. Les mesures de performances utilisées pour comparer les algorithmes sont la précision, le rappel, la mesure F1, l'AUC et Kappa.

Il convient de mentionner que la mesure Kappa est mesurée entre les labels de ground truth, les labels obtenus par notre approche et les labels obtenus par l'algorithme auquel nous comparons notre algorithme, raison pour laquelle il n'y a pas de valeur Kappa dans la rangée «notre approche» de chacun des tableaux.

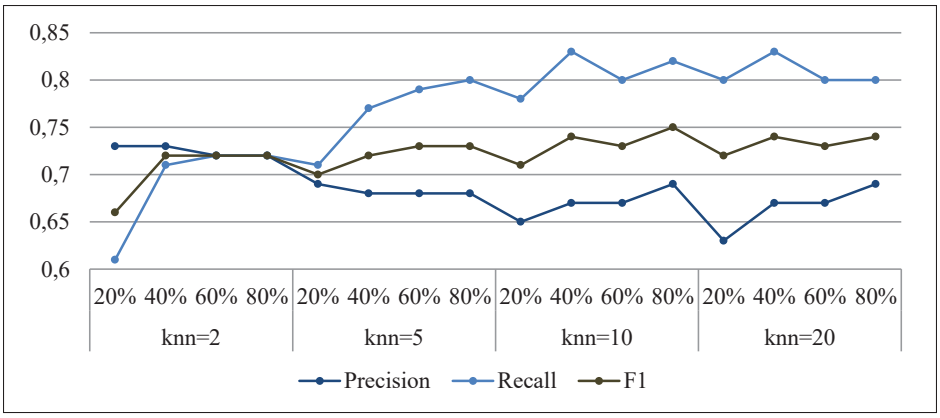


Figure 3.20 Performance de l'approche de classification proposée en utilisant 5 sacs de différentes tailles (UNIBs)

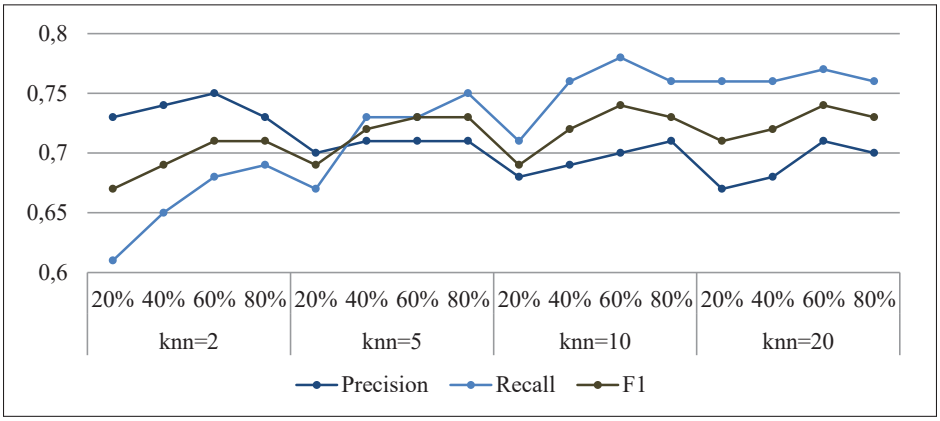


Figure 3.21 Performance de l'approche de classification proposée en utilisant 3 sacs de différentes tailles (UNIBs)

- CAIDA :

Premièrement, pour les mesures AUC et kappa, nous remarquons que les algorithmes de suréchantillonnage et hybrides donnent de bonnes valeurs. Les algorithmes d'ensemble donnent des AUC plus élevés allant de 73 à 92% exprimant un degré élevé de séparabilité du classificateur entre les classes, mais avec 40% Kappa pour l'algorithme RUS, le Kappa avec les autres algorithmes est meilleur, ce qui indique un bon accord entre les classificateurs. Enfin, les algorithmes sensibles aux coûts présentent une AUC acceptable et un Kappa faible.

Tableau 3.1 Comparaison des mesures de performance entre notre approche et celles existantes pour la base de données CAIDA

Dataset	**CAIDA**					
Algorithm	Precision	Recall	F1	AUC	Time(s)	Kappa
SMOTE	0,51	0,74	0,61	0,85	4,36	0,663
kMeans Smote	0,75	0,45	0,56	0,72	3	0,63
ROS	0,67	0,58	0,62	0,78	2,19	0,7
ADASYN	0,51	0,73	0,6	0,85	4,75	0,7
RUS	0,24	0,97	0,38	0,92	0,13	0,36
TomekLink	0,68	0,52	0,59	0,75	1,66	0,7
Condenced NN	0,61	0,57	0,59	0,779	344,77	0,619
Edited NN	0,56	0,74	0,64	0,86	1,73	0,73
Repeaded NN	0,49	0,8	0,61	0,88	5,66	0,7
All kNN	0,52	0,77	0,62	0,87	3	0,73
Hybrid	0,52	0,75	0,61	0,86	5,56	0,66
Balanced RF	0,27	0,96	0,42	0,929	1,32	0,4
RUS Boost	0,39	0,49	0,43	0,73	0,76	0,4
CS (J48)	0,7	0,32	0,44	0,66	0,25	0,43
CS (Bagging)	0,78	0,31	0,45	0,66	0,89	0,44
CS (RF)	0,78	0,32	0,45	0,66	3,03	0,45
Our approach	0,62	0,64	0,63	0,80	22	/

En ce qui concerne les mesures de précision, de rappel et de F1, nous remarquons que tous les algorithmes entraînent une haute précision et un faible rappel ou vice versa et jamais un équilibre entre les deux. Par rapport à ces algorithmes, notre approche proposée aboutit à des mesures de performance plus équilibrées avec 62% de précision, 64% de rappel, 63% de mesure F1 et un AUC plus élevé (80%) en 22 secondes d'entraînement.

- UNI1 :

  Nous remarquons un bon accord entre notre classificateur et le suréchantillonnage, la plupart des algorithmes de sous-échantillonnage (sauf RUS), et les stratégies hybrides avec des valeurs Kappa autour de 70%, mais d'autre part, nous observons des valeurs Kappa plus petites avec les algorithmes de type ensemble et les stratégies sensibles aux coûts, présentant les pires performances de classification. Notre scénario surpasse le meilleur classificateur (TomekLink) en termes de rappel et d'AUC .

Tableau 3.2 Comparaison des mesures de performance entre notre approche et celles existantes pour la base de données UNI1

Dataset	**UNI1**					
Algorithm	Precision	Recall	F1	AUC	Time(s)	Kappa
SMOTE	0,48	0,71	0,58	0,83	4,25	0,62
kMeans Smote	0,67	0,44	0,53	0,71	4,55	0,65
ROS	0,61	0,58	0,59	0,77	2,57	0,7
ADASYN	0,47	0,69	0,56	0,81	4,56	0,61
RUS	0,23	0,88	0,37	0,85	0,28	0,31
TomekLink	0,65	0,5	0,57	0,74	3,21	0,68
Condenced NN	0,52	0,62	0,56	0,79	589,88	0,60
Edited NN	0,5	0,67	0,57	0,81	2,82	0,697
Repeaded NN	0,43	0,71	0,53	0,84	9,41	0,627
All kNN	0,46	0,71	0,56	0,82	5	0,64
Hybrid	0,47	0,72	0,57	0,83	7,08	0,61
Balanced RF	0,3	0,88	0,44	0,87	2,34	0,41
RUS Boost	0,36	0,6	0,45	0,76	1,18	0,42
CS (J48)	0,63	0,32	0,42	0,65	0,53	0,4
CS (Bagging)	0,91	0,238	0,37	0,61	1,59	0,36
CS (RF)	0,8	0,295	0,43	0,64	5,43	0,41
Our approach	0,61	0,53	0,57	0,78	63	/

- UNI2 :

  Nous remarquons une plus grande stabilité avec des valeurs plus élevées pour toutes les performances de classification pour presque tous les algorithmes, même les algorithmes d'ensemble et ceux sensibles aux coûts. Les valeurs Kappa avoisinent les 90% tandis que les AUC excèdent les 90 excluant ainsi la possibilité d'une classification aléatoire selon notre approche. Il convient de mentionner que les approches sensibles aux coûts offrent les meilleures performances pour UNI2 par rapport aux bases de données précédentes. Bien que quelques secondes plus lente, notre approche surpasse tous les algorithmes et se traduit par 82% de précision, 100% de rappel, 90% de mesure F1 et 99% d'AUC.
- UNIBS :

  Enfin, pour la base de données UNIBs, nous observons également de bonnes valeurs Kappa pour tous les algorithmes (plus de 70% mais un peu moins pour les algorithmes sensibles aux

Tableau 3.3 Comparaison des mesures de performance entre notre approche et celles existantes pour la base de données UNI2

Dataset	**UNI2**					
Algorithm	Precision	Recall	F1	AUC	Time(s)	Kappa
SMOTE	0,82	0,94	0,87	0,96	0,1	0,9
kMeans Smote	0,78	0,94	0,85	0,978	0,31	0,88
ROS	0,73	0,97	0,83	0,97	0,07	0,9
ADASYN	0,79	0,94	0,86	0,86	0,12	0,88
RUS	0,66	1	0,8	0,98	0,05	0,95
TomekLink	0,86	0,73	0,79	0,85	0,09	0,71
Condenced NN	0,89	0,73	0,8	0,86	1,93	0,78
Edited NN	0,77	1	0,87	0,989	0,08	0,93
Repeaded NN	0,63	1	0,75	0,98	0,124	0,93
All kNN	0,63	1	0,78	0,98	0,12	0,93
Hybrid	0,74	0,94	0,83	0,859	0,15	0,952
Balanced RF	0,65	1	0,79	0,9814	0,17	0,94
RUS Boost	0,76	0,76	0,76	0,87	0,19	0,92
CS (J48)	0,7	0,909	0,88	0,95	0,01	0,88
CS (Bagging)	0,87	0,901	0,88	0,95	0,03	0,88
CS (RF)	0,87	0,91	0,89	0,95	0,1	0,88
Our approach	0,82	1	0,9	0,99	3,24	/

coûts). De plus, la plupart des valeurs AUC dépassent 80%. En ce qui concerne la précision, le rappel et la mesure F1, ils fluctuent autour de 70% pour les algorithmes de suréchantillonnage et d'ensemble et ont des valeurs entre 60% et 70% pour le sous-échantillonnage. Les algorithmes sensibles aux coûts présentent des améliorations importantes et surpassent le reste des méthodes. Notre approche obtient des résultats satisfaisants estimés à 71% de précision, 73% de rappel et 70% de mesure F1, ce qui est mieux que plusieurs algorithmes, notamment les stratégies de sous-échantillonnage s'approchant des meilleures valeurs de performance.

Il est à noter que pour toutes les bases de données, le pire algorithme en termes de temps d'exécution est le voisin le plus proche condensé (Condensed Nearest neighbor CNN). Cependant, il produit de meilleurs résultats que l'algorithme le plus rapide, qui pour la plupart des bases de

Tableau 3.4 Comparaison des mesures de performance entre notre approche et celles existantes pour la base de données UNIBs

Dataset	**UNIBs**					
Algorithm	Precision	Recall	F1	AUC	Time(s)	Kappa
SMOTE	0.73	0.78	0.75	0.82	2,61	0,73
kMeans Smote	0.79	0.69	0.74	0.80	2,88	0,735
ROS	0.75	0.74	0.75	0.81	2,35	0,75
ADASYN	0.71	0.81	0.75	0.82	3,58	0,72
RUS	0.67	0.85	0.75	0.82	1,05	0,71
TomekLink	0.75	0.74	0.75	0.81	2,26	0,75
Condenced NN	0,63	0,83	0,71	0,80	911,87	0,59
Edited NN	0.64	0.89	0.74	0,82	1,82	0,707
Repeaded NN	0.60	0.91	0.73	0.81	3,62	0,656
All kNN	0.62	0.90	0.73	0.82	2,84	0,7
Hybrid	0.72	0.78	0.75	0.81	3,52	0,74
Balanced RF	0.71	0.81	0.76	0.83	7	0,75
RUS Boost	0,77	0,62	0.9	0.76	3	0,71
CS (J48)	0,86	0,87	0,86	0.77	0,54	0,56
CS (Bagging)	0,88	0,88	0,88	0,81	1,6	0,62
CS (RF)	0,89	0,89	0,89	0.82	5,54	0,64
Our approach	0,71	0,73	0,72	0,80	68,31	/

données est RUS. Pour CAIDA par exemple, CNN (344,77 secondes d'entraînement) fournit 60% de précision, 57% de rappel, 59% de mesure F1, 78% d'AUC et 62% kappa contre 24% de précision, 97% de rappel, 38% de mesure F1, 92,7% d'AUC et 36% de Kappa avec RUS (0,13 seconde).

En résumé, en ce qui concerne notre approche, bien que plus lente que certaines approches, elle surpasse les autres algorithmes en termes de précision, de rappel et de mesure F1 et présente des valeurs AUC et Kappa satisfaisantes. Si nous examinons de plus près les résultats obtenus, nous discutons du fait que les performances de classification ne sont pas directement liées au niveau de déséquilibre, par exemple, si nous comparons CAIDA et UNI1, CAIDA est plus déséquilibré que UNI1, cependant, notre algorithme aboutit à de meilleures performances (62% de précision, 64% de rappel, 63% de mesure F1 et 80% d'ASC pour CAIDA contre 61% de précision, 53% de rappel, 57% de mesure F1 et 78% d'ASC pour UNI1). En outre, ces performances ne sont pas

liées à la taille des bases de données (bien que ce dernier paramètre affecte le temps d'exécution de notre algorithme). Ces performances peuvent être liées à la qualité des traces de trafic.

3.4.3 Analyse statistique

Dans la section précédente, nous avons étudié l'efficacité de notre approche contre le déséquilibre des données et pour la classification en ligne. Nous avons également comparé les résultats obtenus suivant le meilleur scénario pour chaque base de données aux performances de classification des approches de pointe. Cependant, afin de valider la signification statistique de ces résultats, il est important d'établir une analyse statistique entre les différents algorithmes sur les différentes bases de données Gómez *et al.* (2019). En effet, lorsque plusieurs classificateurs sont appliqués sur divers ensembles de données, leurs performances doivent être différentes, et il est donc important de vérifier si ces différences sont aléatoires ou réelles Demšar (2006).

Il existe deux catégories de méthodes d'analyse statistique, paramétriques et non paramétriques. Bien que le premier repose sur plusieurs hypothèses concernant la distribution des données, le second n'exige aucune hypothèse. Lorsque la normalité des données et l'égalité de variance ou comme dans notre cas la distribution est inconnue, les méthodes non paramétriques sont le meilleur choix Ali & Bhaskar (2016). Par conséquent, nous avons décidé de valider statistiquement nos résultats par le test de Friedman, l'une des approches non paramétriques les plus utiliséesDemšar (2006).

Le test de Friedman consiste à confirmer ou non une hypothèse nulle selon laquelle les mesures répétées obtenues différemment ont la même distribution. En d'autres termes, les différents algorithmes ont un comportement similaire dans toutes les bases de données. Nous visons à rejeter l'hypothèse nulle et ainsi prouver la différence statistique entre les algorithmes déployés.

Le test de Friedman commence par trier les algorithmes en fonction de la valeur de performance de classification pour chaque ensemble de données, après quoi il attribue un score à chaque algorithme en fonction de son rang. Enfin, le score de Friedman $\chi_F 2$ est calculé suivant l'équation

ci-dessous. Où N représente le nombre de bases de données sur lesquels les tests ont été établis, k le nombre d'algorithmes formés, et enfin R_j est le score obtenu après le tri des algorithmes.

$$\chi_F^2 = \frac{12N}{k(k+1)} \left[\sum_j R_j^2 - 0.25k * (k+1) \right] \tag{3.10}$$

Dans le tableau ci-dessous, nous présentons les scores de Friedman obtenus en comparant le suréchantillonnage, le sous-échantillonnage, les approches hybrides, les méthodes algorithmiques et les approches sensibles aux coûts sur différents ensembles de données. Afin d'analyser ces valeurs, nous mesurons également leurs valeurs P correspondantes. Les similitudes entre les algorithmes sont exprimées par des valeurs élevées de P, à l'inverse, des valeurs plus petites expriment une signification statistique. À en juger par les valeurs très faibles de la valeur P, nous pouvons confirmer les différences statistiques entre les algorithmes.

Tableau 3.5 Statistical analysis, Friedman's test

	Precision	Recall	F1-measure	AUC
Friedman's score	35.025	38.658	44.354	34.682
P_values	<0.0001	<0.0001	<0.0001	<0.0001

3.4.4 Conclusion

À travers ce chapitre, nous avons présenté notre approche de classification de type stratégie sensible aux coût incluant une nouvelle adaptation de l'algorithme RkNN jumelé à un bagging de forêts aléatoires. Un avantage de notre solution est sa capacité à sélectionner n'importe quel ordre RkNN. En effet, des ordres RkNN plus ou moins élevés n'affectent guère le temps de formation. Ainsi, nous pouvons régler efficacement les hyperparamètres de notre approche sans augmenter le temps d'exécution. Cependant, bien qu'efficace, l'algorithme Forêt aléatoire peut rendre la classification, gourmande en ressources et en temps, selon le nombre d'arbres. Par conséquent, nous testerons l'efficacité de notre approche, avec d'autres stratégies de sélection et d'autres apprenants de base dans les travaux futurs. Pour ce qui est du temps d'essai, nous

élaborerons une stratégie qui ne mesurera pas la corrélation RkNN pour chaque échantillon d'essai.

CHAPITRE 4

MÉTHODOLOGIE

4.1 Introduction

À travers ce chapitre, nous allons tenter de répondre aux questions de recherche énoncées dans le premier chapitre. Ainsi, après avoir présenté les solutions existantes pour remédier au problème de planification de trafic dans un environnement agnostique dans le chapitre de revue de littératures, leurs atouts et leurs limitations, nous allons expliquer à travers ce chapitre la particularité de l'approche que nous proposons, détailler l'aspect mathématique et présenter la solution algorithmique que nous avons testée et comparée aux méthodes existantes. Pour commencer, nous allons effectuer une étude comparative de trois des solutions de planification les plus citées, afin de déterminer les besoins à combler. Par la suite, nous décrirons et modéliserons notre système. Le chapitre est finalement clôturé avec l'approche de résolution du problème.

4.2 Approches existantes en planification agnostique

Dans la figure 4.1 nous comparons certaines des approches de planification existantes qui couvrent la planification indépendante de l'information. Nous présentons un exemple pour illustrer l'inefficacité de ces travaux en considérant les flux de souris et éléphants. Plus précisément, quatre flux entrant dans le réseau à la plage horaire actuelle et destinée à être transférés avant un délai spécifique sont représentés par des tuples où le premier chiffre représente la taille du flux et le second dans son délai. Nous supposons par cet exemple que ces informations sont disponibles afin de pouvoir expliquer l'exemple, mais aussi pour mettre l'accent sur les défis de la planification agnostique de l'information.

Dans le premier graphique, AMOEBA, envisage la possibilité que les flux puissent spécifier leurs besoins en termes de délai maximum. Cette méthode a pour objectif de compléter autant de flux que possible avant ce délai limite. Ameoba suit le concept du tout ou rien, ce qui signifie que si un flux ne peut pas être transmis, il est supprimé. Dans cet exemple, nous pouvons voir

que le flux F2 n'avait pas assez de capacité avant le délai limite, donc AMOEBA choisit de ne pas l'accepter. Tempus, d'autre part, ne se concentre pas sur le transfert du flux, mais plutôt la maximisation de la fraction du flux transmit, et dans cet exemple, l'algorithme a perdu le flux F3. Bien que les concepts soient différents (tout ou rien versus maximisation de la fraction de trafic transmise), Ameoba et Tempus visent tous deux à maximiser l'entité transférée avec la contrainte de respecter un délai ferme. Dans ce cas, les données deviennent insignifiantes si elles sont reçues après le délai limite.

TINA tente d'atteindre l'équité tout en garantissant des délais exprimés par la maximisation de l'utilité. Ainsi, au-delà des délais fermes, TINA introduit une autre contrainte : l'équité. Dans cette approche, l'utilité est représentée par le temps de transmission, donc F2 n'est pas sélectionné parce que l'échéance de F3 ne sera pas respectée et F3 n'est pas non plus sélectionné parce qu'il ne garantit pas les critères d'équité puisque F1 et F3 sont tous deux des flux de souris.

Nous remarquons que pour les trois approches, l'algorithme a la même contrainte qui est le délai ferme. Cependant, avec des flux de souris et d'éléphants ayant chacun des exigences spécifiques, des délais stricts ou fermes peuvent entraîner une dégradation des performances, en particulier lorsque l'on considère une planification agnostique. Par exemple, dans le cas d'Ameoba, cela entraînera toujours la suppression de longs flux, Tempus ne transmettant que des parties de ces flux d'éléphants, et pour TINA, cela peut entraîner la suppression d'un nombre important de flux de souris afin de garantir également l'équité même s'ils sont considérés comme critiques. Dans le dernier graphique, représentant une approche proposée, nous essayons ainsi de trouver un équilibre entre l'exigence de délai et la criticité des flux de souris, nous proposons donc d'intégrer le concept de délai mixte. Les flux de souris seront limités par un délai ferme tandis que les flux d'éléphants toléreront des retards à un certain niveau et seront donc limités par un délai léger.

Ainsi, notre objectif est de construire un module de planification non coûteux visant à accueillir un maximum de flux de différentes classes de trafic tout en exprimant efficacement leurs besoins en termes de délai, dans un environnement agnostique à l'information.

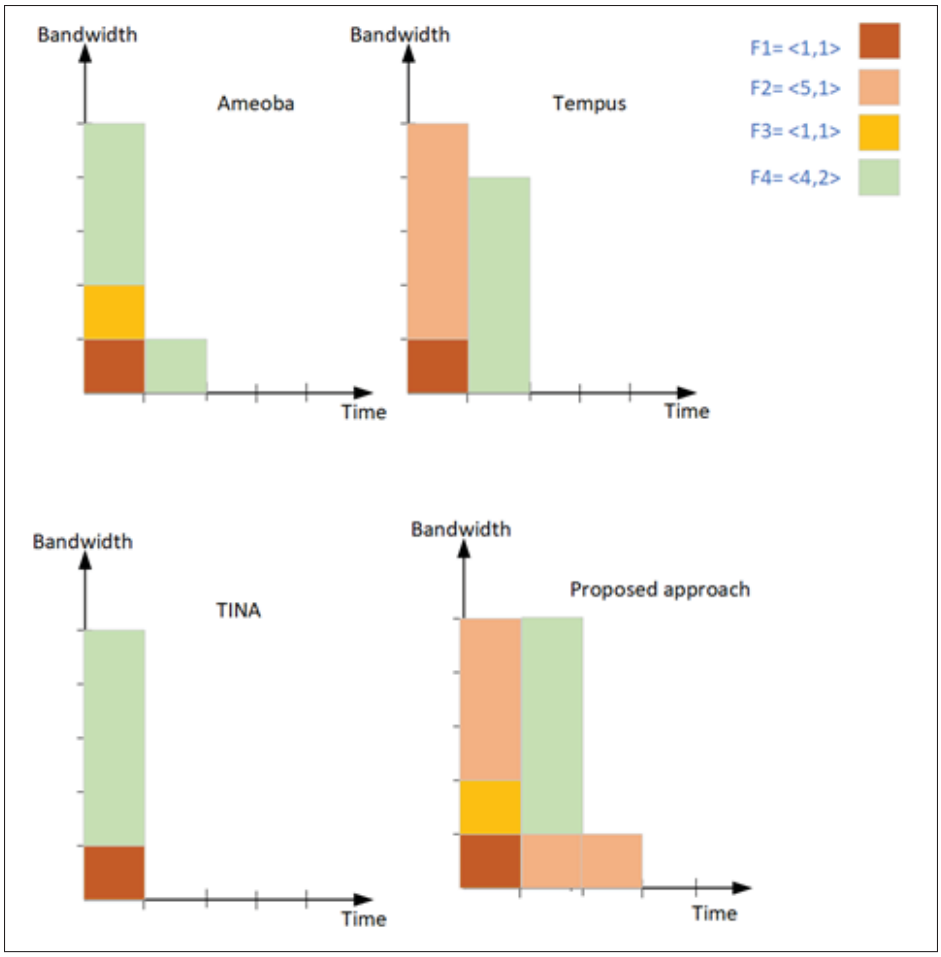


Figure 4.1 Approches de planification agnostiques

4.3 Description du système

Compte tenu du défi de l'incertitude de l'information et en même temps de l'importance de caractériser les exigences de QoS des flux, nous intégrons dans notre modèle de planification un premier module de classification en ligne du trafic tel que dans Saber *et al.* (2020). L'objectif de ce bloc est de regrouper les flux entrant en deux catégories : les souris et les éléphants, sur la base des informations de paquets/flux disponibles dans les premiers paquets.Nous ne détaillerons pas le module de classification dans le présent chapitre, nous allons plutôt nous concentrer sur la partie planification, car ce dernier a déjà était abordé dans les précédents chapitres.

Notre réseau d'interconnexion de centre de données, est composé de N nœuds (nœuds Edge et Core) interconnectés par des liaisons optiques L de capacité $Capl$ et avec une bande passante disponible de B_l, comme illustré par la figure ci-dessous.

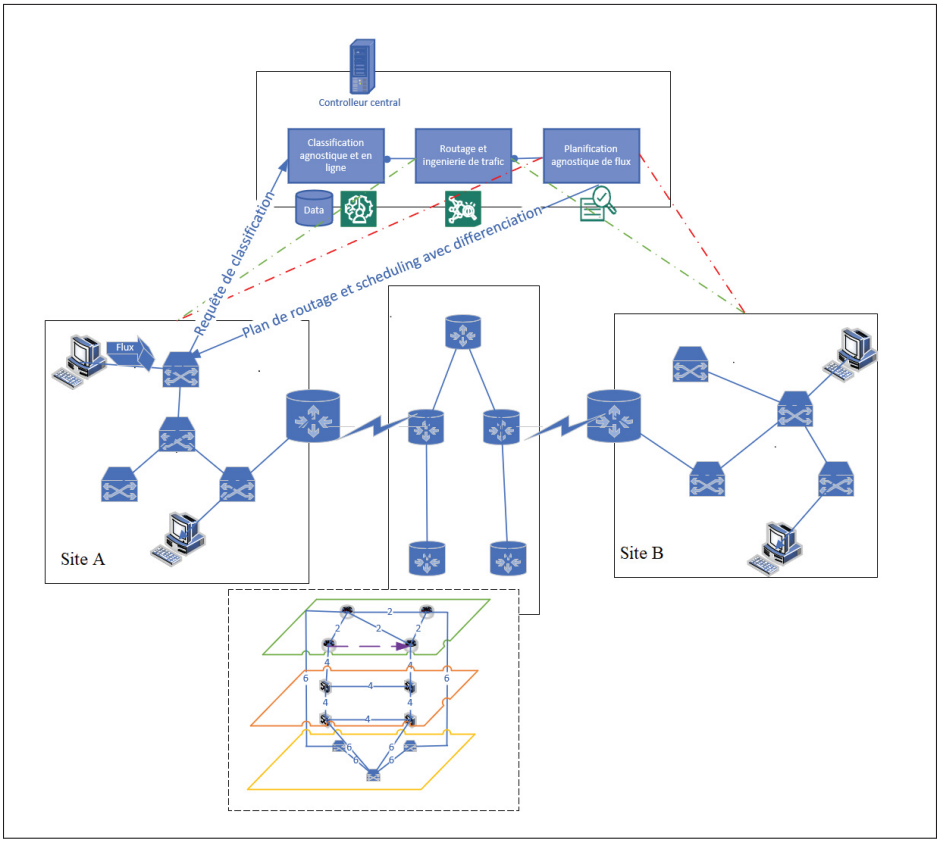


Figure 4.2 Système de planification agnostique avec classification en ligne de trafic

Afin de transporter les flux entre ces nœuds, un module de routage établit les chemins optimaux, soit $P = \{p\}$. P représente les k chemins les plus courts entre une source i et une destination j et qui sont calculés après la réception du flux.

Soit $F = \{f\}$ l'ensemble des flux soumis au réseau. Chaque flux $f \in \{F\}$ est une entité représentée par un couple (w_f, D_f),où w_f, est le poids du flux, il identifie ses exigences de QoS puisque le volume de données et le temps de service sont censés être inconnus. Ce poids peut avoir l'une des deux valeurs disponibles w_1, w_2, car nous définissons dans notre module de classification deux classes de trafic (souris et éléphants). Nous supposons que le poids alloué

aux flux de souris est plus élevé que celui alloué aux flux d'éléphants. D_f est le délai limite requis pour chaque flux.

Concernant notre module de planification, nous proposons de programmer les flux entrants en sous-flux sur les chemins de routage multiple. La plupart des approches se basent sur le protocole ICMP (l'Internet Control Message Protocol) pour garantir l'équilibrage de la charge du trafic, et qui consiste à partager un flux entre plusieurs chemins disponibles avec le même coût métrique. Cependant, en raison de son inefficacité Wang *et al.* (2019); Dong *et al.* (2019); N.M *et al.* (2016a), les approches existantes proposent de diviser les flux en entités plus petites. En N.M *et al.* (2016a), les auteurs suggèrent de transformer les flux d'éléphants en sous-flux de la taille d'un flux souris. En Wang *et al.* (2019), les auteurs étudient le concept des flowlets représentant des flux moins chargés, mais caractérisés par une taille indéfinie et proposent plutôt l'entité flowcell avec une taille fixe (64k).

Cependant, nous visons la minimisation des FCT, et de tels sous-flux de petite taille peuvent contribuer à augmenter les délais, nous proposons donc de diviser les flux d'éléphants en sous-flux en fonction du nombre de voies disponibles et de leurs capacités disponibles, tout en garantissant la priorité pour les flux de souris. Nous définissons une nouvelle entité résultant du fractionnement des flux en plus petit groupe de paquets : les sous-flux $\{s\}$. Il est important de noter que nous transmettrons les sous-flux en groupe de plus grande taille afin d'éviter la saturation des liens et la dégradation des performances des flux souris ; cette entité est dénommée burst ou rafale, plus de détails dans sections suivantes. Nous résumons le reste des notations dans le ci-dessous.

4.4 Formulation du problème

Notre système est non persistant et évite la perte de paquets grâce à la mise en mémoire tampon et à la gestion de congestion. Ainsi, les chemins de routage multiple sont établis par le module de routage et d'ingénierie, avec la condition que les flux de souris sont accueillis par les chemins

Tableau 4.1 Tableau des Notations

Notation	**Description**
N	Ensemble de nœuds
L,Cap_l,B_l	Ensemble de liens, capacité maximale, bande passante disponible sur le lien l
$F = \{f\}$	Ensemble des flux soumis au réseau
$f = \{w_f, D_f\}$	Classe et délai de transmission du flux f
$P = \{p\}$	K chemins les plus courts pour les flux entrants, calculés par le module de routage en fonction du nombre de sauts
$a_{l,p}$	Égal à 1 si le lien appartient au chemin p , 0 dans le cas contraire
C_{fsp}	Coût de traitement du sous-flux s appartenant au flux f sur le chemin p
$\sigma_{min}(w_f)$	Débit minimal requis pour le nouveau flux de la classe i σ_1, si $w_i = souris$, σ_2 if $w_i = elephant$
$S = \{s\}$	Ensemble de sous-flux composant chaque flux
z_s	Limite inférieure de la taille d'un sous-flux s
Variables de décision	***Description***
σ_{fsp}	Répartition des taux (débit) pour les sous-flux s appartenant au flux f sur le chemin p
x_{fsp}	Variable binaire égale à 1 si le sous-flux s du flux f est programmé sur le chemin p, 0 dans le cas contraire

les plus rapides disponibles, car ils ne tolèrent pas les retards, tandis que les flux d'éléphants tolèrent un nombre limité de plages horaires de retard afin de garantir la QoE.

Chaque flux arrivant à l'interface d'entrée se voit attribuer un poids exprimant ses exigences de QoS et donc son niveau de criticité. Ce poids exprimera également le débit minimum et maximum requis par le flux entrant, donc pour un flux souris et selon Xu & Li (2014), le débit maximum doit être de 10kB, nous supposons que le débit minimum est de 5 kB. Et pour un flux d'éléphant, le taux minimum requis est de 1 Mo Truong-Huu *et al.* (2017) et le taux maximum requis est de 100 Mo.

Nous formulons notre problème de planification agnostique à l'information comme le problème suivant de programmation non linéaire d'entier mixte (Mixed Integer Non-Linear Programming problem MINLP).

$$\min \sum_{p}^{P} \sum_{f}^{F} \alpha \sum_{s}^{S} x_{fsp} C_{fsp} \ + \beta \exp\left(w_f / \sigma_{fsp}\right) \tag{4.1}$$

S.t.

$$\sum_{p}^{P} x_{fsp} = 1, \forall s \in S, \forall f \in F \tag{4.2}$$

$$\sum_{s}^{S} x_{fsp}.z_s \leqslant B_p, \forall p \in P, \forall f \in F \tag{4.3}$$

$$\sum_{s}^{S} \sigma_{fsp} \geqslant \sigma_{min}(w_f), \forall p \in P, \forall f \in F \tag{4.4}$$

$$\sum_{s}^{S} \sigma_{fsp}.x_{fsp} \leqslant B_p, \forall p \in P, \forall f \in F \tag{4.5}$$

$$FCT_f = max(FCT(s)) = max(\frac{z_s.x_{fsp}}{\sigma_{fsp}}) \leqslant D_f, \forall f \in F \tag{4.6}$$

$$z_s \geqslant 64KB, \forall s \in S \tag{4.7}$$

$$x_{fsp} \in \{0, 1\}, \sigma_{fsp} \geqslant 0, \forall s \in S, \forall p \in P \tag{4.8}$$

$$C_{fsp} = \frac{Cap_p}{B_p}, \forall p \in P \tag{4.9}$$

4.4.1 La fonction objective :

L'objectif de notre problème est de minimiser, avec des informations limitées sur les flux de trafic, les coûts opérationnels ainsi que le délai de transmission représenté par un coût de pénalité associé aux retards de transmission. La première partie de la fonction objectif représente le coût opérationnel cumulé de tous les flux sur les parcours disponibles, alors que le second terme de la fonction objective représente le coût lié au retard. Cette fonction suit une courbe logarithmique puisque le coût de pénalité de retard commence à zéro à la réception du flux et augmente de

manière logarithmique jusqu'à ce qu'un maximum soit atteint pour la plage horaire représentant le délai, comme dans le graphique de la figure 4.3.

Le coût élémentaire C_{fsp} est calculé par l'équation (4.9), ou Cap_p est la capacité totale minimale de chaque lien sur le chemin p et B_p est la capacité disponible.

Il est important de clarifier la différence entre B_l et B_p. Alors que le premier est la capacité disponible sur le lien l, le dernier est la capacité minimale sur tous les liens du chemin p. En d'autres termes : $B_p = min(B_l), \forall l \in p$

4.4.2 Les contraintes :

Le problème est limité par les contraintes suivantes. La contrainte (4.2) garantit que le sous-flux est la plus petite entité de notre approche. La contrainte (4.3) indique que le nombre total de décisions de planification avec une taille limite de sous flux z_s, ne doit pas excéder la capacité de chaque chemin (contrainte (4.7)).

La contrainte (4.4) représente la contrainte de zéro trafic, et elle évite le scénario où l'algorithme sélectionne par défaut $sigma_{fsp} = 0$ pour tous les sous-flux car c'est la plus petite valeur qu'il peut trouver.

La contrainte (4.5) représente la contrainte de dépassement de capacité. Ainsi, la totalité des débits attribuée aux flux à transmettre sur chacun des chemins multiples ne doit pas dépasser la bande passante disponible sur ce chemin.

Dans la contrainte (4.6), (Flow Completion Time) FCT_i ne doit pas dépasser le delai maximum pour les flux de souris et la delai toléré pour les flux d'éléphants. Dans le modèle, nous exprimons ces deux délais avec le même paramètre D_i.

Enfin, la contrainte (4.8) inclut des limites de valeur pour les deux variables de décision ($sigma_{fsp}$ et x_{fsp}).

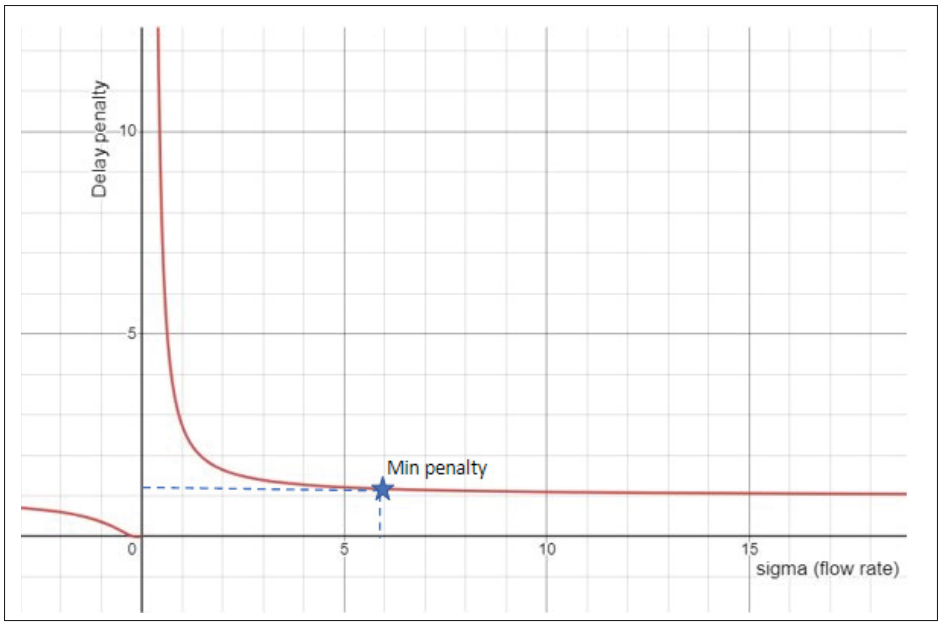


Figure 4.3 Évolution de la fonction de pénalité sur l'allocation du débit

4.4.3 Relaxation du problème et algorithme de planification et de répartition des flux

Notre problème de planification indépendant de l'information est NP-Hard puisqu'il s'agit d'un problème MINLP. L'aspect de programmation d'entier mixte (Mixed Integer) provient de la combinaison de la variable continue $sigma_{fsp}$ et de la variable binaire x_{fsp}, tandis que l'aspect non linéaire résulte de la fonction exponentielle dans la fonction objective du problème.

Afin de résoudre le problème, nous l'assouplissons en deux sous-problèmes : le problème d'allocation de demande à coût minimum (problème d'ordonnancement de flux) qui est un problème d'affectation généralisée (GAP Generalized Assignment Problem) et le problème d'allocation de débits avec minimisation de la latence limitée qui peut être considéré comme un problème de flux multiple (MCF Multi Commodity Flow problem).

La relaxation du problème original représenté dans les équations (4.1) à (4.9) est transformée en sous-problème 1 : Problème d'affectation des demandes au coût minimal dans les équations (4.10) à (4.14) et sous-problème 2 : Problème d'allocation de débit avec contrainte de latence

dans les équations (4.15) à 4.19). Plus de détails sur la contrainte (4.6) dans l'algorithme de résolution du problème.

4.4.3.1 Problème d'affectation des demandes au coût minimal

$$\min \sum_{p}^{P} \sum_{f}^{F} \alpha \sum_{s}^{S} x_{fsp} C_{fsp} \tag{4.10}$$

S.t.

$$\sum_{p}^{P} x_{fsp} = 1, \forall s \in S, \forall f \in F \tag{4.11}$$

$$\sum_{s}^{S} x_{fsp}.z_s \leqslant B_p, \forall p \in P, \forall f \in F \tag{4.12}$$

$$C_{fsp} = \frac{Cap_p}{B_p}, \forall p \in P \tag{4.13}$$

$$x_{fsp} \in \{0, 1\}, \forall s \in S, \forall p \in P \tag{4.14}$$

4.4.3.2 Problème d'allocation de débit avec contrainte de latence

$$\min \sum_{p}^{P} \sum_{f}^{F} \beta \exp\left(w_f / \sigma_{fsp}\right) \tag{4.15}$$

S.t.

$$\sum_{s}^{S} \sigma_{fsp} \geqslant \sigma_{min}(w_f), \forall p \in P, \forall f \in F \tag{4.16}$$

$$\sum_{s}^{S} \sigma_{fsp}.x_{fsp} \leqslant B_p, \forall p \in P, \forall f \in F \tag{4.17}$$

$$z_s \geqslant 64KB, \forall s \in S \tag{4.18}$$

$$\sigma_{fsp} \geqslant 0, \forall s \in S, \forall p \in P \tag{4.19}$$

Algorithme 4.1 Scheduling and Rate Allocation (SRA) algorithm

1 **Algorithme :** Scheduling and Rate Allocation (SRA) algorithm
Input : N, L, Cap_l, B_l, F, P, a_{lp}
Output : x_{sfp}, σ_{sfp}

2 $F \leftarrow$ Nouvel ordre des flux en fonction de leurs priorités
3 $z_s \leftarrow$ Valeure initiale
4 $burstsize \leftarrow$ Valeure initiale
5 $x_{sfp} \leftarrow$ Résoudre la relaxation (1) à travers les équations (10-14)
6 Restructurer x_{sfp} afin de l'introduire en entrée à la prochaien étape
7 $\sigma_{fsp} \leftarrow$ Résoudre la relaxation (2) à travers les équations (15-19)
8 **foreach** *sous flux s dans S* **do**
9 | Calculer FCT_s, $FCT_s = \frac{z_s . x_{fsp}}{\sigma_{fsp}}$
10 **end foreach**
11 **foreach** *flux f dans F* **do**
12 | Calculer FCT_f, $FCT_f = max(FCT(s))$
13 | **if** $FCT_f > D_f$ **then**
14 | | Aller à la ligne 1 et répéter
15 | **end if**
16 | **else**
17 | | Débit optimal obtenu Fin
18 | **end if**
19 **end foreach**

Afin de résoudre les deux sous-problèmes précédents, nous suivons les étapes de l'algorithme ci-dessous. Tout d'abord, nous sélectionnons un ordre de transmission pour les flux entrants dans le créneau horaire actuel. Cette étape vise à prioriser les flux de souris avant leurs délais limites (ligne 2 de l'algorithme). Cela n'affectera pas l'aspect en ligne de notre approche, car cela dépend plutôt du module de classification ; module qui fournit l'identification de flux pour les premiers paquets seulement, ce qui est en accord avec le concept en ligne de la planification.

Une étape importante dans l'optimisation des résultats du deuxième sous-problème consiste à ajuster ses hyperparamètres : taille et nombre de sous-flux. En effet, chaque flux entrant sera divisé en plusieurs sous-flux d'une taille prédéfinie, plus tard, ils seront regroupés en rafales pour

éviter la transmission de tous les sous-flux en même temps ce qui provoquera la dégradation des performances des flux de souris.

Après initialisation de certains paramètres, principalement la taille du groupe de sous flux à transférer "rafal" (lignes 3 et 4), le premier sous-problème est résolu (ligne 5) par un solveur de programmation non linéaire et ses résultats représentent l'entrée du second sous-problème (ligne 6). En d'autres termes, la première relaxation produit la stratégie de planification sur laquelle les débits seront calculés au niveau de la seconde relaxation. Il convient de mentionner que la résolution du deuxième sous-problème par un solveur plus puissant (pyomo) nécessite de relaxer la fonction exponentielle dans l'équation (4.15) à l'équation (4.20).

La contrainte (4.6) est mise à jour ultérieurement et ainsi FCT est mesuré (lignes 8 et 9) et est comparé à l'échéance tolérée, si ce délai n'est pas respecté, une autre exécution de l'algorithme est effectuée, sinon la planification est à son état optimal.

$$\min \beta . \sum_{p}^{P} \sum_{f}^{F} 1 + (w_f / \sigma_{fsp}) \tag{4.20}$$

4.5 Conclusion

Nous avons présenté notre modèle mathématique intelligent permettant de réaliser la planification adaptative des flux entrants afin d'assurer la qualité de service requise par chaque flux. Ainsi, l'entrée du module de planification constitue la sortie du module de classification, ce qui permet de caractériser chaque flux afin de déterminer ses besoins, et ce dans un environnement agnostique à l'information où les distributions et tailles des flux sont inconnues.

Contrairement aux modèles existants, notre méthode met en place un module de planification intégrant les deux classes de trafic à savoir les flux de souris ainsi que les flux d'éléphants permettant en conséquence de prendre en considération à la fois la qualité de service des flux, mais aussi la qualité d'expérience des utilisateurs finaux.

Suite à la contrainte temps réel des flux de souris, nous avons construit un modèle de classification et de planification en ligne qui assurent en même temps la minimisation des coûts de traitement tout en évitant la congestion.

Nous avons résolu le modèle mathématique complexe du type MINLP grâce à la relaxation du problème en deux sous problèmes et en proposant un algorithme reliant les solutions des deux sous problèmes en question. Nous testerons dans le chapitre suivant les performances de cet algorithme dans plusieurs topologies et face à certains scénarios. Par la suite, nous comparerons les performances de notre approche à des approches existantes.

CHAPITRE 5

EXPÉRIMENTATION ET RÉSULTATS

5.1 Introduction

Au cours de ce chapitre, nous allons présenter les résultats obtenus en testant la méthode proposée et détaillée au chapitre précédent sur différentes topologies et sous plusieurs contraintes. Nous détaillerons ainsi les scénarios exécutés et comparerons leurs performances face aux méthodes existantes, toujours dans un environnement agnostique à l'information. Nous analysons les résultats obtenus pour évaluer l'efficacité et l'évolutivité de notre approche.

5.2 Protocole expérimental

Le protocole expérimental vise à présenter les différents scénarios de test déployés ainsi que les topologies de validation utilisées.

5.2.1 Scénarios de test

Notre méthode de planification peut être déployée comme une application SDN suivant le module de classification et le module de routage. Le résultat de notre approche est la matrice de taux ou débits qui garantira la QoS et QoE du trafic des différentes classes échangées entre centres de données. Pour la validation de cette approche, nous avons implémenté notre solution en utilisant Python, pour deux typologies (SWAN [18], GÉANT [19]) et pour deux types de scénarios. Le premier est un réseau proche de la congestion et le second est un réseau qui dispose de plus de ressources. Nous avons aussi comparé notre approche à d'autres méthodes existantes (Amoeba, Tempus, Tina) que nous avons détaillées dans les précédents chapitres.

Pour chacun des deux cas (cas sans congestion, cas avec possible congestion), nous avons effectué différents tests où nous ajustons les hyperparamètres de l'algorithme, particulièrement la taille des sous-flux. Nous effectuons aussi une étude sur la taille des burst (rafale) définie comme le nombre de sous-flux dans un rafale et nous analysons le résultat obtenu afin d'évaluer

l'efficacité de notre approche et déterminer le meilleur compromis en termes de performances et paramétrage du modèle.

5.2.2 Topologies de test

Nous avons tester les performances de notre approche sur deux topologies, en premier lieu, SWAN (figure 5.1) une topologie constituée de l'interconnexion d'un nombre limité de nœuds de transport afin d'interconnecter des centres de données distribués. La topologie est gérée par un contrôleur SDN centralisé. Par la suite, GÉANT (figure 5.2), constitué d'une interconnexion à large échelle interconnectant plusieurs villes d'Europe. Notre objectif est ainsi de tester l'extensibilité de notre approche et son efficacité dans un environnement stable et non stable (sujet aux congestions ou pas), et pour une topologie à petite échelle et aussi à large échelle où le temps et les coûts et le traitement deviennent plus critiques.

5.2.3 Implementation

Afin de résoudre notre problème de planification en ligne, nous exécutons notre algorithme de planification agnostique à l'information avec un solver puissant prénommé pyomo. Il peut être décrit comme un outil regroupant plusieurs solvers (CPLEX, IPOPT, AMPL, GLPK, PICO, CBC) permettant ainsi de disposer d'une panoplie de solvers sous le même environnement de programmation.

Pyomo est implémenté dans l'outil Spider du logiciel de programmation Python Anaconda. La machine utilisée pour l'exécution de pyomo est de type Intel(R) Core(TM) i7-6500U CPU avec une puissance 2.50GHz et une mémoire vive de 8.00 GB.

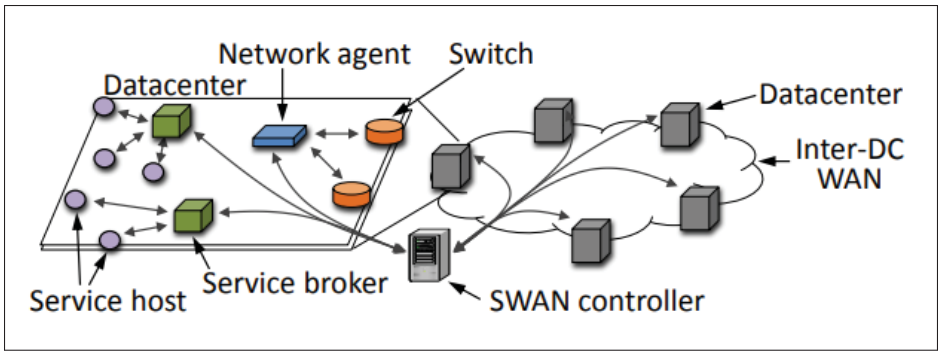


Figure 5.1 Topologie du réseau SWAN

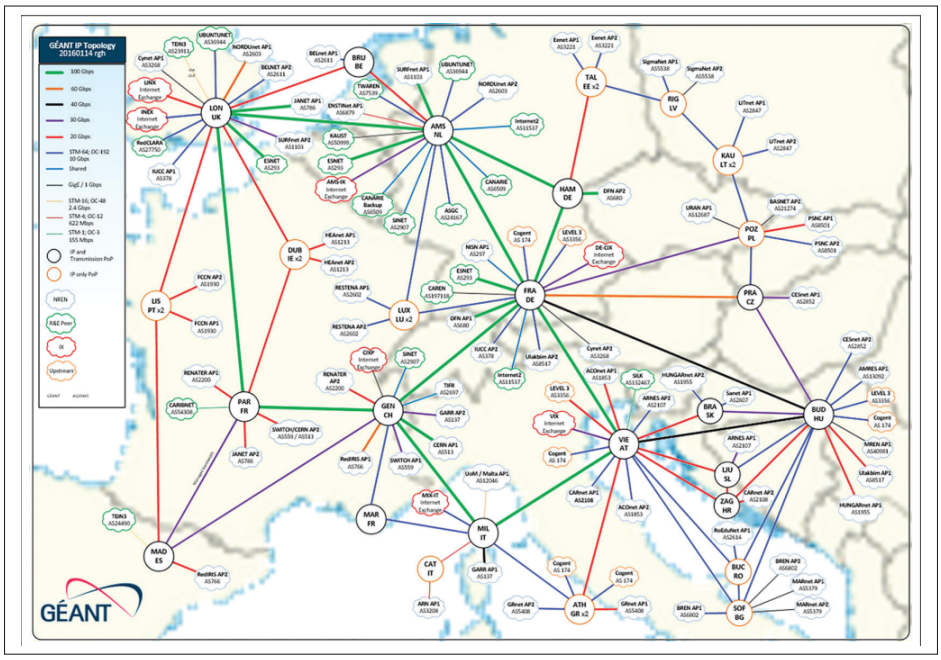


Figure 5.2 Topologie du réseau GEANT

5.2.4 Scénarios des expérimentations et analyses des performances

5.2.4.1 Scénarios pour topologie SWAN

5.2.4.1.1 Scénarios sans congestion avec différents ensembles de flux

Selon Xu & Li (2014); Zhang *et al.* (2015), moins le réseau est encombré, plus la taille du flowlet (sous-flux) est grande. Pour notre premier scénario, nous déployons un réseau moins encombré et nous faisons varier la taille de chaque sous-flux de 64 Ko à 625 Ko. Les valeurs des tailles de sous-flux sont de 64Ko (0,0005 Gbits), 120Ko (0,001 Gbits), 300Ko (0,002 Gbits), 500Ko (0,004 Gbits), 625Ko (0,005 Gbits).

Plusieurs flux de souris et d'éléphants entrent dans le réseau. Nous analysons d'abord les performances de notre approche, avec un trafic mixte des deux classes, puis pour un trafic avec plus de gros flux (éléphant), et observons comment notre algorithme s'adapte à cette variabilité (figures 5.3 et 5.4).

Dans la figure 5.3, nous représentons la première expérience : un nombre mixte de souris et d'éléphants dans un réseau avec presque aucune congestion possible pour le moment. Les plus petits FCT représentent les flux de souris (les pics inférieurs) tandis que les valeurs de FCT plus élevées caractérisent les flux d'éléphants, ce qui est logique en raison des exigences de chaque classe. Pour commencer, nous remarquons que les valeurs FCT doublent lorsque la taille du sous-flux double, et ce pour les deux classes et que comparativement au FCT des flux souris, celui des flux éléphant est stable pour tous les sous-flux.

Cette observation n'est pas valable dans la seconde expérience où plus de flux d'éléphants sont échangés comme le montre la figure 5.4. Le FCT du flux de souris est toujours à la même valeur (0,1 s), contrairement au FCT du flux d'éléphants où nous remarquons une variation non négligeable à mesure que la taille des sous-flux augmente. Cette fluctuation peut apparaître comme un problème de réorganisation des paquets, que nous pouvons omettre, car tant qu'elle n'affecte pas les flux de souris, les flux d'éléphants tolèrent les retards.

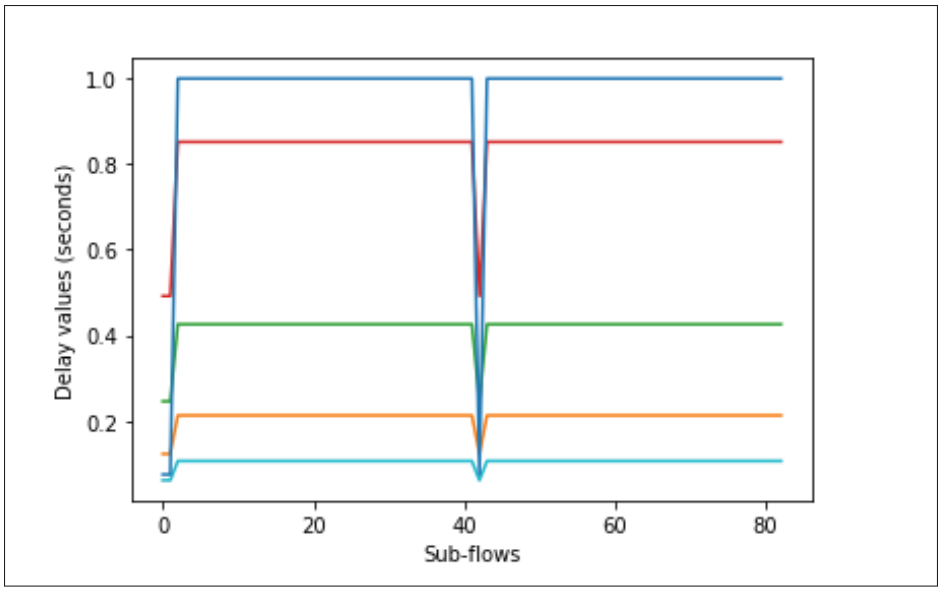


Figure 5.3 Réseau sans congestion avec un nombre mixte d'éléphants et de souris

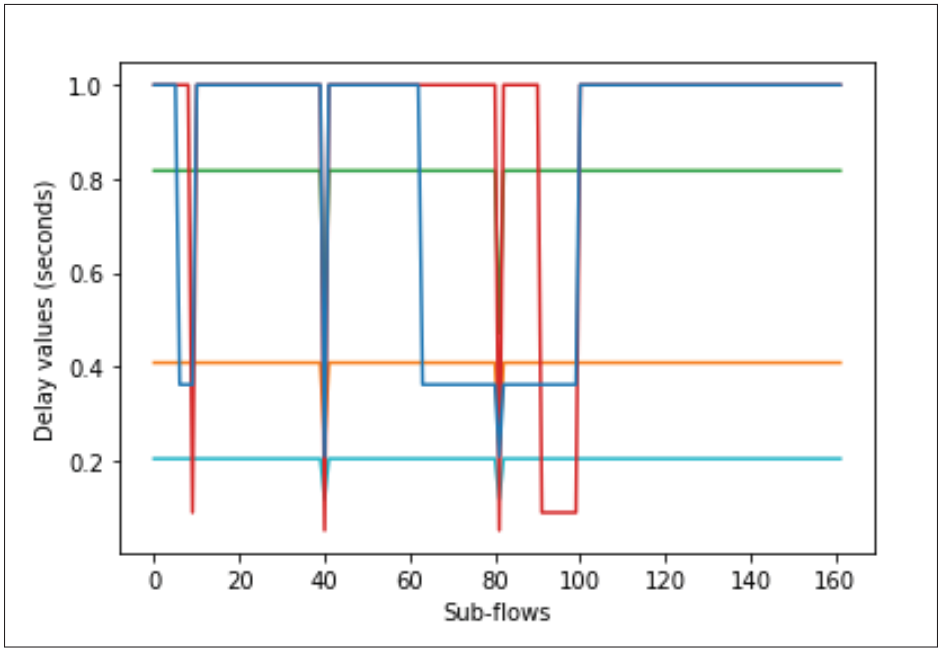


Figure 5.4 Réseau sans congestion avec plus de flux d'éléphants

5.2.4.1.2 Scénarios sans congestion avec variation de la taille des rafales

Pour les trois scénarios suivants, nous faisons varier le nombre de sous-flux dans chaque burst ou rafale de 50 dans les expériences précédentes à 40, 30 et 15 dans les figures 5.5, 5.6, 5.7.

Avec 40 sous-flux par rafale, les meilleurs résultats sont obtenus pour le sous-flux de taille 64 Ko, suivi de 300 Ko puis 120 Ko, à la différence avec les délais des flux d'éléphant pour qui les sous-flux de 300 Ko sont moins stables que ceux avec 120 Ko. Les sous-flux de 625 Ko et de 500 Ko fournissent un délai de flux de souris court, mais des délais de flux d'éléphants très élevés, donc en fonction des délais maximums et de la tolérance QoE pour les flux d'éléphants, nous sélectionnerons le scénario le plus adéquat.

Avec 30 sous-flux dans chaque rafale de flux d'éléphants, nous remarquons pour presque tous les scénarios une diminution du FCT des flux de souris et plus de stabilité pour les valeurs des FCT des flux éléphants. Ceci est confirmé par la dernière expérience (15 sous-flux).

Notre approche consiste à itérer sur les flux en sélectionnant différents hyperparamètres afin de minimiser FCT au délai requis et sélectionner ainsi le délai associé. Dans la plupart des scénarios, le FCT optimal est obtenu avec 64 Ko, donc même si selon Xu & Li (2014); Zhang *et al.* (2015), nous pouvons sélectionner des valeurs plus élevées parce qu'il n'y a pas de congestion, il est préférable d'adapter la taille des rafale, sinon cela affectera négativement les valeurs FCT.

5.2.4.1.3 Comparaison entre notre approche et celles existantes

Nous comparons notre approche à Ameoba, Tempus et Tina. Dans cette expérience, trois flux sont échangés [Flow1, Flow2, Flow3], tandis que le premier flux arrivant est un éléphant, les flux suivants sont des flux de souris. Suivant notre approche, nous sélectionnons l'ordre de flux suivant : [Flow3, Flow2, Flow1]. Le flux éléphants sera divisé en 4 rafales de 40 sous-flux de taille 64Ko, pour un total de 160 sous-flux. Nous avons fixé le délai du flux souris à 200ms et le délai de flux d'éléphant à 800ms. Dans la figure de résultat (figure 5.8), nous utilisons

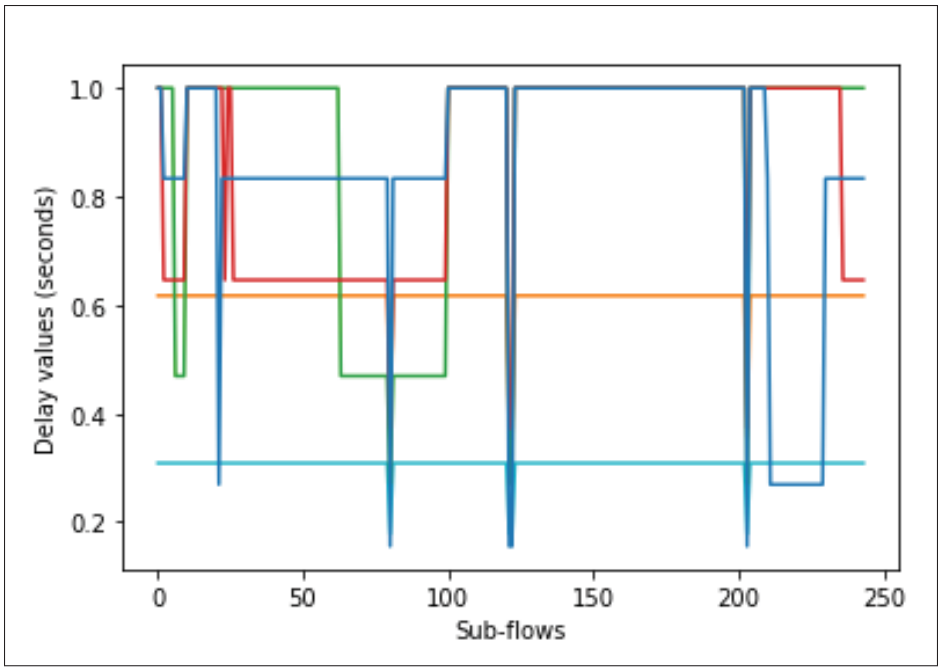


Figure 5.5 Réseau sans congestion avec 40 sous-flux par rafale

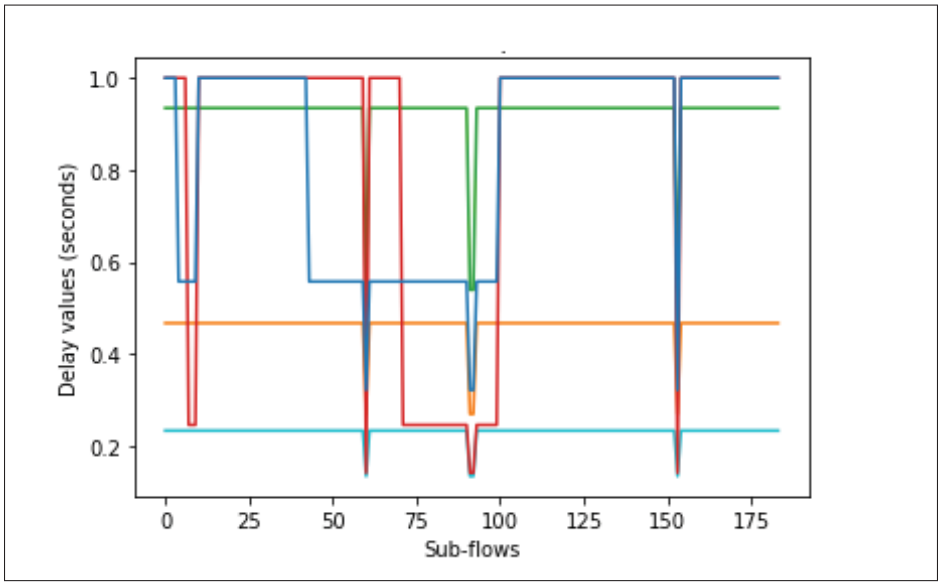


Figure 5.6 Réseau sans congestion avec 30 sous-flux par rafale

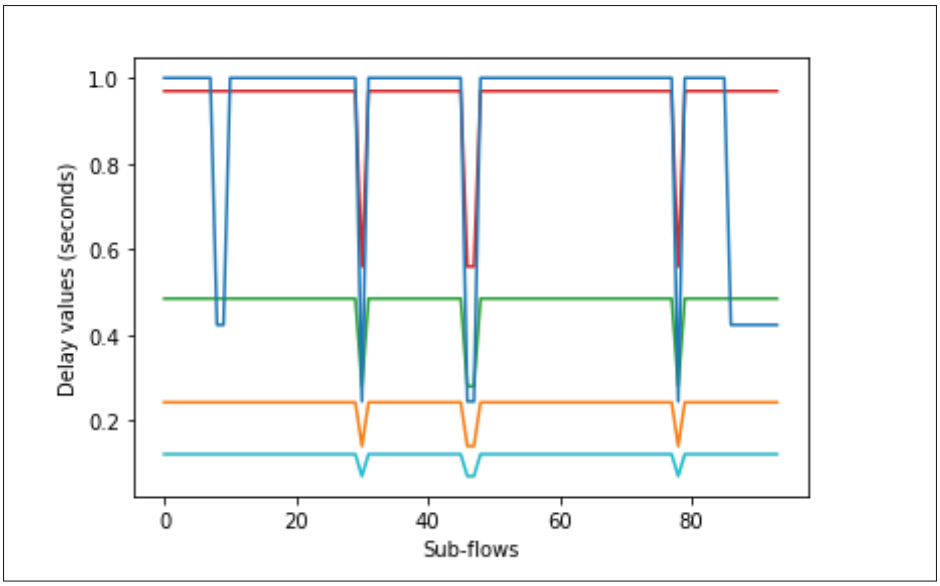


Figure 5.7 Réseau sans congestion avec 15 sous-flux par rafale

l'acronyme MF et EF pour Mice Flow et flux Éléphant Flow, tandis que SF est un acronyme pour Sub-Flow.

Pour Tempus et Ameoba, une partie importante du trafic est perdue. Dans le cas d'Ameoba, la dernière partie du flux éléphants est perdue, mais à cause de la politique du tout ou rien de cette approche, tout le flux éléphants sera perdu. En ce qui concerne Tempus, comme il maximise la quantité de trafic transmis. Ainsi, le flux éléphants est accepté alors que les flux souris sont rejetés (exigence de délai non respectée). Enfin, les pires résultats sont obtenus avec Tina, où un seul flux souris est accepté. Le flux éléphant ne sera pas sélectionné pour la transmission parce qu'il compromettra le retard du flux souris et le second flux souris n'est pas non plus sélectionné parce qu'il contredit les critères d'équité de Tina.

Contrairement aux méthodes précédentes, notre approche différencie les classes de trafic et procède sur plusieurs itérations pour compléter efficacement le flux. Cette approche peut également être considérée comme totalement agnostique, donc outre le fait d'être agnostique de taille et de distribution, nous pouvons toujours ajuster notre algorithme au plus petit FCT possible sans la disponibilité des délais.

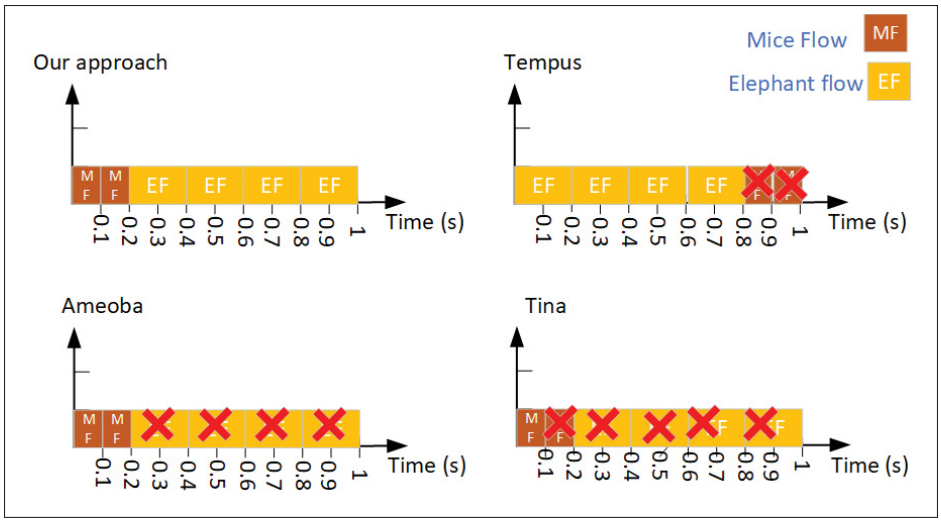


Figure 5.8 Comparaison de notre approches de planification avec Tempus, Ameoba et Tina

5.2.4.1.4 Scénarios prenant en considération la congestion

Dans les scénarios suivants, nous étudions le comportement de notre approche dans un réseau où les ressources sont moins disponibles (congestion possible) figure 5.9. Nous résumons les valeurs FCT pour les cinq tailles de sous-flux en présence de congestion ou non dans le tableau ci-dessous. Nous remarquons que notre algorithme s'adapte à la disponibilité de la bande passante grâce à la planification multichemin entre les chemins disponibles, ce qui conduit à de meilleures performances réseau. De plus, à partir du tableau 5.1, nous observons que les sous-flux sont bien gérés dans un environnement conscient de la congestion, jusqu'à une certaine taille de sous-flux considérée trop grande. Une meilleure façon de gérer cette limitation est de diviser les grands flux en sous-flux plus petits puisque, même si certains arrivent tard, c'est insignifiant, car ils appartiennent à la classe des flux d'éléphants.

5.2.5 Test avec topologie Géant

Nous étudions les performances et la extensibilité de notre approche de planification en mesurant le FCT des souris et des flux éléphants pour le réseau Géant 2, qui en plus de connecter plus de nœuds que Microsoft Swan, les chemins de routage sont extrêmement liés, ce qui met encore

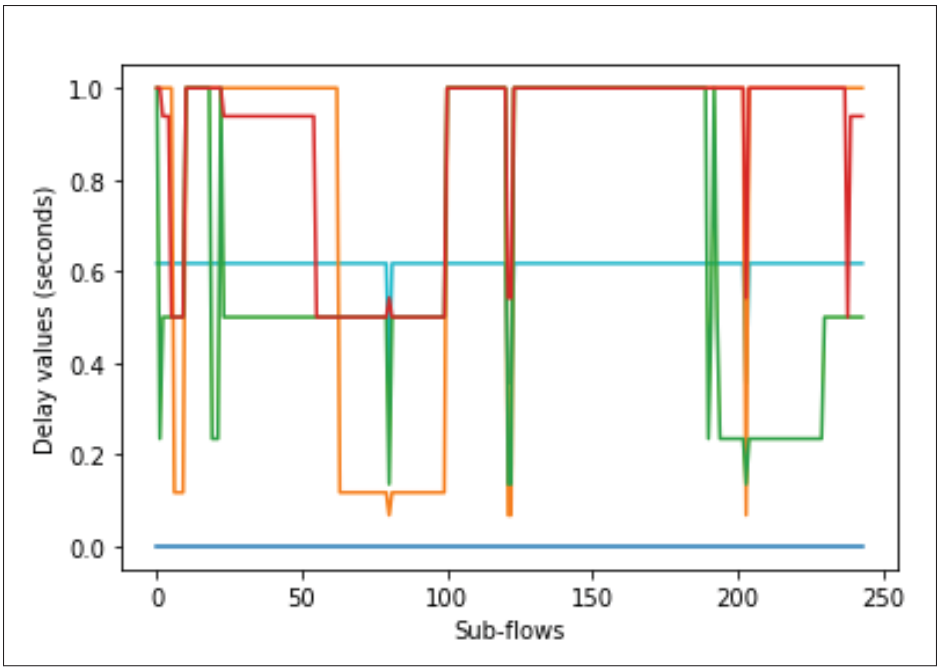


Figure 5.9 Scénarios conscients de la congestion

Tableau 5.1 Comparaison les scenarios avec et sans congestion pour SWAN

	Sans Congestion		Avec Congestion	
Sub-flow size	Mice	Elephant	Mice	Elephant
64 KB	0.18	0.32	0.35	0.65
120 KB	0.35	0.62	0.1	[0.15-1]
300 KB	0.3	[o.5-1]	0.1	[0.2-1]
500 KB	0.35	[o.65-1]	0.5	[0.5-1]
625 KB	0.18	[o.3-1]	Non	applicable

plus l'accent sur la congestion en cas de limitation des ressources. Nous sélectionnons deux nœuds comme source et destination pour l'expérience et construisons une matrice de routage à partir de quatre chemins et 13 liens. Les résultats sont représentés dans la figure 5.10.

Nous remarquons pour le premier scénario (64 Ko), un FCT plus stable pour les flux éléphants, mais le FCT optimal pour le flux souris n'est pas observé pour ce scénario. En fait, alors que 0,35 s est mesurée pour ce scénario, 0,15 est mesuré avec 120 Ko et 0,3 pour 625 Ko. Nous concluons que la planification multichemins constitue l'une des forces de notre approche

puisqu'elle optimise les ressources en bande passante afin d'améliorer le FCT particulièrement pour les flux souris, mais aussi pour les flux éléphants, en opposition à d'autres approches se focalisant sur les flux souris (sensibles au temps) et donc la QoS. Dans notre approche, nous considérons également le flux éléphants garantissant ainsi à la fois la QoS et la QoE.

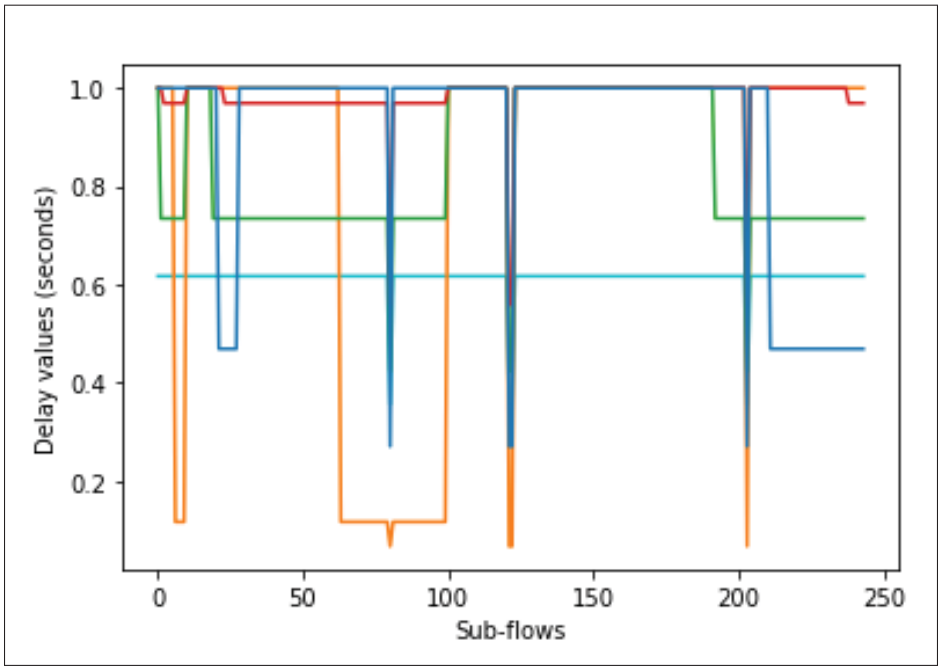


Figure 5.10 Notre approche pour le réseau Geant 2

5.3 Conclusion

Nous avons présenté les tests effectués afin de valider notre approche dans un environnement avec et sans congestion. Notre algorithme a prouvé sa robustesse face à la congestion, et a aussi démontré une amélioration importante comparativement aux méthodes existantes. Les résultats obtenus nous ont permis d'identifier les atouts de notre approche, entre autres son efficacité dans un contexte complètement agnostique et en présence de données déséquilibrées. En conclusion, nous pouvons affirmer avoir rempli les objectifs de cette thèse annoncés au premier chapitre, tout en soulignant la possibilité d'améliorer certains aspects dans des travaux futurs, tels que les temps de traitement, ou encore le type de plateforme d'implémentation.

CONCLUSION ET RECOMMANDATIONS

Afin de garantir les meilleures performances du réseau et satisfaire les exigences des contrats de service (SLA), il est important de mettre en place des mécanismes de routage efficaces jumelés à des approches de planification optimales. La plupart des travaux précédents nécessitent une description détaillée des données d'entrée, telles que la taille totale des flux, leurs tailles restantes pendant le processus de planification, la distribution du trafic, etc. Ce scénario où la disponibilité de ces informations est automatique et évidente, ne relève pas de la réalité des plateformes actuelles et ne peut satisfaire certains critères tels que l'aspect temps réel des applications implémentées.

À travers ce mémoire, nous avons construit un module de planification adoptant une approche à l'écoute des divers besoins des classes de trafic dans un environnement agnostique à l'information. Nous nous sommes également concentrés sur l'optimisation des coûts de traitement dans un réseau interconnectant les centres de données en prenant en considération les Contraintes liées à la congestion des chemins de routage.

Le principal défi réside dans l'incertitude causée par l'aspect indépendant de l'information du problème. Le module de classification vise justement à outrepasser ce défi dans la représentation des données échangées entre les centres de données. En effet, dans un environnement agnostique à l'information, il est important d'instaurer un moyen permettant d'exprimer clairement les besoins des flux entrants afin de satisfaire les demandes.

Comparé aux approches de planification existante, notre méthode ne se concentre pas uniquement sur une seule classe de trafic, elle est plutôt multiclasse et représente les deux classes de souris et d'éléphants. Il s'agit d'une approche de planification en ligne puisque le module de classification est exécuté en ligne, et la partie la plus gourmande en temps de notre méthode est le réglage (même si cela reste assez négligeable).

Notre méthode s'adapte à la congestion du réseau grâce à un réglage minutieux des hyperparamètres, mais aussi à une approche de planification multichemins. Enfin, elle peut également s'appliquer au cas d'utilisation totalement agnostique, où le délai est également inconnu puisque plus le flux est faible, plus le délai est court.

Concernant notre module de préplanification, nous avons proposé une nouvelle approche de classification combinant l'efficacité des méthodes sensibles aux coûts et la robustesse des approches d'ensembles par le biais d'une stratégie de corrélation interflux. Après avoir formé plusieurs forêts aléatoires sur les sacs de trafic de données, nous utilisons l'algorithme RkNN pour mesurer la corrélation entre les échantillons de test représentant les flux de trafic et chaque classificateur de type bagging afin de calculer les poids de rééquilibrage.

C'est une nouvelle combinaison visant à surmonter les défis du trafic déséquilibré tout en respectant les contraintes de temps. Pour évaluer notre approche, nous faisons varier les hyperparamètres de notre algorithme et comparons ses performances sur les traces de trafic des centres de données à d'autres approches pour la classification du trafic déséquilibré. Les résultats obtenus dans plusieurs environnements de test démontrent que notre approche de classification basée sur la corrélation surpasse les autres algorithmes en termes de précision, de rappel et de mesure F1 en rééquilibrant l'ensemble selon une méthode sensible aux coûts qui évite l'utilisation d'une matrice de coûts de classification erronée. Notre algorithme équilibre le compromis entre la précision et le rappel et produit une matrice de confusion avec de bons ratios FP et FN qui empêchent le surprovisionnement ainsi que la congestion.

Dans de futurs travaux de recherche, nous prévoyons considérer un nombre d'approches réaliser certains changements afin d'améliorer les performances de notre approche de planification basée sur la classification.

Nous proposons de nous concentrer sur le temps de calcul de l'algorithme RkNN, en remplaçant la stratégie arborescente par une approche de regroupement (clustering) plus extensible. En outre, nous inclurons des paquets UDP, qui permettront de généraliser notre approche pour des trafics plus riches et améliorent la robustesse de notre approche proposée par rapport à des ensembles de données de trafic plus riches, nous devons ainsi surmonter les défis de la construction de la base de données de validation (ground truth).

Par ailleurs, nous prévoyons d'étendre notre approche de planification à une topologie multicouche pour intégrer différentes sources de bande passante (sur plusieurs couches) et par conséquent améliorer encore plus les performances en termes de délais et de coûts. Nous proposerons également une nouvelle méthode pour résoudre le problème de planification agnostique de l'information couplé avec un module de contrôle d'admission de flux inspiré par la théorie ruin Manzoor *et al.* (2023), le tout basé sur des heuristiques puisque les dimensions du problème augmenteront.

BIBLIOGRAPHIE

Abbasloo, S., Xu, Y. & Chao, H. J. (2018). Hyline : a simple and practical flow scheduling for commodity datacenters. *2018 IFIP Networking Conference (IFIP Networking) and Workshops*, pp. 1–9.

Abbasloo, S., Xu, Y. & Chao, H. J. (2020). To schedule or not to schedule : when no-scheduling can beat the best-known flow scheduling algorithm in datacenter networks. *Computer Networks*, 107177.

Ali, Z. & Bhaskar, S. B. (2016). Basic statistical tools in research and data analysis. *Indian journal of anaesthesia*, 60(9), 662.

Amina, S. S. M., Abdolkhalegh, B., Khoa, N. K. & Mohamed, C. (2018). Featuring Real-Time imbalanced network traffic classification. *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 840–846.

Angiulli, F. (2017). On the Behavior of Intrinsically High-Dimensional Spaces : Distances, Direct and Reverse Nearest Neighbors, and Hubness. *Journal of Machine Learning Research*, 18, 170–1.

Bai, W., Chen, L., Chen, K., Han, D., Tian, C. & Wang, H. (2017). Pias : Practical information-agnostic flow scheduling for commodity data centers. *IEEE/ACM Transactions on Networking*, 25(4), 1954–1967.

Barewar, A., Radke, M. A. & Deshpande, U. A. (2014). Geo skip list data structure-storing spatial data and efficient search of geographical locations. *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 1479–1485.

Benson, T. (2010). Data Set for IMC 2010 Data Center Measurement. Repéré à http://cs.brown.edu/~tab/.

Benson, T., Akella, A. & Maltz, D. A. (2010). Network traffic characteristics of data centers in the wild. *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pp. 267–280.

Bisht, J. & Subrahmanyam, V. (2021). Survey on Load Balancing and Scheduling Algorithms in Cloud Integrated Fog Environment. *ICIDSSD 2020*, 466.

CAIDA. (2011). CAIDA Anonymized 2011 Internet Traces Dataset. Repéré à http://www.caida.org/data/monitors/passive-equinix-chicago.xml.

Casanova, G., Englmeier, E., Houle, M. E., Kröger, P., Nett, M., Schubert, E. & Zimek, A. (2017). Dimensional testing for reverse k-nearest neighbor search. *Proceedings of the VLDB Endowment*, 10(7), 769–780.

Chao, S.-C., Lin, K. C.-J. & Chen, M.-S. (2016). Flow classification for software-defined data centers using stream mining. *IEEE Transactions on Services Computing*, 12(1), 105–116.

Chen, F., Wu, C., Hong, X., Lu, Z., Wang, Z. & Lin, C. (2016). Engineering traffic uncertainty in the openflow data plane. *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9.

Collell, G., Prelec, D. & Patil, K. R. (2018). A simple plug-in bagging ensemble based on threshold-moving for classifying binary and multiclass imbalanced data. *Neurocomputing*, 275, 330–340.

Dawar, S., Goyal, V. & Bera, D. (2015). Efficient Reverse k Nearest Neighbor evaluation for hierarchical index. *arXiv preprint arXiv :1506.04867*.

Del Río, S., López, V., Benítez, J. M. & Herrera, F. (2014). On the use of MapReduce for imbalanced big data using Random Forest. *Information Sciences*, 285, 112–137.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan), 1–30.

Ding, L., Liu, J., Qin, T. & Li, H. (2017). Internet traffic classification based on expanding vector of flow. *Computer Networks*, 129, 178–192.

Dong, H., Munir, A., Tout, H. & Ganjali, Y. (2021). Next-Generation Data Center Network Enabled by Machine Learning : Review, Challenges, and Opportunities. *IEEE Access*.

Dong, X.-D., Chen, S., Zhao, L.-P., Zhou, X.-B., Qi, H. & Li, K.-Q. (2018). More requests, less cost : Uncertain inter-datacenter traffic transmission with multi-tier pricing. *Journal of Computer Science and Technology*, 33(6), 1152–1163.

Dong, X. & Cai, B. (2022). Slardar : Scheduling information incomplete inter-datacenter deadline-aware coflows with a decentralized framework. *Computer Networks*, 214, 109178.

Dong, X., Zhao, L., Zhou, X., Li, K., Guo, D. & Qiu, T. (2019). An Online Cost-Efficient Transmission Scheme for Information-Agnostic Traffic in Inter-Datacenter Networks. *IEEE Transactions on Cloud Computing*, 10(1), 202–215.

Dong, X., Li, W., Zhou, X., Li, K. & Qi, H. (2020). TINA : A Fair Inter-datacenter Transmission Mechanism with Deadline Guarantee. *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pp. 2017–2025.

Doroud, H., Aceto, G., de Donato, W., Jarchlo, E. A., Lopez, A. M., Guerrero, C. D. & Pescape, A. (2018). Speeding-Up DPI Traffic Classification with Chaining. *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6.

Fernández, A., del Río, S., Chawla, N. V. & Herrera, F. (2017). An insight into imbalanced big data classification : outcomes and challenges. *Complex & Intelligent Systems*, 3, 105–120.

Gandhi, S. et al. (2020). Vritti : Guarantees for Mix Flows in Inter-Datacenter Networks.

Gerstel, O., Lopez, V. & Siracusa, D. (2015). Multi-layer orchestration for application-centric networking. *2015 International Conference on Photonics in Switching (PS)*, pp. 318–320.

Gómez, S. E., Martínez, B. C., Sánchez-Esguevillas, A. J. & Callejo, L. H. (2017a). Ensemble network traffic classification : Algorithm comparison and novel ensemble scheme proposal. *Computer Networks*, 127, 68–80.

Gómez, S. E., Martínez, B. C., Sánchez-Esguevillas, A. J. & Callejo, L. H. (2017b). Ensemble network traffic classification : Algorithm comparison and novel ensemble scheme proposal. *Computer Networks*, 127, 68–80.

Gómez, S. E., Hernández-Callejo, L., Martínez, B. C. & Sánchez-Esguevillas, A. J. (2019). Exploratory study on class imbalance and solutions for network traffic classification. *Neurocomputing*, 343, 100–119.

Hasibi, R., Shokri, M. & Dehghan, M. (2019). Augmentation scheme for dealing with imbalanced network traffic classification using deep learning. *arXiv preprint arXiv :1901.00204*.

Hu, C., Liu, B., Xing, C., Yue, Z., Song, L. & Chen, M. (2016). Queueing model based analysis on flow scheduling in information-agnostic datacenter networks. *2016 IEEE International Conference on Communications (ICC)*, pp. 1–6.

Hu, W., Liu, J., Huang, T. & Liu, Y. (2018). A completion time-based flow scheduling for inter-data center traffic optimization. *IEEE Access*, 6, 26181–26193.

Hu, Y., Tian, J. & Ma, J. (2021). A novel way to generate adversarial network traffic samples against network traffic classification. *Wireless Communications and Mobile Computing*, 2021(1), 7367107.

Huang, Y., Li, Y. & Qiang, B. (2016). Internet traffic classification based on min-max ensemble feature selection. *Neural Networks (IJCNN), 2016 International Joint Conference on*, pp. 3485–3492.

Iqbal, M. F., Zahid, M., Habib, D. & John, L. K. (2019). Efficient Prediction of Network Traffic for Real-Time Applications. *Journal of Computer Networks and Communications*, 2019.

Jain, S., Kotsampasakou, E. & Ecker, G. F. (2018). Comparing the performance of meta-classifiers—a case study on selected imbalanced data sets relevant for prediction of liver toxicity. *Journal of computer-aided molecular design*, 32(5), 583–590.

Jalaparti, V., Bliznets, I., Kandula, S., Lucier, B. & Menache, I. (2016). Dynamic pricing and traffic engineering for timely inter-datacenter transfers. *Proceedings of the 2016 ACM SIGCOMM Conference*, pp. 73–86.

Joe-Wong, C., Kamitsos, I. & Ha, S. (2015). Interdatacenter job routing and scheduling with variable costs and deadlines. *IEEE Transactions on Smart Grid*, 6(6), 2669–2680.

Kandula, S., Menache, I., Schwartz, R. & Babbula, S. R. (2014). Calendaring for wide area networks. *Proceedings of the 2014 ACM conference on SIGCOMM*, pp. 515–526.

Kiran, M. & Chhabra, A. (2019). Understanding flows in high-speed scientific networks : A Netflow data study. *Future Generation Computer Systems*, 94, 72–79.

Kulkarni, V. Y., Sinha, P. K. & Petare, M. C. (2016). Weighted hybrid decision tree model for random forest classifier. *Journal of The Institution of Engineers (India) : Series B*, 97, 209–217.

Lei, K., Li, K., Xing, J., Jin, B. & Wang, Y. (2018). Distributed Information-Agnostic Flow Scheduling in Data Centers Based on Wait-Time. *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7.

Li, W., Abdin, K., Dann, R. & Moore, A. (2013). *Approaching real-time network traffic classification*.

Li, W., Li, K., Guo, D., Min, G., Qi, H. & Zhang, J. (2016). Cost-minimizing bandwidth guarantee for inter-datacenter traffic. *IEEE Transactions on Cloud Computing*.

Li, W., Zhou, X., Li, K., Qi, H. & Guo, D. (2018). TrafficShaper : Shaping Inter-Datacenter Traffic to Reduce the Transmission Cost. *IEEE/ACM Transactions on Networking (TON)*, 26(3), 1193–1206.

Liu, B., Xing, C., Hu, C., Yu, P. et al. (2016a). Flow Scheduling Strategies for Minimizing Flow Completion Times in Information-agnostic Data Center Networks. *Journal of Network Computing and Applications*, 1(1), 12–20.

Liu, Z., Wang, R. & Tao, M. (2016b). SmoteAdaNL : a learning method for network traffic classification. *Journal of Ambient Intelligence and Humanized Computing*, 7(1), 121–130.

Loo, H. R., Joseph, S. B. & Marsono, M. N. (2016). Online incremental learning for high bandwidth network traffic classification. *Applied Computational Intelligence and Soft Computing*, 2016, 1.

Luo, L., Yu, H., Ye, Z. & Du, X. (2018). Online deadline-aware bulk transfer over inter-datacenter wans. *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pp. 630–638.

Manzoor, A., Alsenwi, M., Tun, Y. K., Saad, W., Hong, C. S. et al. (2023). Ruin theory for user association and energy optimization in multi-access edge computing. *IEEE Transactions on Vehicular Technology*, 72(9), 12436–12440.

Martinez-Ríos, E., Montesinos, L., Alfaro-Ponce, M. & Pecchia, L. (2021). A review of machine learning in hypertension detection and blood pressure estimation based on clinical and physiological data. *Biomedical Signal Processing and Control*, 68, 102813.

McHugh, M. L. (2012). Interrater reliability : the kappa statistic. *Biochemia medica : Biochemia medica*, 22(3), 276–282.

Nguyen, T. T. & Armitage, G. (2008). A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys & Tutorials*, 10(4), 56–76.

Nguyen, T. T., Armitage, G., Branch, P. & Zander, S. (2012). Timely and continuous machine-learning-based classification for interactive IP traffic. *IEEE/ACM Transactions on Networking (TON)*, 20(6), 1880–1894.

N.M, M., Raghavendra, C. S. & Rao, S. (2016a). Dcroute : Speeding up inter-datacenter traffic allocation while guaranteeing deadlines. *2016 IEEE 23rd International Conference on High Performance Computing (HiPC)*, pp. 82–90.

N.M, M., Raghavendra, C. S., Rao, S. & Madni, A. M. (2016b). Rcd : Rapid close to deadline scheduling for datacenter networks. *2016 World Automation Congress (WAC)*, pp. 1–6.

Noormohammadpour, M. & Raghavendra, C. S. (2017). Datacenter traffic control : Understanding techniques and tradeoffs. *IEEE Communications Surveys & Tutorials*, 20(2), 1492–1525.

Peng, L., Zhang, H., Yang, B. & Chen, Y. (2014). A new approach for imbalanced data classification based on data gravitation. *Information Sciences*, 288, 347–373.

Peng, L., Zhang, H., Chen, Y. & Yang, B. (2017). Imbalanced traffic identification using an imbalanced data gravitation-based classification model. *Computer Communications*, 102, 177–189.

Rastegarfar, H., Glick, M., Viljoen, N., Yang, M., Wissinger, J., LaComb, L. & Peyghambarian, N. (2016). TCP flow classification and bandwidth aggregation in optically interconnected data center networks. *Journal of Optical Communications and Networking*, 8(10), 777–786.

Saber, M. A. S., Ghorbani, M., Bayati, A., Nguyen, K.-K. & Cheriet, M. (2020). Online data center traffic classification based on inter-flow correlations. *IEEE Access*, 8, 60401–60416.

Shi, H., Li, H., Zhang, D., Cheng, C. & Cao, X. (2018). An Efficient Feature Generation Approach based on Deep Learning and Feature Selection Techniques for traffic classification. *Computer Networks*.

Shilbayeh, S. A. et al. (2015). *Cost sensitive meta-learning*. (Thèse de doctorat, University Of Salford).

Subedi, T. N. (2018). *SDN-based traffic engineering in data centers, Interconnects, and Carrier Networks*. (Thèse de doctorat, École de technologie supérieure).

Sun, D., Zhao, K., Fang, Y. & Cui, J. (2018). Dynamic Traffic Scheduling and Congestion Control across Data Centers Based on SDN. *Future Internet*, 10(7), 64.

Sun, Y., Kamel, M. S., Wong, A. K. & Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern recognition*, 40(12), 3358–3378.

Tabatabaei, T. S., Karray, F. & Kamel, M. (2012). Early internet traffic recognition based on machine learning methods. *Electrical & Computer Engineering (CCECE), 2012 25th IEEE Canadian Conference on*, pp. 1–5.

Tang, F., Li, L., Barolli, L. & Tang, C. (2017). An efficient sampling and classification approach for flow detection in SDN-based big data centers. *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, pp. 1106–1115.

Tao, Y., Papadias, D., Lian, X. & Xiao, X. (2007). Multidimensional reverse kNN search. *The VLDB Journal*, 16(3), 293–316.

Truong-Huu, T., Gurusamy, M. & Girisankar, S. T. (2017). Dynamic flow scheduling with uncertain flow duration in optical data centers. *IEEE access*, 5, 11200–11214.

UNIBS. (2009). UNIBS : Data Sharing. Repéré à http://netweb.ing.unibs.it/~ntw/tools/traces/index.php.

Wang, J. H., Wang, J., An, C. & Zhang, Q. (2019). A survey on resource scheduling for data transfers in inter-datacenter WANs. *Computer Networks*, 161, 115–137.

Wang, S., Zhang, J., Huang, T., Pan, T., Liu, J. & Liu, Y. (2020). Improving Flow Scheduling Scheme With Mix-Traffic in Multi-Tenant Data Centers. *IEEE Access*, 8, 64666–64677.

Wang, T., Xu, H. & Liu, F. (2017). Aemon : Information-agnostic mix-flow scheduling in data center networks. *Proceedings of the First Asia-Pacific Workshop on Networking*, pp. 106–112.

Wu, D., Chen, X., Chen, C., Zhang, J., Xiang, Y. & Zhou, W. (2015a). On addressing the imbalance problem : a correlated knn approach for network traffic classification. *International Conference on Network and System Security*, pp. 138–151.

Wu, Y., Zhang, Z., Wu, C., Guo, C., Li, Z. & Lau, F. C. (2015b). Orchestrating bulk data transfers across geo-distributed datacenters. *IEEE Transactions on Cloud Computing*, 5(1), 112–125.

Xie, W., Wang, F., Hua, Y., Feng, D., Hu, Y. & Li, C. (2017). A cost-efficient scheme with decoupling host-side flow scheduling from switches in DCNs. *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*, pp. 1–6.

Xie, W., Wang, F., Feng, D., Zhang, L., Zhu, T. & Shi, Q. (2018). Host-based scheduling : Achieving near-optimal transport for datacenter networks. *Computer Networks*, 143, 49–61.

Xu, H. & Li, B. (2014). TinyFlow : Breaking elephants down into mice in data center networks. *2014 IEEE 20th International Workshop on Local & Metropolitan Area Networks (LANMAN)*, pp. 1–6.

Xu, X., Li, W., Qi, H., Wang, J. & Li, K. (2020). Latency-Constrained Cost-Minimized Request Allocation for Geo-Distributed Cloud Services. *IEEE Open Journal of the Communications Society*, 1, 125–132.

Yan, D., Zhao, Z. & Ng, W. (2012). Monochromatic and bichromatic reverse nearest neighbor queries on land surfaces. *Proceedings of the 21st ACM international conference on Information and knowledge management*, pp. 942–951.

Yang, Z., Cui, Y., Wang, X., Liu, Y., Li, M. & Zhang, Z. (2019). Towards maximal service profit in geo-distributed clouds. *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 442–452.

Zhang, H., Tang, F. & Barolli, L. (2019). Efficient flow detection and scheduling for SDN-based big data centers. *Journal of Ambient Intelligence and Humanized Computing*, 10(5), 1915–1926.

Zhang, H., Chen, K., Bai, W., Han, D., Tian, C., Wang, H., Guan, H. & Zhang, M. (2015). Guaranteeing deadlines for inter-datacenter transfers. *Proceedings of the Tenth European Conference on Computer Systems*, pp. 1–14.

Zhang, H., Chen, K., Bai, W., Han, D., Tian, C., Wang, H., Guan, H. & Zhang, M. (2016). Guaranteeing deadlines for inter-data center transfers. *IEEE/ACM transactions on networking*, 25(1), 579–595.

Zhang, H., Lu, G., Qassrawi, M. T., Zhang, Y. & Yu, X. (2012). Feature selection for optimizing traffic classification. *Computer Communications*, 35(12), 1457–1471.

Zhen, L. & Qiong, L. (2012). Studying cost-sensitive learning for multi-class imbalance in Internet traffic classification. *The Journal of China Universities of Posts and Telecommunications*, 19(6), 63–72.