# Analyse et modélisation du trafic Internet

# THÈSE

présentée et soutenue publiquement le 12 Octobre 2009

pour l'obtention du

**Doctorat de l'Université Pierre et Marie Curie – Paris VI**

**(Spécialité Informatique)**

par

Yousra Haddar Chabchoub

**Composition du jury**

| | |
|---|---|
| François Roueff | TELECOM ParisTech – Rapporteur |
| Patrick Thiran | EPFL (Lausanne) – Rapporteur |
| Fabrice Guillemin | Orange Labs – Examinateur |
| Philippe Flajolet | INRIA – Examinateur |
| Philippe Chrétienne | LIP6 – Examinateur |
| Serge Fdida | LIP6 – Directeur de thèse |
| Philippe Robert | INRIA – Directeur de thèse |

# Remerciements

Je tiens à remercier, Philippe Robert et Christine Fricker qui ont encadré, avec pédagogie et enthousiasme, mes travaux pendant ces années de thèse. J'ai découvert grâce à eux un thème de recherche passionnant : la conception et l'analyse d'algorithmes pour le trafic Internet. Je voudrais leur exprimer ma profonde gratitude pour leur patience, leur disponibilité, leurs conseils pertinents et leur gentillesse.

Je remercie également Serge Fdida d'avoir accepté de diriger ma thèse ainsi que François Roueff et Patrick Thiran, mes rapporteurs, pour leurs remarques et suggestions sur la version préliminaire de ce document .

Je suis reconnaissante à Philippe Chrétienne et à Philippe Flajolet pour avoir accepté de m'honorer de leur participation au jury de cette thèse.

Je souhaite aussi exprimer le plaisir que j'ai eu à collaborer avec Fabrice Guillemin, Frédéric Meunier avec qui j'ai passé des moments inoubliables au tableau, Hanène Mohamed que je remercie pour son soutien moral et sa bonne humeur, Danielle Tibi et Nelson Antunes. Je les remercie tous pour leur apport dans cette thèse et pour toutes les discussions fructueuses.

Mes plus chaleureux remerciements s'adressent aux autres membres anciens et présents du projet RAP, Florian Simatos avec qui j'ai eu le plaisir de partager l'espace de travail, Youssef Azzana pour son aide précieuse, Virginie Collette qui a toujours su intervenir efficacement et au bon moment, Stelios Kouremenos, Mathieu Feuillet et James Roberts. Merci pour la bonne ambiance et la convivialité qui ont régné au sein de ce projet et que je ne saurai oublier.

Je remercie aussi profondément Fabrice Guillemin et son équipe à Orange Labs pour leur collaboration fructueuse et pour nous avoir fourni un riche support d'expérimentation.

Un grand merci à mes parents pour leur encouragement et à Kamel, mon époux qui a su me réconforter et me soutenir durant ces trois années. Et pour finir un clin d'oeil à mon petit Mehdi.

# Table des matières

# Introduction

Le rêve ultime des opérateurs de télécommunications est de savoir à tout instant qui fait quoi sur leurs liens où les débits n'ont cessé d'augmenter ces dernières années. La mesure de trafic est une tâche indispensable pour une meilleure compréhension de sa composition et de l'évolution de ses caractéristiques due à l'émergence de nouvelles applications. Le but final étant de superviser le réseau, d'améliorer la qualité des services offerts aux clients, et de s'adapter à leurs nouveaux besoins.

Pour accéder aux informations relatives aux clients, il est indispensable de faire des mesures de trafic à l'échelle des flots. Un flot est un ensemble de paquets IP ayant certains identifiants communs. Ces identifiants dépendent de l'objectif de la mesure. Par exemple, pour facturer les clients en fonction de leur consommation, il suffit d'agréger les paquets par adresse source. Par contre, pour détecter une attaque par déni de service vers une cible particulière, il faut mesurer le trafic reçu par une cible potentielle et donc il est plus convenable d'identifier le flot par son adresse destination. L'analyse des statistiques sur les flots permet aussi de mieux comprendre le fonctionnement de nouvelles applications comme le Pair à Pair et de prédire leur impact sur le réseau. En effet, les applications Pair à Pair représentent aujourd'hui la majeure partie du trafic commercial dans les réseaux IP des opérateurs de télécommunications quasiment dans le monde entier. Notamment elles engendrent plus de 70% du volume total des données acheminées par le réseau de France Télécom. Les statistiques sur les flots fournissent des informations utiles sur les clients (le nombre de clients actifs, leur débit, leurs comportements, le temps de téléchargement des fichiers...). Ces informations sont exploitables dans différents domaines comme l'ingénierie du trafic ou la sécurité.

La méthode préalablement utilisée par les opérateurs de télécommunication pour faire des statistiques sur les flots de données consiste à stocker tous les paquets, reconstituer les flots et extraire les statistiques sur ces flots. Cette méthode souffre clairement du problème de passage à l'échelle puisqu'elle nécessite une mémoire proportionnelle au nombre de flots. Elle a donc vite vu ses limites suite à l'explosion des volumes de données qui sont transférées. En effet, le stockage d'une heure de trafic sur un lien haut débit nécessite aujourd'hui une mémoire de plusieurs dizaines de giga-octets et pour analyser ce volume gigantesque de données, cette ancienne méthode demande de grandes capacités calculatoires et un temps de traitement considérable. Ce nouveau contexte a poussé la communauté scientifique à réfléchir sur de nouvelles techniques et approches permettant d'accéder à moindre coût (mémoire, temps de traitement) aux statistiques sur les flots. De plus, l'analyse du trafic doit aussi être suffisamment rapide pour répondre aux exigences de réactivité des applications comme par exemple la détection des attaques. Un traitement en ligne du trafic doit être évidemment plus rapide que le débit de réception des données à analyser. Cette contrainte impose un nombre très limité d'instructions ainsi qu'un temps d'exécution extrêmement court car le temps d'interarrivée des paquets est de l'ordre de quelques nanosecondes.

Le point commun de toutes les méthodes récemment conçues est qu'elles ne fournissent pas de statistiques exactes sur les flots, mais uniquement une estimation. C'est le prix à payer pour limiter les ressources (CPU, mémoire) nécessaires au traitement.
Ces méthodes peuvent être divisées en deux catégories en se basant sur leurs principes. Ces deux catégories sont utilisées dans des contextes différents qui dépendent de la nature du trafic disponible :

- **Des algorithmes qui utilisent le trafic exhaustif**
  Ce sont des algorithmes qui traitent la totalité du trafic et fournissent des estimations sur les statistiques des flots. Pour supporter le passage à l'échelle, ces méthodes sont soumises à deux conditions nécessaires : D'une part, le traitement des données doit se faire à la volée, en une seule passe. D'autre part, la mémoire utilisée doit être constante, indépendante du volume des données à analyser. Ces algorithmes sont essentiellement basés sur le hachage des flots et les filtres de comptage. La connaissance à priori des caractéristiques du trafic n'est pas indispensable pour ce genre d'algorithmes.

- **Des algorithmes qui analysent une fraction du trafic**
  L'échantillonnage du trafic est le fait de sélectionner une petite partie du trafic réel (1/100, 1/500, 1/1000...) et de l'utiliser comme support pour faire des analyses et des statistiques. En considérant ainsi un volume de données plus réduit, on arrive à baisser sensiblement le coût de l'analyse en terme de temps de traitement et de mémoire. En revanche, l'analyse du trafic échantillonné peut induire des interprétations erronées, vu qu'on n'a pas une connaissance précise de tout le trafic, mais simplement celle d'un échantillon de ce trafic. Le trafic échantillonné contient en effet très peu d'information sur les flots initialement présents dans le trafic original. Même avec un taux d'échantillonnage de 1/100, la majorité de ces flots ne seront pas identifiés suite à l'échantillonnage. Les quelques paquets des flots capturés sont insuffisants pour reconstituer les propriétés statistiques des flots dans le trafic d'origine (nombre de flots, distribution de leur taille...). L'inférence des caractéristiques du trafic original nécessite donc une compensation de la perte d'information causée par l'échantillonnage. Ainsi, il est indispensable de s'appuyer sur un modèle statistique sur la composition du trafic original pour aboutir à des résultats significatifs. Ces hypothèses, souvent présentées sous forme de modèle statistique décrivant la structure du trafic, doivent bien sûr être testées et validées.

Ces deux cadres algorithmiques sont plus longuement expliqués et illustrés par plusieurs exemples dans ce qui suit.

# 1 Comptage probabiliste dans le trafic exhaustif

## 1.1 Etat de l'art

On trouve dans la littérature plusieurs algorithmes probabilistes qui traitent le trafic exhaustif et fournissent des estimations sur les statistiques des flots. Notamment, le nombre de flots dans le trafic est l'un des paramètres les plus intéressants à calculer. Ce paramètre est particulièrement utile pour la détection des attaques. En effet, si on définit le flot comme une connexion active, l'augmentation brusque du nombre de flots peut être due à la propagation d'un vers ou à une attaque par port scan consistant à balayer les ports d'une ou de plusieurs machines en les

interrogeant pour identifier les ports ouverts.

Il est clair que, pour un comptage exact du nombre $n$ de flots dans une trace de trafic, on a besoin d'une mémoire de taille linéaire en $n$. C'est pour cette raison que le comptage exact a été abandonné au profit du comptage probabiliste qui s'avère suffisant dans plusieurs applications. Le comptage des flots dans le trafic Internet s'inscrit dans un cadre plus général qui est le comptage de la cardinalité d'un multi-ensemble. Un multi-ensemble est un ensemble où les éléments peuvent être répétés. Sa cardinalité est le nombre de ses éléments distincts. Il s'agit d'un problème intéressant qu'on retrouve sous différentes autres formulations comme le comptage de mots différents dans un texte. Parmi les pionniers qui se sont intéressés à ce problème, on trouve Flajolet et Martin avec leur algorithme "Probabilistic Counting" [42]. L'idée de base de cet algorithme est de hacher $n$ flots sur $L$ bits, et de remarquer que la probabilité d'obtenir une séquence du type "$0^k1$..." dans la représentation binaire des valeurs hachées est $1/2^{k+1}$. En notant $R$ la position du 0 le plus à gauche après hachage des $n$ flots, les auteurs proposent d'utiliser $\hat{n} = 2^{\mathbb{E}(R)}/\phi$, comme estimateur du nombre de flots, $\phi$ étant un facteur correctif. Pour évaluer $\mathbb{E}(R)$, le même traitement est effectué en parallèle pour $m$ fonctions de hachage différentes. L'erreur standard de l'estimateur est $0.78/\sqrt{m}$ et la mémoire totale utilisé par l'algorithme est $M = m\ \log_2\ n$. La taille de la mémoire utilisée a été encore plus optimisée dans le cadre d'un autre algorithme "LogLog Counting" conçu par Durand et Flajolet dans [31]. Le principe de cet algorithme s'inspire beaucoup de la technique utilisée dans l'algorithme précédent "Probabilistic counting". L'amélioration consiste à stocker uniquement la position du bit intéressant ($R$ pour "Probabilistic counting") au lieu de stocker toute la séquence des $L$ bits. Ainsi la mémoire totale utilisée devient $M = m\ \log_2\ \log_2\ n$, d'où le nom de l'algorithme. Dans la pratique $\log_2\ \log_2\ n = 5\ bits$. L'erreur standard de l'estimateur fourni par cet l'algorithme "LogLog counting" est $1.04/\sqrt{m}$ pour sa dernière version "HyperLogLog" par Flajolet, Fusy, Gandouet et Meunier [41].
Toujours dans le même contexte, on trouve aussi l'algorithme "mincount" décrit et analysé par Giroire dans [44]. L'idée cruciale de cet algorithme est que l'espérance du minimum de $n$ variables aléatoires tirées au hasard dans $[0,1]$ est $1/(n+1)$. Il est donc possible d'avoir une estimation de $n$ à partir de ce minimum. Le simple passage à l'inverse ne résout pas le problème car cet inverse a une espérance infinie. Pour surmonter cet obstacle, l'auteur estime la moyenne du $k^{\text{ème}}$ minimum en utilisant des fonctions sous-linéaires. La précision de cet algorithme est $1/\sqrt{m}$ pour une mémoire totale de l'ordre de $m\ \log_2\ n$.

Bien que ces algorithmes fournissent une bonne estimation du nombre de flots en utilisant une mémoire limitée, l'information qu'ils donnent en sortie reste assez réduite. Ils sont essentiellement basés sur l'analyse de l'empreinte obtenue suite au hachage des flots. Dès qu'un flot est haché on perd toute information sur son identifiant ou sa taille. Dans le cadre de la détection des attaques par exemple, ces algorithmes sont incapables d'identifier les victimes ou les attaquants. Ces informations sont nécessaires pour un opérateur réseau afin de pouvoir intervenir efficacement et arrêter l'attaque.

Outre ces algorithmes de comptage probabiliste, on trouve dans la littérature une deuxième famille d'algorithmes adaptés à l'analyse du trafic Internet. Ces algorithmes sont basés sur les tables de hachage (bitmap) appelées aussi filtres. Le premier filtre a été conçu par Bloom en 1970 [15]. Il s'agit d'une structure de données de taille fixe ($m$ bits) qui à l'origine était conçue pour répondre au problème d'appartenance d'un élément à un ensemble. Plus précisément, on suppose que l'on dispose d'un ensemble de $n$ éléments $A = \{x_1, x_2, .., x_n\}$ et d'un élément $x$. En utilisant le filtre de Bloom, on veut tester rapidement l'appartenance de $x$ à $A$. L'idée est de

stocker dans le filtre une représentation abstraite de $A$. Pour cela, on initialise les $m$ bits du filtre à 0, et on utilise $k$ fonctions de hachage $h_i$ aléatoires, indépendantes et à valeurs dans $[1,m]$. Pour chaque élément de $A$, les bits aux positions $h_1(x_i), h_2(x_i), .., h_k(x_i)$ sont mis à 1. Ensuite pour tester si $x$ appartient ou non à $A$, on considère les bits aux positions $h_1(x), h_2(x), .., h_k(x)$. Si au moins l'un de ces bits est nul, alors $x$ n'appartient certainement pas à $A$, sinon il y a de fortes chances que $x$ soit dans $A$. Dans ce dernier cas, on ne peut pas affirmer avec certitude l'appartenance de $x$ à $A$ à cause des collisions possibles entre $x$ et les autres éléments de $A$. Une collision entre deux éléments consiste à les associer à une même valeur par l'une des fonctions de hachage. L'utilisation des filtres de Bloom a été ensuite étendue pour d'autres types d'applications notamment le comptage des flots par Estan et Varghese. En effet dans [35], les auteurs proposent un algorithme "multiresolution bitmap" qui donne en ligne une bonne estimation du nombre de flots en utilisant une mémoire fixe $m$ (taille du filtre de Bloom). L'idée de départ de cet algorithme est de hacher vers l'intervalle $[1,m]$ l'identifiant du flot pour chaque paquet et de mettre à 1 le bit indexé par cette valeur dans le filtre, tous les bits du filtre étant initialisés à 0 au début. Le nombre de flots sera à la fin estimé par le nombre de bits à 1. Cette première version de l'algorithme donne une bonne estimation du nombre de flots $n$ sous condition d'utiliser une mémoire linéaire en $n$. Cette restriction a été assouplie en divisant le filtre d'une façon astucieuse en plusieurs parties correspondant à différentes résolutions.
Whang *et al.* utilisent dans [74] le même procédé de hachage mais, pour estimer le nombre de flots, ils tiennent compte des collisions possibles entre les différents flots. Pour cela, ils introduisent la proportion $V_n$ de cases vides dans le filtre après hachage des $n$ flots. Leur estimateur est alors donné par $\hat{n} = -m \log V_n$. Ceci nous rappelle le problème du collectionneur de coupons. Il s'agit en effet du nombre moyen de coupons à acheter pour remplir une proportion $1 - V_n$ des $m$ places de l'album. Cet estimateur donne une erreur standard de l'ordre de 5% pour une mémoire de taille $m = n/10$. On voit qu'un tel estimateur n'est pas adapté aux débits actuels.
On trouve aussi dans la littérature des études qui ne s'intéressent pas à tous les flots, mais focalisent sur un type particulier à savoir les longs flots. Ces longs flots appelés aussi "éléphants" ont suscité un intérêt particulier dans la communauté scientifique. Cet intérêt est motivé par l'impact significatif de ce type de flots sur les performances du réseau. Bien que peu nombreux, les éléphants contribuent au volume total du trafic dans une forte proportion. Contrairement aux petits flots, le débit des éléphants est régulé par la boucle de contrôle de TCP. Les statistiques sur les éléphants sont très utiles pour différents domaines comme la sécurité et l'ingénierie du trafic. De plus dans certaines applications comme la facturation ou la détection d'attaques, l'opérateur a besoin de la liste des éléphants (caractérisés par leurs adresses).

Les filtres de Bloom ont aussi été utilisés par Estan et Varghese dans [34] pour l'identification des éléphants et l'estimation de la distribution de leur taille. Leur algorithme "Multistage filters" utilise $k$ filtres associés à $k$ fonctions de hachage aléatoires et indépendantes $h_i, 1 \leq i \leq k$. Chaque filtre est une mémoire de $m$ compteurs cette fois et non $m$ bits. Pour chaque paquet reçu, l'identifiant $x$ du flot est haché et pout tout $i$, le compteur à la position $h_i(x)$ est incrémenté de 1, dans le ième étage. L'estimation de la taille du flot $x$ est alors donnée par le minimum des compteurs qui lui sont associés $min_i(h_i(x))$. Ces compteurs ne sont pas forcément égaux à cause du problème de collision entre les flots. C'est le fait que deux flots différents peuvent être hachés vers la même valeur par l'une des fonctions de hachage. Les deux flots vont donc incrémenter le même compteur dont la valeur ne sera plus significative. Pour faire face à ce problème, un mécanisme d'incrémentation conservative a été introduit. Il s'agit simplement d'incrémenter le minimum des $k$ compteurs $min_i(h_i(x))$ et non pas tous les compteurs. Ainsi l'utilisation de $k$ filtres réduit les collisions entre les flots et atténue par conséquent la surestimation de la taille des flots. Quand tous les compteurs relatifs à un flot donné dépassent un certain seuil $S$, ce flot

sera déclaré comme éléphant et sera stocké dans une mémoire dédiée. Quelle que soit la taille des filtres, ce comptage ne peut pas être effectué sur le trafic Internet indéfiniment. En effet, vu le nombre gigantesque de flots, les compteurs des filtres vont tous augmenter au fur et à mesure que les paquets arrivent et, à partir d'un certain moment, ils dépasseront tous le seuil $S$. Dans ce cas, tous les nouveaux flots seront déclarés éléphants. Pour remédier à ce problème, Estan et Varghese proposent de rafraîchir les filtres en réinitialisant tous les compteurs à 0 toutes les 5 secondes. Les résultats par cette méthode dépendent fortement de l'intensité du trafic. Les éléphants risquent d'être manqués s'ils ne parviennent pas à émettre $S$ paquets en 5 secondes. Inversement, dans le cas d'un trafic très intense, cette fréquence de rafraîchissement des filtres peut s'avérer insuffisante, auquel cas le nombre d'éléphants sera surestimé. Ce paramètre de 5 secondes ne doit donc pas être statique, et préalablement fixé. Il faut que l'algorithme puisse choisir cette durée automatiquement et la changer au cours du temps en fonction des variations de l'intensité du trafic.

Dans sa thèse [6], Azzana s'est particulièrement intéressé à ce problème. Rappelons que l'objectif de cette méthode est de fournir la liste des éléphants. Azzana a introduit un mécanisme d'effacement adaptatif qui permet de rafraîchir les filtres avec une fréquence qui dépend de l'intensité du trafic. Cette fréquence est choisie d'une façon dynamique au cours du déroulement de l'algorithme de façon à s'adapter aussi aux variations du trafic. Nous avons ici repris la méthode pour la développer, l'améliorer et l'adapter à la détection des attaques dans le trafic Internet.

Les algorithmes d'identification des grands flots ont plusieurs domaines d'application notamment la détection d'attaques par déni de service (DoS). Ces attaques affectent la disponibilité des ressources en empêchant un réseau ou un ordinateur de fournir son service habituel. Les attaques DoS les plus fréquentes visent la connectivité ou la bande passante d'un ordinateur. Elles empêchent les utilisateurs légitimes de se connecter sur le serveur soit en inondant le réseau par un grand volume de trafic qui consomme la totalité de la bande passante (Volume flood) ou en envoyant au serveur un grand nombre de demandes de connexions qui épuisent sa capacité (Syn flood). Hussain *et al.* proposent dans [52] une classification plus complète des attaques DoS.
Selon une étude réalisée en 2001, 90% des attaques DoS utilisent TCP précisément le Syn flood [32]. Tout système qui offre un service basé sur TCP est donc vulnérable aux attaques DoS par Syn flood (serveur Web, serveur FTP, serveur mail....). Le Syn flood exploite une faiblesse dans la conception de la phase de connexion de TCP : Pour chaque demande, le serveur maintient une connexion semi-ouverte pendant un délai de 75 secondes en attendant la réponse pour finaliser l'établissement de la connexion. Pour épuiser la capacité d'un serveur, il suffit donc de lui envoyer plusieurs demandes de connexion, sous forme d'un grand nombre de paquets Syn, en une brève durée. Un paquet Syn est simplement un paquet IP avec l'option Syn dans l'entête, il se traduit par une demande d'établissement de synchronisation de numéro de séquence. L'établissement des connexions n'étant jamais achevé, on arrive alors à occuper inutilement une partie des ressources du serveur pendant une certaine durée. Quand le nombre de demandes de connexions est suffisamment grand, le serveur doit rejeter toute nouvelle demande car toutes ses ressources sont occupées.
En utilisant une définition adéquate d'un flot, certaines attaques DoS peuvent se traduire au niveau réseau par un très grand flot. Pour détecter un Syn flood par exemple, il est naturel de commencer par agréger les paquets Syn par adresse de destination et de chercher ensuite les destinations qui reçoivent un nombre inhabituel de paquets Syn. Autrement dit, on définit le flot comme un ensemble de paquets Syn ayant même destination et on cherche les flots ayant une taille "anormale". Il faut bien sûr fixer des seuils pour faire la différence entre le trafic légitime et les attaques. En général, une attaque est définie comme une déviation notable par rapport à

un comportement standard, mais il n'existe pas de seuil universel pour caractériser cette déviation. D'où une difficulté supplémentaire du problème de détection des attaques par rapport à l'identification des grands flots.
Dans la littérature sur la détection d'attaques, les méthodes basées sur l'analyse de flots souffrent en général d'un problème de passage à l'échelle. Elles sont applicables sur une centaine de flots mais elles ne sont pas adaptées à un trafic intense avec des millions de flots. Dans ce contexte, Lakhina *et al.* [61] proposent une méthode de détection des attaques DoS utilisant comme observable le nombre de paquets par flot sous forme de séries temporelles. L'idée est de surveiller au cours du temps l'évolution du nombre de paquets pour chaque flot. Étant basée sur un comptage exact, cette méthode devient très vite lente et gourmande en mémoire pour un grand nombre de flots. Dans [71] Cheng *et al.* proposent une méthode similaire analysant le taux d'arrivée des paquets par flot. Par un procédé de traitement de signal, ils remarquent que RTT (Round-Trip Time) impose une certaine périodicité dans le signal lié à l'arrivée des paquets par flot. Une attaque sera alors définie comme une perturbation de cette périodicité. L'inconvénient de cette méthode est que si l'attaquant envoie des attaques périodiques, elles ne seront pas détectables. De plus, la périodicité liée à RTT ne semble pas très robuste. Le problème majeur avec ces méthodes demeure celui du passage à l'échelle. Une façon de surmonter ce problème est de compacter l'information sur les flots en utilisant les filtres appelés aussi "sketches". Les sketches permettent de réduire la dimension et d'éviter de maintenir un état par flot. L'avantage des sketches est le fait qu'ils utilisent une mémoire constante, indépendante du nombre de flots. Ils assurent aussi un traitement assez rapide. Dans ce contexte Zhang *et al.* [70] proposent une méthode utilisant les séries temporelles calculées à l'aide des sketches. Les sketches permettent d'estimer rapidement le nombre de paquets par flot. Pour détecter l'attaque, les auteurs utilisent la méthode suivante : on commence par estimer les valeurs futures des sketches en se basant sur les observations passées, puis on compare cette estimation aux valeurs réelles. Si la différence dépasse un certain seuil, une alarme sera déclenchée. Une approche récente développée par Benmammar *et al.* dans [13] consiste à filtrer d'abord le trafic avec les sketches, afin d'identifier et mettre à jour le top $M$ correspondant à la liste des $M$ adresses les plus sollicitées. Ensuite les données censurées sont analysées grâce à une méthode statistique de détection de ruptures fondée sur un test de rang non-paramétrique. L'idée de cette méthode est de suivre au cours du temps la variation du trafic reçu par chaque destination faisant partie des machines les plus sollicitées. Si cette variation correspond à une augmentation importante, l'adresse en question sera considérée comme attaquée. L'inconvénient de cette approche est qu'elle permet de détecter au plus $M$ attaques à un instant donné. Toujours dans le même esprit Chatelain *et al.* proposent dans [21] un algorithme de détection des attaques basé sur les sketches. La détection des anomalies est formulée comme un test statistique de valeurs aberrantes, le trafic normal étant modélisé par des lois Gamma. La faiblesse de cette méthode est qu'elle est incapable de déterminer la nature de l'attaque car elle ne manipule pas les flots.

## 1.2 Notre contribution

**Définition éléphant/souris**

Nous nous sommes particulièrement intéressés à l'étude des longs flots (appelés aussi éléphants par opposition à souris). La dichotomie éléphant/souris a été largement abordée dans la littérature car elle facilite l'analyse des caractéristiques du trafic. En effet, ces deux types de flots ont des caractéristiques différentes et agissent différemment sur les performances du réseau. A l'origine, la notion éléphant/souris était liée au mécanisme de contrôle de congestion du protocole TCP. Les éléphants correspondent à des volumes de transferts assez importants et leurs débits

sont régulés par la boucle de contrôle de TCP afin de partager la bande passante d'une façon équitable. Par contre, les souris sont des transferts de données de courte durée. Elles ne sont pas assez volumineuses pour pouvoir s'adapter au mécanisme de contrôle de congestion imposé par TCP. En effet, elles ne dépassent pas la phase de slow start. Une souris est souvent définie comme un flot comprenant un nombre de paquets inférieur à 20. Un éléphant est un flot qui n'est pas souris. Cette définition a été ensuite élargie à différents autres contextes.

La définition éléphant/souris utilisée dans cette thèse n'est pas unique, mais varie suivant le contexte. Cette classification est essentiellement basée sur la taille du flot, mais la frontière séparant les deux catégories est variable en fonction du problème traité. Nous établissons dans le chapitre 5 une méthode permettant d'estimer le seuil (en nombre de paquets) séparant les éléphants et les souris, en se basant sur un changement de l'allure de la distribution de la taille des flots . Nous montrons également que ce seuil dépend du trafic considéré. Par contre, dans le domaine des attaques, nous proposons une autre définition d'éléphants/souris car dans ce contexte l'éléphant représente l'attaque et se traduit par une nette déviation par rapport à un comportement standard. De plus, la définition du flot dépend à son tour de l'application. Un flot est un ensemble de paquets IP avec certains champs communs, choisis en fonction de l'objectif de la mesure. La notion d'éléphant/souris sera définie dans chaque chapitre de cette thèse suivant le contexte.

La majorité des algorithmes proposés dans la littérature fournissent une information réduite sur les flots. Ils estiment en général le nombre total de flots sans pouvoir donner plus de détails sur la distribution des tailles des flots ou sur les identifiants des longs flots. Pourtant, ces informations sont très intéressantes et très utiles pour l'opérateur réseau. De plus, même si ces algorithmes peuvent fonctionner en ligne grâce au mécanisme de hachage des flots, ils ont souvent un problème d'adaptation aux variations du trafic.

Dans cette thèse nous avons choisi d'étudier plus en profondeur et d'améliorer l'algorithme proposé par Azzana. Rappelons que cet algorithme permet d'identifier et de compter les longs flots en se basant sur les filtres de Bloom. Pour éviter de saturer les filtres, Azzana introduit un mécanisme de rafraîchissement dont la fréquence suit les variations de l'intensité du trafic. Le principe du rafraîchissement est de maintenir le taux de remplissage des filtres au dessous d'un certain seuil. Le taux de remplissage est défini comme la proportion de cases non nulles. Intuitivement, moins les filtres sont remplis, moins un nouveau flot risque d'être haché vers des compteurs associés à d'autres flots. Contrôler le taux de remplissage est donc une façon d'atténuer les collisions entre les flots. Il est très important de limiter les collisions, faute de quoi le comptage sera erroné et les tailles de tous les flots seront surestimés. En particulier, s'il y a beaucoup de collisions, une souris a toutes les chances d'être hachée vers des compteurs égaux à la taille minimale des éléphants, elle sera donc considérée comme éléphant. Les détails de l'algorithme sont expliqués dans le Chapitre 2. Dans ce chapitre, nous proposons également une étude théorique des performances de l'algorithme pour évaluer son erreur moyenne sur l'estimation du nombre d'éléphants. On cherche en particulier à quantifier les faux positifs, c'est à dire le nombre de petits flots (appelés aussi souris) que l'algorithme considère comme des éléphants. Pour ce faire, nous utilisons un modèle simplifié où le trafic est uniquement composé de flots à un seul paquet. Nous montrons que le problème peut être formulé avec une file d'attente M/G/1/C, ce qui facilite l'étude et nous permet moyennant quelques hypothèses d'exprimer le nombre de faux positifs en fonction des différents paramètres de l'algorithme. L'analyse a été ensuite généralisée à des souris de taille quelconque.

Une nouvelle version de l'algorithme est proposée dans le chapitre 1. Intuitivement, pour

limiter les faux positifs (les souris considérées comme éléphant par l'algorithme), il faut que les compteurs aient des valeurs suffisamment basses comparées à la taille minimale des éléphants. Dans certaines configurations, le contrôle du taux de remplissage des filtres est insuffisant à la réalisation de cette condition. En se basant sur cette remarque, nous introduisons un nouveau mécanisme de rafraîchissement contrôlé par la valeur moyenne des compteurs non nuls. Il s'agit d'une nouvelle façon de caractérisation de la charge des filtres. L'idée est de procéder au rafraîchissement dès que cette valeur atteint un certain seuil. Cette méthode améliore les résultats de l'ancien algorithme dans le cas où la taille moyenne des souris n'est pas très petite comparée à la taille minimale des éléphants. A titre d'exemple, si on considère un trafic avec des souris de taille moyenne la moitié de la taille minimale $C$ d'un éléphant, il suffit que deux souris collisionnent pour que leur compteur associé soit à $C$ générant ainsi un faux positif. Pour éviter cette situation, on se fixe un seuil pour la valeur moyenne des cases remplies. L'algorithme ainsi obtenu est testé sur différents types de trafic. Les résultats sont comparés à ceux de l'algorithme d'origine.

Dans le chapitre 1, nous développons aussi un nouvel algorithme de détection en ligne des attaques par Syn flood et par Volume flood. Cet algorithme s'inspire de l'algorithme d'identification des grands flots discuté dans le chapitre 2. Pour repérer les attaques, les paquets sont agrégés par adresse de destination, d'où une nouvelle définition du flot. L'algorithme proposé est aussi basé sur les filtres de Bloom, mais avec un mécanisme de rafraîchissement plus agressif. En effet, dans ce nouveau contexte, le but du rafraîchissement des filtres est d'éliminer rapidement le trafic normal pour ne garder dans les filtres que les flots susceptibles de correspondre à des attaques. Même avec un rafraîchissement agressif, l'attaque sera toujours détectable car elle s'écarte d'une façon considérable du comportement standard. Ceci n'est pas vrai dans le cas de la détection des éléphants car la frontière entre souris et éléphants est très sensible (plus ou moins de $C$ paquets). Le nouvel algorithme a été testé en ligne sur un trafic contenant des attaques et a fourni de bons résultats avec un délai de détection suffisamment court.

# 2 Inférence des paramètres du trafic par échantillonnage

## 2.1 Les méthodes existantes

Plusieurs travaux de recherche ont été menés autour de l'échantillonnage, dont les RFC et les spécifications récemment réalisées par le groupe de travail "PSAMP" (Packet SAMpling) de l'IETF [1]. PSAMP spécifie plusieurs méthodes d'échantillonnage que l'on va décrire.

L'échantillonnage est dit déterministe lorsque les instants d'échantillonnage sont connus à l'avance. Il peut être, soit basé sur le temps (par exemple, on sélectionne un paquet toutes les 5s), ou sur le nombre de paquets (par exemple, on sélectionne un paquet sur 100). L'échantillonnage déterministe peut aussi être apériodique.

Un deuxième type d'échantillonnage est l'échantillonnage aléatoire ou probabiliste. Pour ce type d'échantillonnage, chaque paquet a une probabilité $p$ d'être sélectionné. L'échantillonnage est uniforme si $p$ est constante. Dans le cas contraire, les paquets n'ont pas la même chance d'être sélectionnés, mais on donne plus de chance à certains paquets en fonction de leur contenu, taille... Pour la facturation, par exemple, Duffield *et al.* [59] proposent une méthode d'échantillonnage probabiliste basée sur la taille des flots. Cette méthode permet d'échantillonner avec une grande probabilité les flots ayant un volume assez important. Ainsi, ils estiment le volume de trafic appartenant à un client particulier si sa consommation dépasse un certain seuil. Cette mesure est utilisée comme base de facturation de ce client en fonction de son usage. Duffield *et al.* montrent que les résultats obtenus par un tel échantillonnage sont nettement meilleurs que

ceux issus d'un échantillonnage déterministe. Cette idée est théoriquement intéressante, mais elle est incompatible avec un fonctionnement en ligne. Elle nécessite le traitement de chaque paquet pour calculer la probabilité de le sélectionner en fonction de sa taille, ce qui représente une tâche lourde. Outre la facturation, l'échantillonnage est utilisé afin de détecter les variations des caractéristiques du trafic qui peuvent être causées par le changement de comportement des utilisateurs ou simplement par des accidents au niveau des équipements réseau. Il est nécessaire de s'apercevoir assez rapidement de ces variations pour pouvoir intervenir d'une manière efficace. Dans ce contexte, Choi *et al.* [22] proposent un mécanisme d'échantillonnage probabiliste avec un taux adaptatif permettant d'estimer avec une bonne précision le volume total du trafic en octets. Le taux d'échantillonnage dépend d'une part de l'erreur tolérée sur l'estimation de la charge du lien et d'autre part de la distribution des tailles des $m$ derniers paquets reçus. Il est proportionnel au coefficient $S = (\nu/\sigma)^2$ où $\sigma$ et $\nu$ sont respectivement la moyenne et l'écart type des tailles des paquets. Un modèle auto-régressif est utilisé afin d'actualiser le taux d'échantillonnage. La fréquence de la mise à jour doit être convenablement choisie pour résoudre le problème du compromis entre la rapidité de la détection de la variation de la charge et la quantité des calculs à réaliser. Ainsi le mécanisme d'échantillonnage adaptatif présenté dans leur étude détermine d'une manière dynamique le taux d'échantillonnage minimal permettant d'atteindre le niveau d'exactitude souhaité pour l'estimation du volume du trafic. D'un point de vue pratique, l'échantillonnage s'effectue au niveau du routeur qui a des ressources limitées consacrées essentiellement à sa fonction principale à savoir le routage. L'échantillonnage doit donc surcharger au minimum le routeur. Sur le plan opérationnel, seul l'échantillonnage déterministe est implémenté dans les routeurs, notamment par les sondes NetFlow de Cisco. Sur un lien haut débit, il paraît inconcevable qu'un routeur teste certains champs ou le contenu de chaque paquet pour le sélectionner. Ainsi les deux algorithmes présentés ci-dessus ne sont pas adaptés à un traitement en ligne car ils sont tous les deux basés sur l'étude des tailles des paquets (moyenne, variance).

Aboutir à des statistiques sur le trafic réel en examinant uniquement le trafic échantillonné est une tâche difficile. En effet, il faudra compenser la perte d'information causée par l'échantillonnage en tenant compte de son effet sur les différentes statistiques. Toutefois, certaines caractéristiques générales du trafic sont relativement simples à inférer à partir du trafic échantillonné. A titre d'exemple, il est tout à fait simple d'estimer le volume total du trafic réel en octets ou en paquets à partir de l'échantillonnage déterministe de taux $1/N$. En effet, le volume du trafic échantillonné n'est par définition de l'échantillonnage qu'une fraction $1/N$ du volume du trafic original. Ceci est dû au fait que grâce à l'échantillonnage déterministe, on sélectionnera exactement un paquet tous les $N$ paquets. Ainsi, si on note $P$ et $\hat{p}$ le nombre de paquets respectivement dans le trafic original et échantillonné, alors $\hat{P} = N\hat{p}$ est un estimateur non biaisé de $P$. De même, soit $B$ le volume du trafic original en octets et $\hat{b}$ le volume du trafic échantillonné, alors $\hat{B} = N\hat{b}$ est un estimateur non biaisé de $B$. Pour le cas de l'échantillonnage probabiliste $1/N$, Duffield *et al.* [29] montrent que $\hat{P}$ et $\hat{B}$ sont aussi deux estimateurs non biaisés de $P$ et $B$. Les erreurs standards de ces deux estimateurs sont respectivement :

$$\frac{\sqrt{\mathrm{Var}\hat{P}}}{P} \leq \sqrt{\frac{N}{P}} \; et \; \frac{\sqrt{\mathrm{Var}\hat{B}}}{B} \leq \sqrt{\frac{N}{P}}\frac{b_{max}}{b_{avr}}$$

avec $b_{max}$ et $b_{avr}$ sont respectivement la taille maximale et moyenne des paquets.
Le débit global du trafic original peut simplement être obtenu en divisant le débit du trafic échantillonné par le taux d'échantillonnage.
De plus, avec l'échantillonnage probabiliste ou déterministe, la sélection d'un paquet est complètement indépendante de sa taille. Ainsi, le trafic échantillonné est constitué d'un ensemble de

paquets choisis aléatoirement parmi les paquets du trafic original. La taille moyenne des paquets dans le trafic original peut donc être estimée par la taille moyenne des paquets dans le trafic échantillonné.

Contrairement aux caractéristiques générales du trafic, les paramètres de composition en flots du trafic sont très sensibles à l'échantillonnage. Il n'est pas évident d'inférer des informations sur les flots du trafic original en examinant simplement le trafic échantillonné. En effet, suite à l'échantillonnage, certains flots du trafic réel ne sont pas détectés et ne seront pas pris en compte dans les mesures faites sur le trafic échantillonné. Dans ce contexte, Duffield *et al.* proposent une méthode d'estimation du nombre de flots TCP basée sur les paquets SYN qui annoncent les débuts des connexions TCP [28]. Ils supposent pour cela que chaque flot TCP comporte un seul paquet SYN émis au début du transfert. Si on note $\hat{P}_{SYN}$ le nombre de paquets SYN dans le trafic échantillonné, alors le nombre de flots TCP dans le trafic réel est donné par $\hat{F} = N\hat{P}_{SYN}$, où $1/N$ est le taux d'échantillonnage. L'efficacité de cette estimation dépend de la réalisation de leur hypothèse. En effet, dans le cas des applications pair à pair, par exemple, un flot contient en général plusieurs paquets SYN, d'où une surestimation du nombre total de flots (voir [7]).

Dans [64], Mori *et al.* proposent une technique basée sur l'échantillonnage déterministe pour identifier les éléphants. A l'aide du théorème bayésien, on calcule le seuil du nombre de paquets échantillonnés pour un flot donné pour que ce dernier soit considéré comme éléphant. La valeur du seuil est choisie de telle façon à minimiser la proportion de faux négatifs tout en vérifiant que la proportion de faux positifs reste assez faible. Les auteurs supposent une connaissance à priori de la distribution des tailles des flots qu'ils considèrent à queue lourde. Avec cette méthode et en utilisant un taux d'échantillonnage modéré, ils estiment avec une bonne précision le nombre d'éléphants et les identifient.

Toujours dans le même contexte, Jean-Marie et Gandouet ont récemment proposé une autre approche pour estimer le nombre d'éléphants (voir [43]). Pour cela, ils comptent par l'une des méthodes existantes dans la littérature le nombre total de flots. Ils prennent le cas idéal où le trafic est composé uniquement de deux types de flots : les petits flots ont exactement $\alpha$ paquets et les grands flots ou les éléphants contiennent exactement $\alpha\lambda$ paquets. Sous ces conditions, et en utilisant l'échantillonnage probabiliste de taux $p$, la probabilité d'échantillonner un petit flot ou un éléphant est respectivement donnée par $f = 1-(1-p)^{\alpha}$, et $g = 1-(1-p)^{\alpha\lambda}$. L'estimateur $n_e$ du nombre d'éléphants se déduit alors facilement de la résolution du système suivant :

$$n = n_e + n_m \text{ et } n' = gn_e + fn_m$$

où $n$ et $n'$ sont respectivement le nombre de flots dans le trafic réel et échantillonné et $n_m$ est l'estimation du nombre de petits flots. L'inconvénient de cette méthode est qu'elle nécessite une séparation nette entre les tailles des éléphants et des petits flots, ce qui n'est pas forcément le cas dans le trafic Internet.

Dans [9], Ben Azzouna *et al.* proposent de déterminer la distribution de la durée des éléphants à partir du trafic échantillonné. Pour cela, les flots actifs sont modélisés comme les clients dans une file $M/G/\infty$ avec un temps de service qui suit une loi de Weibull ($\mathbb{P}(\sigma \geq x) = e^{-(\frac{x}{\eta})^{\beta}}$ avec $\beta > 0$ et $\eta > 0$). Ils montrent qu'il est possible d'inférer les deux paramètres de cette loi par analyse du trafic échantillonné, car la loi de la durée de transmission dans le trafic échantillonné est de même nature. Mais cette méthode manque de robustesse car elle repose sur les queues des distributions difficiles à déterminer dans le trafic échantillonné.

Outre l'échantillonnage des paquets, on trouve dans la littérature un deuxième type d'échantillonnage moins connu qui est l'échantillonnage des flots. Cet échantillonnage a pour but la

constitution d'un échantillon aléatoire et représentatif de l'ensemble de tous les flots. Son principe est de sélectionner tous les flots avec la même probabilité indépendamment de leurs tailles. Un flot choisi sera entièrement (par tous ses paquets) présent dans le trafic échantillonné. Flajolet montre dans [40] qu'à partir de cet échantillon aléatoire, on peut estimer le nombre total de flots et même la distribution de leur taille dans le trafic réel. Pour cela il analyse un algorithme d'échantillonnage adaptatif des flots conçu par Wegman. Le principe de cet algorithme est de hacher sur $D$ bits l'identifiant du flot pour chaque paquet et de stocker dans une mémoire de taille $m$ uniquement les flots dont les $d$ premiers bits sont des 0. Ceci revient à échantillonner les flots avec une probabilité de $1/2^d$. Le paramètre $d$ est configuré dynamiquement au cours du déroulement de l'algorithme de telle façon à optimiser l'utilisation de la mémoire. $\hat{F} = 2^d \hat{f}$, où $\hat{f}$ est le nombre de flots dans le trafic échantillonné, est un estimateur non biaisé du nombre total de flots dans le trafic original. L'erreur standard de cet estimateur est $1.20/\sqrt{m}$.

## 2.2 Contribution

L'échantillonnage du trafic réduit certes le volume de données à traiter mais il engendre aussi une perte d'information considérable. Par conséquent il est très difficile d'aboutir à des informations à l'échelle du flot à partir du trafic échantillonné. Ceci explique le fait que dans la littérature on trouve très peu de méthodes robustes permettant d'inférer les caractéristiques des flots par échantillonnage.

Dans cette thèse, nous nous sommes particulièrement intéressés à ce problème. D'abord, dans le chapitre 4 nous avons comparé les deux types d'échantillonnage (déterministe et probabiliste). Dans la pratique c'est l'échantillonnage déterministe qui est le plus utilisé, mais pour modéliser et analyser théoriquement les résultats de l'échantillonnage, il est plus intéressant de considérer l'échantillonnage probabiliste. Nous avons donc montré que, sous condition que les paquets des différents flots soient suffisamment mélangés, ces deux méthodes d'échantillonnage donnent des résultats équivalents du point de vue composition du trafic échantillonné (nombre de flots, tailles des flots). Nous avons aussi montré que si la queue de distribution des flots est lourde (loi de Pareto par exemple), ce qui est souvent le cas, elle peut être inférée à partir de la queue de distribution de la taille des flots dans le trafic échantillonné par une simple division par le taux d'échantillonnage.

Les chapitres 5 et 6, sont dédiés à la caractérisation de la distribution de la taille des longs flots. Dans un premier temps, en considérant différents types de trafic (académique et commercial), on a remarqué que sur une échelle de temps assez longue (de l'ordre de quelques heures), la distribution de la taille des éléphants présente une allure multi-modale. Elle peut être approchée par deux ou trois lois de Pareto, en fonction de l'intervalle de taille considéré. Cependant, sur une petite fenêtre de temps $\Delta$ convenablement choisie, la distribution de la taille des éléphants a clairement un unique comportement et peut être représentée par une seule loi de Pareto. Le choix de $\Delta$ est soumis au compromis suivant : il faut considérer une durée assez courte pour éviter le comportement multi-modal, en même temps, le nombre de flots observés sur cette durée doit être significatif pour pouvoir faire des statistiques. On propose dans le chapitre 5 une méthode expérimentale permettant de bien choisir la durée $\Delta$ et de trouver les différents paramètres de la loi de Pareto à partir du trafic total. Cette méthode permet en particulier de trouver, en fonction du trafic considéré, les tailles des flots pour lesquels l'approximation par la loi de Pareto est valable. En effet, les petits flots (souris) sont exclus car ils ont un comportement différent. De même pour les très grands flots, l'approximation avec la loi de Pareto n'est pas bonne car expérimentalement les tailles des flots sont bornées sur une durée $\Delta$, alors que théoriquement ce n'est pas le cas. Les tailles sont distribuées suivant une loide Pareto. L'avantage de cette méthode

est qu'elle est générale et ne nécessite aucune information à priori sur le trafic en question. Elle a été testée et validée sur des traces de trafic appartenant à France Télécom et au réseau Abilène. Dans un deuxième temps, on suppose qu'on ne dispose que du trafic échantillonné. En faisant l'hypothèse que la distribution de la taille des flots dans le trafic original suit une loi de Pareto, on cherche à caractériser cette loi à partir des informations contenues dans le trafic échantillonné uniquement. Nous avons développé une nouvelle méthode permettant d'estimer, à partir du trafic échantillonné, les paramètres de cette loi de Pareto, la durée d'observation $\Delta$, et le nombre total d'éléphants présents dans le trafic réel. Cette méthode est basée sur $W_k$ le nombre de flots vus $k$ fois après échantillonnage, c'est-à-dire le nombre de flots ayant $k$ paquets dans le trafic échantillonné. Cette observable peut être facilement obtenue par comptage dans le trafic échantillonné. La durée $\Delta$ est par exemple choisie de telle façon que $W_2$ soit assez grand pour garantir un nombre de flots suffisamment grand dans le trafic initial. On montre aussi que sous certaines conditions, on peut exprimer les paramètres de la loi de Pareto en fonction de $W_k$ et $W_{k+1}$, pour un $k$ convenablement choisi. Ainsi, en faisant une hypothèse sur la distribution de la taille des éléphants, nous avons pu exploité l'information réduite donnée par l'échantillonnage pour inférer les caractéristiques des longs flots dans le trafic d'origine. Cette hypothèse compense en quelque sorte la perte d'information causée par l'échantillonnage.

# Première partie

# Identifying large flows in the Internet traffic

# Introduction

To be efficient, network traffic measurement methods have to be adapted to the actual traffic characteristics. Internet links currently carry a huge amount of data at a very high bit rate (40 Gb/s in OC-768). To analyze on-line this traffic, scalable algorithms are required. They have to operate fast, using a limited small memory. The traffic is mainly analyzed at the *flow* level. A flow is a sequence of packets defined by the classical 5tuple composed of the source and destination addresses, the source and destination port numbers together with the protocol type. Flow statistics are very useful for traffic engineering and network management. In particular, information about large flows (also called elephants) is very interesting for many applications. The convention is to fix a threshold $C$ and to call elephant any flow having more than $C$ packets. Elephants are not numerous (around 5 to 20% of the number of flows), but they represent the main part (80-90%) of the traffic volume in terms of packets. Elephant statistics can be exploited in various fields such as attack detection or accounting. For many applications, it is important to design *on-line* algorithms that can identify some of these long flows on the fly.

Due to the very high bit rate and the huge number of flows in IP traffic, it is unrealistic to maintain data structures that can handle the set of active flows. Indeed, maintaining the list of active flows and updating counters for each of them is hardly possible in an on-line context. Consequently, only an estimation of the characteristics of elephants can be expected within these constraints.

A natural solution to cope with the huge amount of data in IP traffic is to use hash tables. A data structure using hash tables, a *Bloom filter*, proposed by B. Bloom [15] in 1970, has been used to test whether an element is a member of a given set. Bloom filters have been used in various domains : database queries, peer-to-peer networks, packets routing, etc. See Broder and Mitzenmacher [16] for a survey. Bloom filters have been used by Estan and Varghese [34] to detect large flows, see the discussion below.

A Bloom filter consists of $k$ tables of counters indexed by $k$ hash functions. The general principle is the following : for each table, the flow ID of a given packet is hashed onto some entry and the corresponding counter is incremented by 1. Ideally, as soon as a counter exceeds the value $C$, it should be concluded that the corresponding flow has more than $C$ packets.

Unfortunately, since there is a huge number of small flows, it is very likely for instance that a significant fraction (i.e. more than $C$ for example) of them will have the same entry, incrementing the same counter, thereby creating a false large flow. To avoid this problem, Estan and Varghese [34] propose to periodically erase *all* counters. Without any a priori knowledge on traffic (intensity, flow arriving rate, etc.) which is usually the case in practical situations, the erasure frequency can be either

1. too low, and, in this case, the filters can be saturated : Because of the large number of small

flows, many of them may be hashed on the same entry of the hash table and, therefore, the corresponding counter is increased accordingly, and consequently creating a "false" large flow.

2. too high and a significant fraction of elephants can be missed in this case : Indeed, the value of the counter of a given entry corresponding to a large flow with a low throughput may not reach the value $C$ if the value of this entry is set to $0$ too often.

The efficiency of the algorithm is therefore highly dependent on the period $T$ of the erasure mechanism of the filters. This quantity is clearly related to the traffic intensity.

Azzana in [6] proposes an improvement for this algorithm by adding a refreshment mechanism that depends on traffic variations. The idea is to decrease all counters by one every time the proportion of non null counters reaches a given threshold $r$. In this way, the refreshment frequency of the filter depends closely on the actual traffic intensity. Notice that the algorithm uses an improvement, the *min-rule*, also called *conservative update* in [34]. It consists in incrementing only the counters among $k$ having the minimum value, for an arriving packet. Indeed, because of collisions, the flow size is greater than or equal to the smallest associated counters, in the ideal configuration where no packet is lost because of the refreshment mechanism. So the min-rule tends to reduce the overestimation of flow size.

Our contribution is the following : In chapter 1, we propose an improvement to the algorithm proposed by Azzana using a new refreshment mechanism. A modified version of the algorithm is also designed in this work to attacks detection. A theoretical analysis of Azzana algorithm is presented in chapter 2. The aim of the analysis is to evaluate the proportion of false positives generated by the algorithm, using a simple model. In chapter 3, we use the supermarket model to study the impact of the min-rule (described above) on the performance of the algorithm.

# 1

# Adaptive algorithms for the identification of large flows

**Contents**




We propose in this chapter an on-line algorithm based on Bloom filters for identifying large flows in IP traffic (a.k.a. elephants). Because of the large number of small flows, hash tables of these algorithms have to be regularly refreshed. Recognizing that the periodic erasure scheme usually used in the technical literature turns out to be quite inefficient when using real traffic traces over a long period of time, we introduce a simple adaptive scheme that closely follows the variations of traffic. When tested against real traffic traces, the proposed on-line algorithm performs well in the sense that the detection ratio of long flows by the algorithm over a long time period is quite high. Beyond the identification of elephants, this same class of algorithms is applied to the closely related problem of detection of anomalies in IP traffic, e.g., SYN flood due for instance to attacks. For that purpose, an original algorithm for detecting SYN and volume flood anomalies in Internet traffic is designed. Experiments show that an anomaly is detected in less than one minute and the targeted destinations are identified at the same time.

## 1.1 Introduction

### Problem statement

We address in this chapter the problem of designing an on-line algorithm for identifying long flows in IP traffic. From the point of view of traffic engineering, this is an important issue. This is also an illustrative and simple example exhibiting the importance for on-line algorithms to adapt to traffic variations. Traffic variations within a flow of packets may be due to several factors but the way the TCP protocol adapts the throughput of connections based on the congestion of the network, notably through the number of packet losses, naturally leads to a stochastic behavior in the arrival patterns of packets. This is a crucial issue which is sometimes underestimated in the technical literature. Moreover, as it will be seen in the second part, the methods developed for this problem can be in fact used to design a quite efficient anomaly detection algorithm.

An algorithm which can satisfactorily run in some instances on limited traces can fail when handling a large traffic trace (e.g., several hours of transit network IP traffic) because of various reasons :

1. Performances deteriorate with time. The size of data structures increases without bounds as well as the time taken by the algorithm to update them.
2. Poor performances occur even from the beginning. Quite often, algorithms depend (sometimes in a hidden way in the technical literature) on constants directly related to the traffic intensity. For a limited set of traces, they can be tuned "by hand" to get reasonable performances. This procedure is, however, not acceptable in the context of an operational network. As a general requirement, it is highly desirable that the constants used by algorithms automatically adapt, as simply as possible, to varying traffic conditions.

### Identification of large flows

Starting from Estan and Varghese's algorithm, an algorithm based on Bloom filters with an additional structure, the virtual filter, and a completely adaptive refreshment scheme is proposed. As it will be seen, the proposed algorithm, based on simple principles, significantly improves the accuracy of algorithms based on Bloom filters. Moreover, the role of the constants used by the algorithm is thoroughly discussed to avoid the shortcomings mentioned above.

### Anomaly detection

An interesting application field of these methods is the detection of anomalous behavior, for instance due to denial of service. During such an attack, a victim is the target of a huge number of small flows coming from numerous sources connected to the network. An on-line identification of such anomalous behavior is necessary for a network administrator to be able to react quickly and to limit the impact of the attack on the victim. The main problem is in this case to be able to separate quantitatively "normal" variations of traffic from these sudden bursts of traffic. Here again, adaptive properties of the detection algorithms to traffic conditions are essential to distinguish between normal variations of traffic and attacks.

Via an adequate aggregation by destination addresses, the problem is expressed in terms of the detection of a single large flow. The problem is then analogous to the one considered in the detection of large flows : Most flows have to be quickly discarded so that only anomalous flows show up. Another algorithm using Bloom filters with an adaptive refreshment mechanism is also proposed in this case : It is based on a fast refreshment scheme depending on the traffic intensity

and on an adaptive estimation of some constants. This algorithm offers good performances to detect SYN flooding attacks and also, via a variant, to detect more subtle (i.e. progressive) attacks such volume flood attacks.

The organization of this chapter is as follows : A detailed description of the algorithm identifying large flows is given in Section 1.2. The algorithm proposed is tested against experimental data collected from different types of IP networks in Section 1.3. The application to the detection of denial of service (DoS) attacks is developed in Section 1.4. Some performance issues of the algorithm are discussed in Section 1.5. Concluding remarks are presented in Section 1.6.

## 1.2 Algorithms with Bloom Filters

### 1.2.1 Preliminary definitions

In this section, we describe the on-line algorithm used to identify large flows and estimate their volume. Recall that a flow is the set of those packets with the same source and destination IP addresses together with the same source and destination port numbers and of the same protocol type. In the following, we shall consider TCP traffic only.

To simplify the notation, large flows will be sometimes referred to as elephants and small flows as mice. For several reasons, this dichotomy is largely used in the literature, see the discussion in Papagiannaki *et al.* [65] for example.

**Definition 1** (Mouse/Elephant)**.** *A mouse is a flow with less than $C$ packets. An elephant is a flow with at least $C$ packets.*

The constant $C$ is left as a degree of freedom in the analysis. Depending on the target application, $C$ can be chosen to be equal to a few tens up to several hundreds of packets. The choice of $C$ is left to the discretion of the operator.

In this first part, one investigates the problem of on-line estimation of the number of elephants. This is probably the simplest problem with all the main common difficulties in the design of algorithms handling Internet traffic : large order of magnitudes and reduced computing and memory capacities.

Note that the estimation of the *total* number of flows in an efficient and nearly optimal way is a quite different problem. Several on-line algorithms have recently been proposed by Flajolet *et al.* [41] and Giroire and Fusy [45]. Unfortunately, the corresponding algorithms are not able to identify elephants as previously defined.

### 1.2.2 Bloom filters

The starting point is the algorithm based on Bloom filters designed by Estan and Varghese [34]. The filter, see Figure 1, consists of $k$ stages. Each stage $i \leq k$ contains $m$ counters taking values from $0$ to $C$. It is assumed that $k$ independent hash functions $h_1$, $h_2$, ..., $h_k$ are available. The total size of the memory used for the filter is denoted by $M$, recall that $M$ should be of the order of several Mega-Bytes. An additional auxiliary memory is used to store the identifiers of detected elephants.

The algorithm works as follows : All counters are initially set equal to $0$ ; if a packet belonging to a flow $A$ is received then :
— If $A$ is in the memory storing elephant IDs then next packet.
— If not, let $\mathsf{min}(A)$ the minimum value of the counters at the entries $h_1(A)$, ..., $h_k(A)$ of the $k$ hash tables.

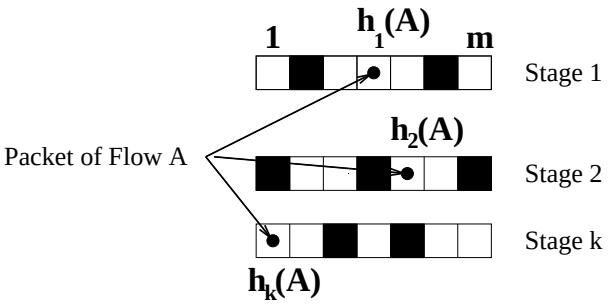



FIG. 1 – The Bloom filter

- If $\min(A) < C$, all the corresponding counters having the value $\min(A)$ are incremented by 1.
- If $\min(A) = C$, the flow of $A$ is added to the memory storing elephant IDs. The flow is detected as an elephant.

The algorithm as such is of course not complete since small flows can be mapped repeatedly to the same entries and create false elephants. One has therefore to clear the filters from the influence of these undesirable flows. Estan and Varghese [34] proposed to erase all counters of the filters on a periodic basis (5 seconds in their paper) :

*Estan and Varghese's refreshing mechanism*

- Every $T$ time units :
  All counters are re-initialized to 0.

Ideally, the constant $T$ should be directly related to the traffic intensity. On the one hand, if the refreshment mechanism occurs frequently, the counters corresponding to real elephants are decreased too often and a significant fraction of them may not reach the value $C$ and therefore many elephants will be missed. On the other hand, if $T$ is too large then, because of their huge number, small flows may be mapped onto the same entry and would increase the corresponding counter to the value $C$, creating a false elephant. This periodic refreshing mechanism could perfectly work if there would be a way to change the value of $T$ according to the order of magnitude of the number of small flows. Such a scheme is however not easy to implement in practice. Moreover, in Section 1.3 experiments based on real traffic traces show that the periodic erasure scheme leads to a poor detection ratio of elephants. This clearly exemplifies the fact that the design of simple robust traffic adaptation scheme is not an easy task in general (think again to the congestion avoidance mechanism of TCP).

Our contribution to this algorithmic setting is two-fold : First, a refreshing mechanism of hash tables properly defined on the current state of the filters and not bound to some fixed time scale is proposed. Second, an additional data structure, the virtual filter, maintained to get a precise estimation of the *statistics* of these large flows (and not only their number) is introduced. These two aspects are separately described in the following.

### 1.2.3 Adaptive Refreshing Mechanism

The general principle is the following : If the state of filters is declared as overloaded then all positive counters are decremented by one.

Note that the values of the counters are only decreased by one instead of reinitialized to 0. The idea is that, if the overload condition of filters is properly chosen, then most of the

values of the non-zero counters will be low. Remember that from the structure of traffic, when compared against the number of mice, there are only a few elephants. The key point is that counters corresponding to elephants will not be decremented to 0. This property is important if one wants to accurately estimate the number of packets of elephants.

Two different criteria to declare when the state of the filter is overloaded are proposed.

- RATIO criterion. Define $r$ as the proportion of non null counters in the multistage filter, the filter is overloaded when $r$ is above some threshold $R$ (90% for example).
- AVERAGE criterion. Define $avg$ as the average of counters values. The filter is overloaded when $avg$ is above some threshold $AVG$ ($C/2$ for example).

The adaptive property of the scheme proposed is clear : As long as the state of the filters is not overloaded then nothing occurs and if there is a peak of activity, the filters are quite quickly filled and the refreshment mechanism is automatically executed.

The rationale behind the RATIO criterion is that if most of counters are non-zero then, very likely, mice have contributed to a significant fraction of the values of the counters so that false elephants show up. Thus, this proportion must be bounded. The best threshold is difficult to find. Thus an interesting alternative is the AVERAGE condition, which considers the average value of counters rather than the number of non-zero counters. Roughly speaking, this corresponds to the saturation threshold of the filter. Notice the mean size of mice which is in practice around 4, can raise up to 7 for some traffic types. In this case, even if the proportion of non-null counters is significant, counters must be decremented more often to avoid accumulation of mice generating false elephants. This is why the AVERAGE condition considers the average value of counters. Our experiments show nevertheless that condition RATIO is sufficient for most of IP traffic types.

Because the number of mice is much larger than the number of elephants, collisions between elephants and mice can be neglected. False elephants are mainly caused by collisions in the hash table between short flows. Missing elephants is the drawback of the algorithm. An elephant having $f$ packets, $f \geq C$, can be missed if its counters do not reach the threshold $C$ because of the refreshment mechanism (all counters are decreased by one when the state of the filters is overloaded).

The number of entries in the memory storing elephants gives an estimation of their total number. It is also possible to store additional variables for each flow in this memory, for instance the starting and finishing time of the elephant corresponding to the arrival times of the first and last packets, the number of packets, the total volume in bytes, the number of segments of a certain type (typically SYN segments for attack detection), etc.

### 1.2.4 Virtual Filter

Missed elephants can be divided into two categories : elephants with low throughput (less than the refreshment frequency) and small elephants. An elephant having a number of packets slightly larger than $C$, can then be missed if there is at least one refreshment during its life time. The following improvement of the algorithm aims at reducing the number of missed elephants by giving elephants more chance to be captured.

The available memory is divided in two halves. In the first half, a Bloom filter as defined above is implemented, it will be called the virtual filter. It operates exactly in the same way for incrementing and refreshing counters. The second half is another Bloom filter, called the real filter ; its counters are incremented in the same way as for the virtual filter but no refreshment mechanism is used except that when a counter becomes equal to 0 in the virtual filter, in that case, it is also set to 0 in the real filter.

The proportion of non null counters is thus the same for the two filters. The identification of elephants is done with the values of counters of the real filter, when all the counters corresponding to some flow are equal to $C$. Note that since the counters are not decremented by one, it is less likely that some packets of elephants will be lost in this manner. The value of a counter in the real filter is therefore always higher than (or equal to) the corresponding counter in the virtual filter. The number of identified elephants is thus higher than what is obtained with the initial version of the algorithm. In particular small elephants have more chance of being identified.

The drawback of the virtual filter is that, in some cases, it can introduce new false positives. As the counters in the real filter are higher, mice are more likely to be considered as elephants. This especially happens when the mean size of mice is not small enough compared to the threshold $C$.

## 1.3 Experimental Results

In this section, the efficiency of the algorithm and the impact of some of its parameters are discussed.

To evaluate the performance of the algorithm, two different traces have been tested : the first trace contains commercial traffic from the France Telecom IP backbone network carrying ADSL traffic. This traffic trace has been captured on a Gigabit Ethernet link in October 2003 between 9 :00 pm and 10 :00 pm. This time period corresponds to the peak activity by ADSL customers, its duration is 1 hour and contains more than 10 millions of TCP flows. The second trace "20040601-193121-1", URL : http ://pma.nlanr.net/ Traces/ Traces/ long/ ipls/ 3/, contains academic traffic issued from Abilene III.

### 1.3.1 Results

In our experiments, the filter consists of 10 stages associated to 10 independent random hash functions ($k = 10$). Elephants are here defined as flows with at least 20 packets ($C = 20$).

First we apply the algorithm proposed by Estan and Varghese [34] to the France Telecom trace in order to identify elephants for which the refreshment time period is set to 5 seconds as specified in that paper. Recall that this algorithm uses a periodic erasure scheme of all counters to refresh the filter. Results are compared to the adaptive refreshment using the RATIO criterion. To be fair in the comparison, at a refreshment instant, instead of decrementing them by one, all counters are set to zero like in Estan and Varghese algorithm.

The number of new elephants per minute found by the algorithms and its exact value are plotted in Figure 2. It shows that the periodic refreshment of Estan and Varghese (5 seconds) is not adapted to the traffic trace since many elephants are missed in this case. The refreshment frequency is too high and elephants cannot send their 20 packets in only 5 seconds. This is due to the fact that in the ADSL traffic trace, elephants are generated by peer to peer file transfers, which are basically with low bit rates (see Ben Azzouna *et al.* [8] for more details).

A change of the value of the period in Estan and Varghese's algorithm would probably improve the accuracy but it is not clear how it can be done "on line". On a one hour long traffic trace, this parameter has to be in fact changed regularly. This is not necessary for short traces, a few tens of thousands of packets say, but this becomes an issue for long traffic traces.

Using the adaptive refreshment with a threshold $R = 90\%$ and a small memory of size $M = 1.31MB$, only about 12% of the elephants are missed. With a memory size of 5.24MB, the error is of the order of 2%. See Figure 7 below.

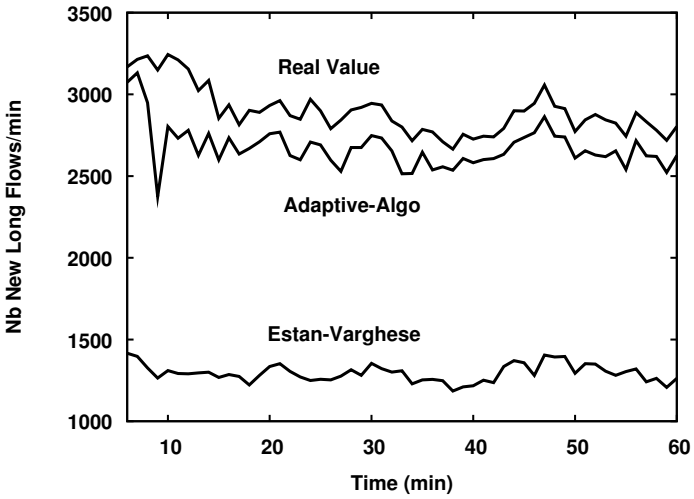



FIG. 2 – Impact of the adaptive refreshment on the estimation of elephants number, $M = 1.31MB$, $R = 90\%$, France Telecom trace

Another important feature of the adaptive algorithm which can be seen from Figure 2 is that it follows very closely the variations of elephant traffic, this is also true for Estan and Varghese algorithm but in a much less accurate way. This is, in our view, the benefit of the adaptive property of our algorithm.

Figure 3 gives the relative error on the estimation of the number of elephants for the three versions of the algorithm : with the refreshment using RATIO and AVERAGE criteria. Both RATIO and AVERAGE criteria give accurate estimations of the total number of elephants. The fact that the relative error remains under 7% for all the duration of the trace shows stability and robustness of the algorithm. The same experiments performed on Abilene trace give similar results ; see Figure 4. So the adaptive algorithm is an efficient method of refreshing the filter without impacting too much the estimation of the number of elephants.

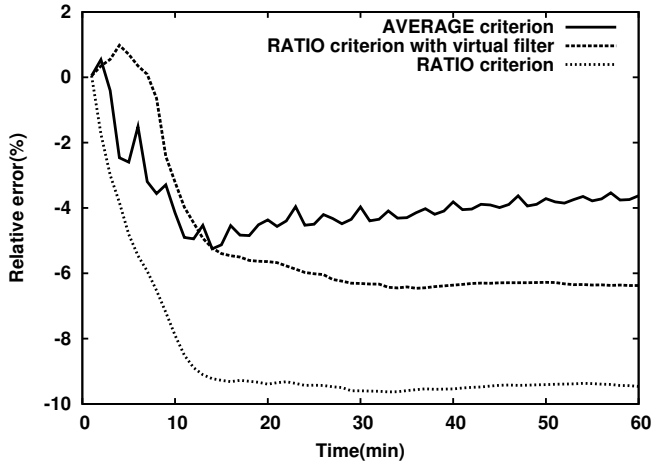



FIG. 3 – Impact of refreshing mechanism and the virtual filter on the estimation of elephants number, $M = 1.31MB$, $R = 90\%$, France Telecom trace

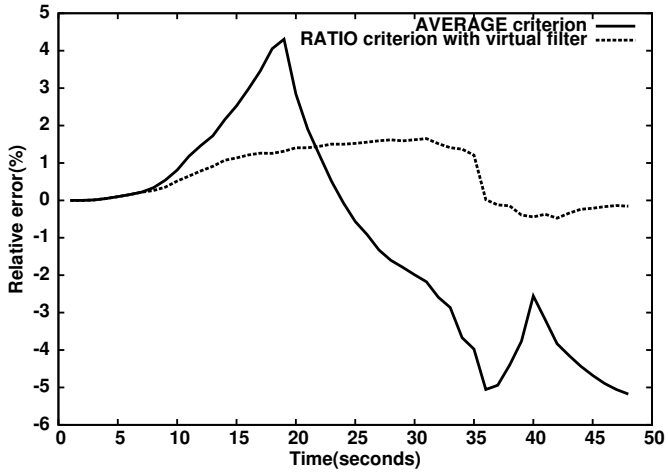



FIG. 4 – Comparison of refreshing criteria for Abilene trace with $M$=1.31$MB$ and $R = 90\%$

Once an elephant has been identified, it is registered in an auxiliary memory together with the number of packets seen and each time a packet of this flow is seen, this value is incremented by 1. In this way, one can estimate the statistics of the sizes of elephants. Figures 5 and 6 show that this statistics of this estimation of the number of packets per elephant is really very close to the real value for the two different traces.

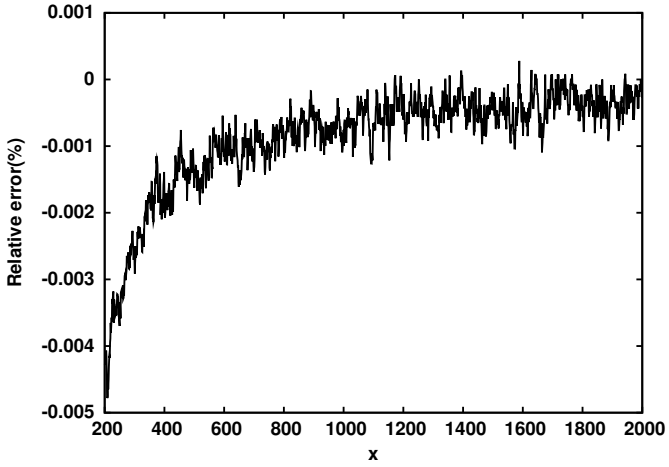



FIG. 5 – Relative error on the number of flows with more than $x$ packets, $M = 1.31$MB, $R = 90\%$, France Telecom trace

### 1.3.2 Impact of the $M$ and $R$ parameters

In Figure 7, we analyze the impact of the size $M$ of the memory used for the Bloom filter on the estimation of the number of the elephants. As expected, using a larger memory improves the accuracy. The error is very close to zero with a memory size of only 5$MB$. In fact the filter is refreshed less frequently which gives more chance for elephants to be detected.

Figure 8 shows the dependence of the accuracy of the estimate for several values of the threshold $R$. A threshold of 90% gives a good estimation of the number of elephants. We just

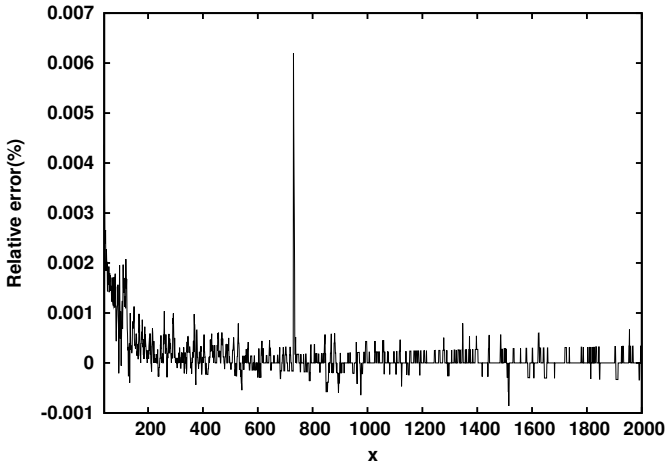



FIG. 6 – Relative error on the number of flows with more than $x$ packets, $M = 1.31MB$, $R = 90\%$, Abilene trace

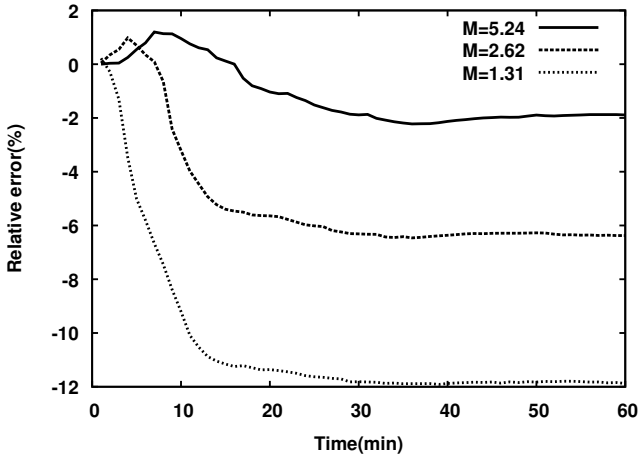



FIG. 7 – Impact of memory size $M$ of the filter, $R = 90\%$

miss about 7% of the elephants. With a higher threshold, we miss less elephants but some false positives can be added. So there is clearly a trade-off on the choice of $R$. See Chabchoub *et al.* [19] for more details.

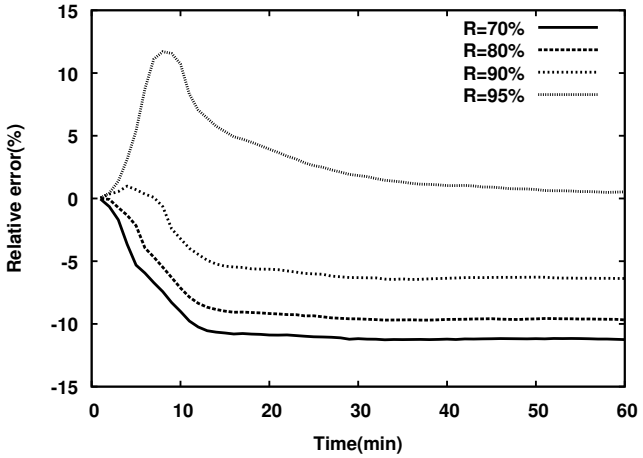



FIG. 8 – Impact of $R$ for RATIO criterion, $M = 1.31MB$

## 1.4 Anomaly Detection

### 1.4.1 Context

Several types of anomalies are considered in this section in connection with denial of service (DoS) attacks. Here we are only interested in *SYN flood* and *volume flood* attacks which are the most common DoS attacks. See Hussain *et al.* [52] for a classification of DoS attacks.

A *SYN flood* exploits a weakness in the connection phase of TCP, also called "the three way handshake". This attack consists of sending a large number of SYN packets to the same destination (or group of destinations) during a small interval of time. Due to the TCP implementation, the destination allocates resources to all these connection requests and will maintain many half-open connections waiting for acknowledgments from sources for about one minute. A large number of SYN packets consume therefore a significant fraction of the resources of the targets and, at the end, the corresponding machines become unreachable (see Wang *et al.* [32] for more details). In this setting the goal is to design an one-line algorithm which can detect an attack in less than one minute. Such a detection can be used by the network operator in order to filter SYN segments towards the victim.

While a SYN flood consists of a sudden arrival of a large number of SYN segments, a *volume flood* attack uses a few TCP flows and gradually transmits with a steady increase of the transmission rate a huge amount of data which will consume the available bandwidth of the target.

Several methods have been developed in the technical literature for DoS flooding detection ; they are mainly based on TCP properties such as periodicity in Chang *et al.* [71], or SYN and FIN packets counting in Wang *et al.* [32], Barford *et al.* [12], Krishnamurty *et al.* [70]. Most of them suffer from scalability or robustness, especially if only sampled traffic is available as it is usually the case in backbone networks.

For SYN flood detection, the main difficulty is in distinguishing between the (normal) variations of traffic and a sudden and anomalous sequence of SYN packets. In the technical literature, attacks are sometimes defined as a notable variation from the standard behavior of specific parameters of the model : parameters of some specific statistical models or long range dependence variables like Hurst parameters for signal processing approaches for example. If the algorithms based on these representations may be efficient to detect some anomalous behaviors, they cannot, in general, assert the nature of the attack because they handle a aggregated information on the flows rather than a more detailed description of the traffic. See Chatelain *et al.* [21] and Lakhina *et al.* [54].

### 1.4.2 SYN flood attacks

The algorithm proposed for an on-line detection of SYN and volume flood is derived from the algorithm presented in Section 1.2, but with a different refreshing mechanism of the Bloom filter.

As explained above, SYN packets with a given destination address are aggregated as a single "flow". In this case, by using a Bloom filter as before, the refreshing mechanism of the multistage filter has a different purpose : it should eliminate quickly all normal flows using an aggressive refreshing mechanism so that if a "large" flow survives then it must be a SYN flood attack. As it is easily seen, the term "large" has to be properly defined. Roughly speaking, this means that such a flow is much larger than the other "normal" flows. Again, because of the variation of traffic, an adaptive scheme has to be devised to properly define these concepts.

The main idea of the algorithm is to evaluate a varying average $m_n$ of the largest flow in several sliding time windows of length $\Delta$. The quantity $m_n$ describes "normal flows"; it is periodically updated in order to adapt to varying traffic conditions. It is a weighted average that takes into account all its past values to follow carefully traffic variations but not too closely. If a flow in the $n$th time window is much larger than $m_{n-1}$, it is considered as an attack, and the moving average is not updated for this time window.

The following variables are used.

- As before, $r$ is the proportion of non-zero counters in the Bloom filter.
- $S$ is a multiplicative detection threshold. Roughly speaking, an attack is declared when an observation is $S$ times greater than the "normal" behavior. The value of $S$ is fixed by the administrator.
- $R_s$ and $R$ are thresholds for the variable $r$. The constant $R_s$ is independent of traffic and taken once and for all equal to $50\%$ and $R$ is a variable threshold depending on the traffic type considered.
- $\alpha$ is the updating coefficient for averages, ; $\alpha = 0.85$ in our experiments.
- $\Delta$ is the duration of the initialization phase (1 mn in the chapter). It is in fact a bound for the time before which an attack should be detected.
- $m_n$ is the weighted moving average for the $n$th time window.

The algorithm starts with an initialization phase of length $\Delta$ in order to evaluate the threshold $R$. At the end of this phase, $R$ will be definitively fixed for the rest of the experiment. In addition, as this phase corresponds to the first time window, the moving average $m_1$ will be initialized as the biggest counter obtained. See Table 1.1 for the description of the algorithm.

Note that an alarm is declared during the $n$th time window when the value of a counter is greater than $Sm_n$. At the beginning, the first time window is fixed (its duration is $\Delta$) but, since the evolution depends on the occupation rate of the filters, the duration of the other time windows is variable. If traffic characteristics are not much varying, time windows durations remain around

Initialization phase :
- All counters are 0.
- The Bloom filter is progressively updated with SYN packets by using their destination address.
  - After a duration $\Delta$, evaluate the variable $r$
    - if $r \leq R_s$ then $R := r$ else $R := R_s$.
    - $m_1 :=$ maximum of the values of counters of the multistage filter.

Detection phase : the $n\ th$ time window
- At the beginning all counters are initialized to 0.
- The Bloom filter is progressively updated with SYN packets by using their destination address.
  - if a counter exceeds $S\ m_{n-1}$, an attack is declared.
  - if $r \geq R$
    - $\max_n$ : maximum of the values of counters of the multistage filter.
    - if $\max_n < S\ m_{n-1}$
      $m_n = \alpha\ m_{n-1} + (1-\alpha)\max_n$
    - start the $(n+1)$th time window.

TAB. 1.1 – Algorithm for SYN flood detection.

one minute. In this case, an attack is detected at the latest after one minute so that the network administrator can react quickly.

### 1.4.3 Volume flood attacks

For progressive attacks, the impact on traffic cannot be clearly seen in a time window of one minute. In fact the attack can be so slow that it could be locally considered as a normal traffic variation. This kind of attacks has typically a long duration. In this situation, we consider a larger time window in order to detect the anomalous impact of the attack on traffic. Thus, to cope with these attacks, the algorithm is used but with a larger time window $\Delta'$ of 5 minutes. This new filter operates in the same way but on a longer time scale and is completely independent of the first filter. In particular, it has its own parameters : $R'$, $r'$, $R'_s$, $m'_n$, $\max'_n$, and $S'$.

### 1.4.4 Experimental Results

To evaluate and validate the attack detection algorithm described in the previous section, we run experiments with two France Telecom traces, one from the IP collect network carrying in majority ADSL traffic and the other from the IP transit network (OTIP). In this latter case, only sampled traffic is available. The characteristics of the traffic traces are given in Table 1.2.

| Traces | Nb. IP pack. | Nb. Flows | Duration |
|---|---|---|---|
| OTIP | $105.10^6$ | $4.10^6$ | 3 days |
| ADSL | $825.10^5$ | $32.10^5$ | 3 hours |

TAB. 1.2 – Characteristics of sampled traffic traces used for attack detection.

To detect SYN and volume flood using two time scales ($\Delta = 1$ mn and $\Delta' = 5$ mn), we need four filters. Each filter contains ten stages ($k = 10$) and has a total size $M$ around $1MB$.

In Figure 9, the ADSL trace is divided into several time windows of 5 mn and, for each interval, the volume of the largest SYN flow is computed. The observed peaks seem to correspond

to attacks. Tested on this trace, the algorithm detected two SYN flood against two different IP addresses. The response time of the algorithm is satisfactory as the alarms are raised at the beginning of the attacks. It should be noted that when the duration of the time window is 1 mn, only the second attack is detected.

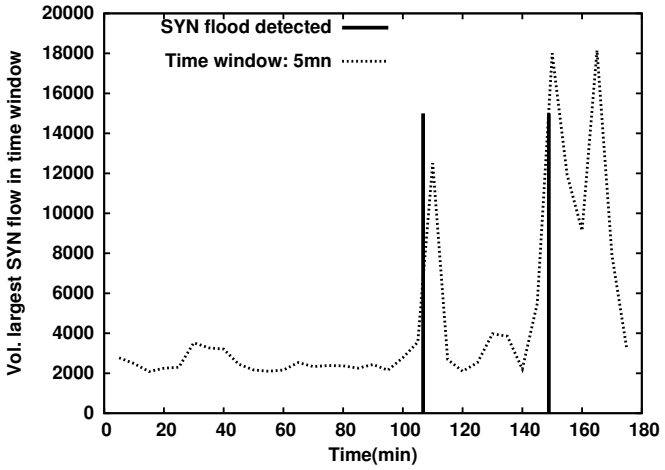



FIG. 9 – SYN flood detection for ADSL trace with $S$=5 and $S'$=3

In Figure 10, the same trace is used to detect volume flood. The volume of the flow is now the number of packets which are not SYN packets. SYN packets are not computed to prevent from considering some SYN flood as volume flood. The algorithm detects one volume flood using the time window of 5 mn. When the duration of the time window 1 mn, no attack is detected.

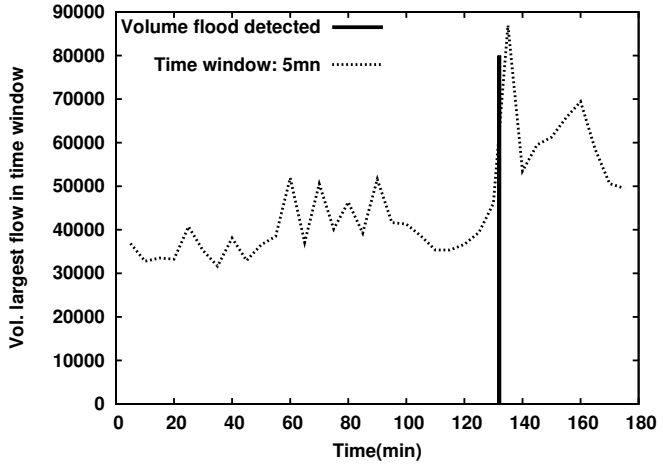



FIG. 10 – Volume flood detection for ADSL trace with $S$=5 and $S'$=2

In Figures 11 and 12 the OTIP trace is considered. This trace contains many attacks. As it can be seen, the algorithm raises several alarms which coincide with the largest flows represented by the highest peaks.

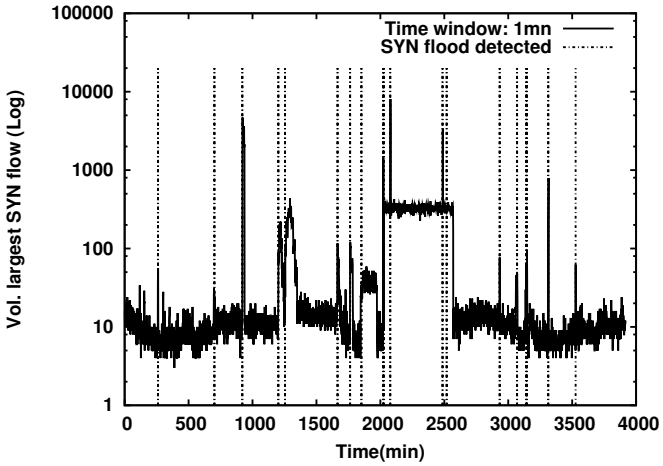



FIG. 11 – SYN flood detection for OTIP trace with $S = 5$ and $S' = 3$

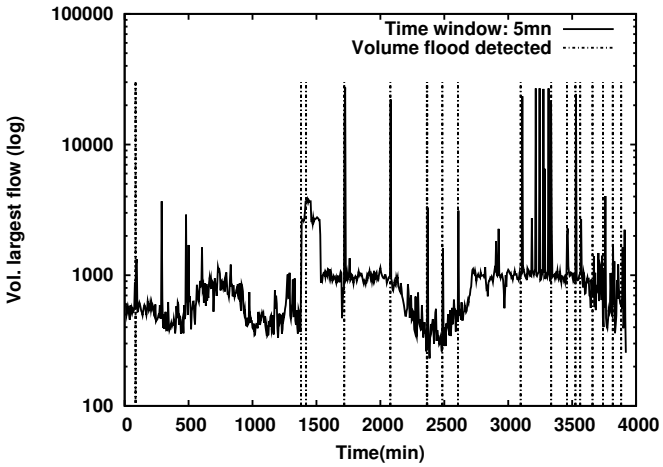



FIG. 12 – Volume flood detection for OTIP trace with $S = 5$ and $S' = 2$

### 1.4.5 Remark on thresholds

The algorithms use the variables $S$ for SYN flood and $S'$ for volume flood. They are related to the network administrator's decision about the precise definition of an anomalous behavior. In the experiments with the OTIP trace for SYN flood detection or volume flood detection, there is clearly a set of events which will be qualified as "attacks" for a large range of values of $S$ and $S'$. Note however that, for some large but "milder" variations, the qualification as attack will depend on the particular value of these parameters. There is no way to avoid this situation in our view. This is the role of the administrator to define the level of abnormality in traffic.

## 1.5 Performance issues

In this section, we discuss briefly, from a modeling point of view, some of the performance issues concerning the refreshing mechanism of the algorithms presented in the chapter. They are addressed in more detail in Chabchoub *et al.* [19, 20]. Recall that our algorithms have the following parameters :

- $R$ : overload ratio of the filter,
- $C$ : minimum size of large flows (elephants),
- $m$ : number of counters per hash table,
- $k$ : number of hash tables in the filter.

The number $m$ is large and the value of $C$ is fixed. The main issue is in fact to investigate the sensitivity of the value of $R$ on the performances : How can one choose $R$ no too large to avoid as much as possible false elephants but large enough to prevent from missing many true elephants. The problem reduces to estimate the error generated by mice, i.e., the ratio of false positives in a simplified model where there are just mice. The case where $k = 1$ is first investigated as the simplest model. Then simplified models are developed for the case where $k \geq 2$.

In a first step, a one-stage filter is considered and traffic is supposed to be composed of only of mice of size 1. The analysis uses Markovian techniques : If $W_n^m(i)$ is defined as the proportion of counters with values $i$, $i \in 0,\ldots,C$ just before the $n$th refreshment for a filter with $m$ counters then the process

$$(W_n^m, n \geq 0) = ((W_n^m(i), i = 0,\ldots,C), n \geq 0)$$

is a Markov chain on a finite state space with invariant distribution $\pi_m$. As $m$ gets large, it is shown that the sequence $(W_n^m, n \geq 1)$ converges to a dynamical system with unique fixed point $\overline{w}$. It turns out that $\overline{w}$ has a nice interpretation in terms of the stationary measure $\mu_\lambda$ of a $M/G/1/C$ queue with service time 1 and arrival rate $\lambda$ and

$$\overline{w} = \mu_{\lambda(\overline{w})}$$

where $\lambda(w) = \log(1 + w(1)/(1-r))$. As the invariant measure $\mu_\lambda$ can be computed as the solution of a linear system of $C+1$ equations, $\lambda(\overline{w})$ is thus the solution to a fixed point equation.

The behavior of the system can be described as follows. At equilibrium, the average number of packets between two refreshing instants is of the order of $\lambda(\overline{w})m$. Due to finite capacity $C$, this quantity is greater than the number of removed packets $Rm$ at each refreshment. In particular $\lambda(\overline{w})$ is not necessarily less than 1. For $R$ enough close to 1, it is shown that $\lambda(\overline{w})$ can in fact exceed 1 which changes the qualitative behavior of the system : if the arrival rate $\lambda(\overline{w})$ is less than 1, then $\overline{w}$ is concentrated on small values 0, 1, 2... of the state space $\{0,1,\ldots C\}$. When $\lambda(\overline{w}) > 1$ the distribution $\overline{w}$ is mainly concentrated on the highest values $C$ and $C-1$. Since false positives are closely related to the quantity $\overline{w}(C)$, this implies that the proportion of false

positives is much higher in this case. Consequently, there is a critical value $r_c < 1$ of $R$, which corresponds to $\lambda(\overline{w}) = 1$, so that the performances of the algorithm deteriorate when $R > r_c$. Similar conclusions hold also in the case of $k$-stage Bloom filter.

In practice, in the (simplified) case of 1-packet mice, the critical rate $r_c$ is close to 1. But, for general mice size distribution, $r_c$ is much lower than 1. Thus, the RATIO criterion does not perform well for any value of $R$. To conclude, the analysis confirms that the threshold $R$ has to be chosen, otherwise the RATIO criterion cannot control the saturation of the filter. This control is contained in the design of the AVR criterion.

Let us give an insight into a model taking into account the case where just the counters having the minimum value are incremented. The analysis can be generalized to more general situations. The idea is also to express quantities via a continuous time process. The main tool is the system of $m$ queues with a Poisson arrival process with rate $\lambda m$ where customers join the shortest of the $k$ queues chosen at random among $m$. This system is well-known in the literature (see Mitzenmacher [62], Vvedenskaya [73], Graham [47] and others). In order to use it, the model must be slightly modified : the mouse increments just one counter having the minimum value and $C$ is taken as $20/k$. The conclusion is that the behavior of the model should follow the same lines but many points are more difficult to catch. For example, as far as we know, the system of queues corresponding to mice with general size distribution has never been studied in the literature. Nevertheless, the analysis gives rise to related models which could lead to improve or simplify the algorithm.

## 1.6 Concluding remarks

We have presented in this chapter an original adaptive algorithm for identifying elephants in Internet traffic. As earlier proposed by Estan and Varghese, this algorithm is based on Bloom filters, but instead of periodically erasing the filter, we introduce different original criteria to decrement the various counters of the filter. In order to improve the accuracy of the algorithm, we have introduced the concept of virtual filter, whose counters are less frequently decreased. The proposed algorithm has been tested against different traffic traces and performs better than the one by Estan and Varghese.

Finally, the proposed elephant identification mechanism has been adapted in order to detect flood anomalies (SYN and volume floods) in Internet traffic. This gives rise to a new algorithm, whose key parameters adapt to network traffic. This algorithm has been successfully tested with two types of traffic traces (corresponding to residential and transit traffic).

# 2

# Analysis of an algorithm catching elephants on the Internet

**Contents**



The chapter deals with the problem of catching the elephants in the Internet traffic. The aim is to investigate an algorithm proposed by Azzana based on a multistage Bloom filter, with a refreshment mechanism (called *shift* in the present chapter), able to treat on-line a huge amount of flows with high traffic variations. The algorithm is simplified in order to focus on this refreshment mechanism. For that, the so-called "min rule" is not taken into account. An analysis of a simplified model estimates the number of false positives. Limit theorems for the Markov chain that describes the algorithm for large filters are rigorously obtained. The asymptotic behavior of the stochastic model is here deterministic. The limit has a nice formulation in terms of a $M/G/1/C$ queue, which is analytically tractable and which allows to tune the algorithm optimally.

# Introduction

## Description of the algorithm

The algorithm designed by Azzana uses a *Bloom filter with counters* and involves four parameters in input : the number $k$ of stages in the Bloom filter, the number $m$ of counters in each stage, the maximum value $C$ of each counter, i.e. the size threshold $C$ to be declared as an elephant and the *filling rate* $r$.

A Bloom filter is a set of $k$ stages, each of these stages being a set of $m$ counters, initially at 0 and taking values in $\{0,\dots,C\}$. Together with the $k$ stages $F_1,\dots,F_k$, one supposes that $k$ hashing functions $h_1,\dots,h_k$ are given, one for each stage. We make the (strong) assumption that these hashing functions are independent, which implies that $k$ is small ($k = 10$ is probably the upper limit). Each hashing function $h_i$ maps the part of the IP header of a packet indicating the flow to which it belongs, to one of the counters of stage $F_i$.

The algorithm works on-line on the stream, processing the packets one after the other. Flows identified as elephants are stored in a list $\mathcal{E}$. When a packet is processed, it is first checked if it belongs to a flow already identified as an elephant (that is a flow already in $\mathcal{E}$). Indeed, in this case, there is no interest in mapping it to the $k$ counters, and the algorithm simply forgets this packet. If not, it is mapped by the hashing functions on one counter per stage and it increases these counters by one, except for those that have already reached $C$, in which case they remain at $C$. When, after processing some packet, all the $k$ counters are at $C$, the flow is declared to be an elephant and stored in the dedicated memory $\mathcal{E}$. When the proportion of nonzero counters reaches $r$ in the whole set of $km$ counters, one decreases all nonzero counters by one. This last operation is called the *shift*.

## Motivation of the algorithm

Packets of a same flow hit the same $k$ counters, but two distinct flows may also increase the same counter in one or several stages. The idea of using several stages where flows are mapped independently and uniformly, intends to reduce the probability of collisions between flows. The shift is crucial in the sense that it prevents the filters to be completely saturated, that is, to have many counters with high values. Without the shift operation, mice would be very quickly mapped to counters equal to $C$ and declared as elephants. The algorithm would have a finite *lifetime* because when the filter is saturated, nothing can be detected.

## False positive and false negative

A *false positive* is a mouse detected as an elephant by the algorithm. A *false negative* is an elephant not declared as such (hence considered as a mouse) by the algorithm. Generally, a false negative is worse than a false positive. Think of an attack : One does not want to miss it, and a false alarm has less serious consequences than a successful attack.

In our context, a false positive is a mouse one packet of which is mapped onto counters all $\geq C-1$. A false negative is due to the shift, and if it happens, it means that there were at least $f-C+1$ shifts during the transmission time of some elephant of size $f$. If shifts do not occur too often, a false negative is then an elephant whose packets are broadcast at a slow rate.

Intuitively, and it will be confirmed by the forthcoming analysis, if the parameters (actually $r$) are chosen so as to maintain counters at low values, then shifts occur often, and if one tries to decrease the shift frequency, then the counters tend to have high values. Therefore, a compromise has to be found between these two properties (frequency of the shifts, height of the

counters), which translates into a compromise between false positives and false negatives. This last compromise depends on the applications.

## A Markovian representation

In this chapter, we will mainly focus our analysis on the one-stage filter case when the traffic is made up only of mice of size 1. The aim is to estimate the proportion of false positives. From this analysis, we will then derive results for the general case. Let us now introduce our main notations.

We thus assume that $k = 1$ and that all flows have size 1. Throughout the chapter, $W_n^m(i)$ denotes the proportion of counters having value $i$ just before the $n$th shift in a filter with $m$ counters. According to this notation, $W_n^m(0)$ is close to $1-r$ and $\sum_{i=0}^{C} W_n^m(i) = 1$. Notice that the $n$th shift exactly decreases the number of nonzero counters by $mW_n^m(1)$. An important part of our analysis will consist in *estimating* $W_n^m(C-1) + W_n^m(C)$. Indeed, it gives an upper bound on the probability that a flow is declared as an elephant (that is a false positive) between the $(n-1)$th and the $n$th shifts, since, according to our assumptions, there is no elephant at all.

In this framework ($k = 1$ and flows of size one), the algorithm has a simple description in terms of urns and balls. Each flow is a ball thrown at random into one of $m$ urns (each urn being one of the $m$ counters). When a ball falls into an urn with $C$ balls, it is immediately removed, in order to have at most $C$ balls in each urn. When the proportion of non empty urns reaches $r$, one ball is removed in every non empty urn.

For $m$ fixed, $(W_n^m)_{n\in\mathbb{N}} = \left((W_n^m(i))_{i=0,\dots,C}\right)_{n\in\mathbb{N}}$ is an ergodic Markov chain on some finite state space. Its invariant probability measure $\pi_m$ is the distribution of some variable $W_\infty^m$. For $C = 2$, the first non-trivial case, even the expression of the transition matrix $P_m$ of the Markov chain is combinatorially quite complicated and an expression for $\pi_m$ seems out of reach. In practice, the number $m$ of counters per stage is large. This suggests to look at the limiting behavior of the algorithm when $m$ tends to $\infty$. We use as far as possible the Markovian structure of the algorithm in order to derive rigorous limit theorems and analytical expressions for the limiting regime. This is the longest and most technical part of the chapter, which also contains the main result, from a mathematical point of view.

## Main results

The model considered in the chapter describes the collisions between mice in order to evaluate the number of false positives due to these collisions. In a one-stage filter where all flows are mice of size 1, the Markov chain $(W_n^m)_{n\in\mathbb{N}}$ describes the evolution of the counters observed just before shift times. The main result is that, when $m$ is large, the random vector $W_\infty^m$ converges in distribution to some deterministic value $\overline{w}$.

This result is not quite completely proved. The way to proceed is classical for large Markovian models (see for example [30] and [4]). The idea is to study the convergence of the process over finite times. It is shown that the Markov chain given by the empirical distributions $(W_n^m)_{n\in\mathbb{N}}$ converges to a deterministic dynamical system $w_{n+1} = F(w_n)$, which has a unique fixed point $\overline{w}$. The situation is analogous in discrete time to the study by Antunes and al. [4]. A Lyapunov function for $F$ would allow to prove the convergence in distribution of $W_\infty^m$. Such a Lyapunov function is exhibited in the particular case $C = 2$. The dynamical system provides a limiting description of the original chain which stationary behavior is then described by $\overline{w}$. The fixed point $\overline{w}$ has the following interpretation.

The fixed point $\overline{w}$ is identified as the invariant probability measure $\mu_{\overline{\lambda}}$ of the number of customers in an $M/G/1/C$ queue where service times are 1 and arrival rate is some $\overline{\lambda}$ satisfying the fixed point equation

$$\mu_{\overline{\lambda}}(0) = 1 - r$$

or equivalently

$$\overline{\lambda} = \log\left(1 + \frac{\mu_{\overline{\lambda}}(1)}{1-r}\right).$$

As a byproduct, the stationary time between two shifts divided by $m$ converges in distribution to the constant $\overline{\lambda}$. Thus the inter-shift time (closely related to the number of false negatives) and the probability of false positives are respectively approximated by $\overline{\lambda}m$ and bounded by $\mu_{\overline{\lambda}}(C-1) + \mu_{\overline{\lambda}}(C)$ when $m$ is large.

When mice have general size distribution, the previous model is extended to an approximated model where packets of a given mouse arrive simultaneously. The involved quantity is the invariant measure of an $M/G/1/C$ queue with arrivals by batches with distribution the mouse size distribution. In the case of size 1 mice, the multi-stage filter case is investigated.

Even if $\mu_{\overline{\lambda}}$ is not explicit, which complicates the exhibition of a Lyapunov function, the quantities $\overline{\lambda}$ and $\mu_{\overline{\lambda}}(C-1) + \mu_{\overline{\lambda}}(C)$ can be numerically computed. It appears that the latter quantity is an increasing function of $r$ (as $r$ varies from 0 to 1). Hence, given the mouse size distribution, one can numerically determine the values of $r$ for which the algorithm performs well.

Section 2.1 is the most technical part of the chapter. It investigates the one-stage filter in case of size 1 flows. In Section 2.2, this analysis is generalized to a general mouse size distribution in a simplified model and to a multi-stage filter. Then Section 2.3 is devoted to discussing the performance of the algorithm, to experimental results and improvements (validated through an implementation).

## 2.1 The Markovian urn and ball model

In this section, $C$ is fixed and we consider the sequence $(W_n^m)_{n\in\mathbb{N}}$, where $W_n^m$ denotes the vector of the proportions of urns with $0,\ldots,C$ balls just before the $n$th shift time. For $m \geq 1$, $(W_n^m)_{n\in\mathbb{N}}$ is an ergodic Markov chain on the finite state space

$$\mathcal{P}_m^{(r)} = \left\{ w = (w(0),\ldots,w(C)) \in \left(\frac{\mathbb{N}}{m}\right)^{C+1}, \sum_{i=0}^{C} w(i) = 1 \text{ and } \sum_{i=1}^{C} w(i) = \frac{\lceil rm \rceil}{m} \right\},$$

(where $\lceil rm \rceil$ denotes the smallest integer larger or equal to $rm$) with transition matrix $P_m$ defined as follows : If $W_n^m = w \in \mathcal{P}_m^{(r)}$, then $W_{n+1}^m$, distributed according to $P_m(w,.)$, is the empirical distribution of $m$ urns when, starting with distribution $w$, one ball is removed from every non empty urn and then balls are thrown at random until $\lceil rm \rceil$ urns are non empty again, balls overflowing the capacity $C$ being rejected. The required number of thrown balls is

$$\tau_n^m = \sum_{l=\lceil rm \rceil - W_n^m(1)m}^{\lceil rm \rceil - 1} Y_l, \tag{2.1}$$

where $Y_l$, $l \in \mathbb{N}$ are independent random variables with geometrical distributions on $\mathbb{N}^*$ with respective parameters $l/m$, i.e. $\mathbb{P}(Y_l = k) = (l/m)^{k-1}(1-l/m)$, $k \geq 1$.

Let $F$ be defined on $\mathcal{P} = \left\{w \in \mathbb{R}_+^{C+1}, \sum_{i=0}^{C} w(i) = 1\right\}$ by

$$F(w) = T_C\left(s(w) * \mathcal{P}_{\lambda(w)}\right) \tag{2.2}$$

where

$$\begin{aligned} &s : w \mapsto (w(0)+w(1), w(2), \ldots, w(C), 0) \text{ on } \mathcal{P} \\ &T_C : \mathcal{P}(\mathbb{N}) = \left\{(w_n)_{n\in\mathbb{N}}, \sum_{i=0}^{+\infty} w_i = 1\right\} \to \mathcal{P},\ w \mapsto \left(w(0), \ldots, w(C-1), \sum_{i\geq C} w(i)\right) \\ &\lambda : \mathcal{P} \to \mathbb{R}^+,\ w \mapsto \log\left(1 + \frac{w(1)}{1-r}\right) \end{aligned}$$

and $\mathcal{P}_\lambda$ is the Poisson distribution with parameter $\lambda$. Notice that $F$ maps $\mathcal{P}$ to itself and, also by definition of $\lambda$, $\mathcal{P}^{(r)} \stackrel{\text{def}}{=} \left\{w \in \mathbb{R}_+^{C+1}, \sum_{i=0}^{C} w(i) = 1 \text{ and } \sum_{i=1}^{C} w(i) = r\right\}$ to itself.

### 2.1.1 Convergence to a dynamical system

We prove the convergence of $(W_n^m)_{n\in\mathbb{N}}$ to the dynamical system given by $F$ as $m$ tends to $+\infty$. The following lemma is the key argument. The uniform convergence stated below appears as the convenient way to express the convergence of $P_m(w,.)$ to $\delta_{F(w)}$ in order to prove both the convergence of $(W_n^m)_{n\in\mathbb{N}}$, and, later on, the convergence of the stationary distributions.

Define $\| x \| = \sup_{i=0}^{C} |x_i|$ for $x \in \mathbb{R}^{C+1}$.

**Lemme 1.** *For $\varepsilon > 0$,*

$$\sup_{w\in\mathcal{P}_m^{(r)}} P_m(w, \{w' \in \mathcal{P}_m^{(r)} : ||w' - F(w)|| > \varepsilon\}) \underset{m\to+\infty}{\longrightarrow} 0.$$

*Démonstration.* The first step is to prove that, for $\varepsilon > 0$,

$$\sup_{w\in\mathcal{P}_m^{(r)}} \mathbb{P}_w\left(\left|\frac{\tau_1^m}{m} - \lambda(w)\right| > \varepsilon\right) \underset{m\to\infty}{\to} 0 \tag{2.3}$$

where $\lambda(w) = \log\left(1 + \frac{w(1)}{1-r}\right)$ and $\mathbb{P}_w(.)$ denotes $\mathbb{P}(.|W_0^m = w)$. By Bienaymé-Chebyshev's inequality, it is enough to prove that

$$\sup_{w\in\mathcal{P}_m^{(r)}} \left|\mathbb{E}_w\left(\frac{\tau_1^m}{m}\right) - \lambda(w)\right| \underset{m\to\infty}{\to} 0 \tag{2.4}$$

and

$$\sup_{w\in\mathcal{P}_m^{(r)}} \mathrm{Var}_w\left(\frac{\tau_1^m}{m}\right) \underset{m\to\infty}{\to} 0. \tag{2.5}$$

By equation (2.1), as $\mathbb{E}(Y_l) = 1/(1-l/m)$, using a change of index,

$$\mathbb{E}_w\left(\frac{\tau_1^m}{m}\right) = \sum_{l=\lceil rm\rceil - w(1)m}^{\lceil rm\rceil - 1} \frac{1}{m-l} = \sum_{j=m-\lceil rm\rceil+1}^{m-\lceil rm\rceil+w(1)m} \frac{1}{j}. \tag{2.6}$$

A comparison with integrals leads to the following inequalities :

$$\log \frac{1-\frac{\lceil rm\rceil}{m}+w(1)+\frac{1}{m}}{1-\frac{\lceil rm\rceil}{m}+\frac{1}{m}} \leq \mathbb{E}_w\left(\frac{\tau_1^m}{m}\right) \leq \log \frac{1-\frac{\lceil rm\rceil}{m}+w(1)}{1-\frac{\lceil rm\rceil}{m}}.$$

It is then easy to show that the two extreme terms tend to $\lambda(w)=\log(1+w(1)/(1-r))$, uniformly in $w(1)\in[0,1]$. This gives (2.4). For (2.5), as $\mathrm{Var}(Y_l)=(l/m)/(1-l/m)^2$, by the same change of index,

$$\mathrm{Var}_w\left(\frac{\tau_1^m}{m}\right)=\frac{1}{m}\sum_{j=m-\lceil rm\rceil+1}^{m-\lceil rm\rceil+w(1)m}\frac{m-j}{j^2}=\sum_{j=m-\lceil rm\rceil+1}^{m-\lceil rm\rceil+w(1)m}\frac{1}{j^2}-\frac{1}{m}\mathbb{E}_w\left(\frac{\tau_1^m}{m}\right). \tag{2.7}$$

The first term of the right-hand side is bounded independently of $w$ by $\sum_{j=m-\lceil rm\rceil+1}^{+\infty}1/j^2$, which tends to 0 as $m$ tends to $+\infty$. The second term tends to 0 uniformly in $w$ using (2.4) together with the uniform bound $\lambda(w)\leq\log(1+1/(1-r))$.

To obtain the lemma, it is then sufficient to prove that, for each $\varepsilon>0$,

$$\sup_{w\in\mathcal{P}_m^{(r)}}\mathbb{P}_w\left(||W_1^m-F(w)||>\varepsilon,\left|\frac{\tau_1^m}{m}-\lambda(w)\right|\leq\frac{\varepsilon}{2}\right)\underset{m\to\infty}{\to}0. \tag{2.8}$$

Since $W_1^m$ and $F(w)$ are probability measures on $\{0,\ldots,C\}$, to get (2.8), it is sufficient to prove that for $j\in\{0,\ldots,C-1\}$,

$$\sup_{w\in\mathcal{P}_m^{(r)}}\mathbb{P}_w\left(|W_1^m(j)-F(w)(j)|>\varepsilon,\left|\frac{\tau_1^m}{m}-\lambda(w)\right|\leq\frac{\varepsilon}{2}\right)\underset{m\to\infty}{\to}0. \tag{2.9}$$

Let $w\in\mathcal{P}_m^{(r)}$. Define the following random variables : For $1\leq i\leq m$, $N_i^m$ (respectively $\widetilde{N}_i^m(w)$) is the number of additional balls in urn $i$ when $\tau_1^m$ (respectively $m\lambda(w)$) new balls are thrown in the $m$ urns. One can construct these variables from the same sequence of balls (i.e. of i.i.d. uniform on $\{1,\ldots,m\}$ random variables), meaning that balls are thrown in the same locations for both operations until stopping. This provides a natural coupling for the $N_i$'s and $\widetilde{N}_i$'s. Let $j\in\{0,\ldots,C-1\}$ be fixed. Given $W_0^m=w$, as $j\leq C-1$, the capacity constraint does not interfere and $W_1^m(j)$ can be represented as

$$W_1^m(j)=\frac{1}{m}\sum_{k=0}^{j}\sum_{i\in I_{w,k}^m}1_{\{N_i^m=j-k\}} \tag{2.10}$$

where $I_{w,k}^m$ is the set of urns with $k$ balls in some configuration of $m$ urns with distribution $s(w)$, so that $\mathrm{card}\, I_{w,k}^m=ms(w)(k)$. The sum over $i$ is exactly the number of urns that contains $k$ balls after the removing of one ball per urn, and having $j$ balls after new balls have been thrown. By coupling, on the event $\{W_0^m=w,\,|\tau_1^m/m-\lambda(w)|\leq\varepsilon/2\}$, the following is true :

$$\mathrm{card}\{i,N_i^m\neq\widetilde{N}_i^m(w)\}\leq\frac{\varepsilon}{2}m \tag{2.11}$$

thus, denoting $\widetilde{W}_1^m(j)=\frac{1}{m}\sum_{k=0}^{j}\sum_{i\in I_{w,k}^m}1_{\{\widetilde{N}_i^m(w)=j-k\}}$, on the same event,

$$|W_1^m(j)-\widetilde{W}_1^m(j)|\leq\frac{\varepsilon}{2}.$$

To prove equation (2.9), it is then sufficient to show that

$$\sup_{w\in\mathcal{P}_m^{(r)}} \mathbb{P}_w\left(|\widetilde{W}_1^m(j)-F(w)(j)|>\varepsilon\right)\underset{m\to\infty}{\longrightarrow}0.$$

This will result from

$$\begin{array}{rcl}\sup_{w\in\mathcal{P}_m^{(r)}}|\mathbb{E}_w(\widetilde{W}_1^m(j))-F(w)(j)| & \underset{m\to\infty}{\longrightarrow} & 0 \quad \text{and}\\ \sup_{w\in\mathcal{P}_m^{(r)}}\mathrm{Var}_w(\widetilde{W}_1^m(j)) & \underset{m\to\infty}{\longrightarrow} & 0.\end{array} \tag{2.12}$$

which is quite standard to prove. The key argument, with classical proof, is the following : If $L_i^m$ is the number of balls in urn $i$ when throwing $m\lambda$ balls at random in $m$ urns, if $0<a<b$, then, for all $(i_1,i_2)\in\mathbb{N}^2$,

$$\begin{aligned}&(i)\ \sup_{\lambda\in[a,b]}|\mathbb{P}(L_1^m=i_1)-\mathcal{P}_\lambda(i_1)|\underset{m\to\infty}{\longrightarrow}0,\\ &(ii)\ \sup_{\lambda\in[a,b]}|\mathbb{P}(L_1^m=i_1,L_2^m=i_2)-\mathcal{P}_\lambda(i_1)\mathcal{P}_\lambda(i_2)|\underset{m\to\infty}{\longrightarrow}0.\end{aligned}$$

It is applied since $\lambda(w)\in[0,\log(1+1/(1-r)]$. It ends the proof. □

**Proposition 1.** *If $W_0^m$ converges in distribution to $w_0\in\mathcal{P}_m^{(r)}$ then $(W_n^m)_{n\in\mathbb{N}}$ converges in distribution to the dynamical system $(w_n)_{n\in\mathbb{N}}$ given by the recursion $w_{n+1}=F(w_n)$, $n\in\mathbb{N}$.*

*Démonstration.* Assume that $W_0^m$ converges in distribution to $w_0\in\mathcal{P}_m^{(r)}$. Convergence of $(W_0^m,\dots,W_n^m)$ can be proved by induction on $n\in\mathbb{N}$. By assumption it is true for $n=0$. Let us just prove it for $n=1$, the same arguments holding for general $n$, from the assumed property for $n-1$. Let $g$ be continuous on the (compact) set $\mathcal{P}_m^{(r)2}$. Since the distribution $\mu_m$ of $W_0^m$ has support in $\mathcal{P}_m^{(r)}$,

$$\begin{aligned}\mathbb{E}\left(g(W_0^m,W_1^m)\right)&=\int_{\mathcal{P}_m^{(r)2}}g(w,w')P_m(w,dw')d\mu_m(w)\\ &=\int_{\mathcal{P}_m^{(r)}}\int_{\mathcal{P}_m^{(r)}}\left(g(w,w')P_m(w,dw')-g(w,F(w))\right)d\mu_m(w)+\int_{\mathcal{P}_m^{(r)}}g(w,F(w))d\mu_m(w).\end{aligned}$$

Since $g(.,F(.))$ is continuous on $\mathcal{P}_m^{(r)}$ ($F$ being continuous as can be easily checked), the last integral converges to $g(w_0,w_1)$ by assumption (or case $n=0$). The first term is bounded in modulus, for each $\eta>0$, by

$$\sup_{w\in\mathcal{P}_m^{(r)}}\left|\int_{\mathcal{P}_m^{(r)}}g(w,w')P_m(w,dw')-g(w,F(w))\right|\le 2\,\|g\|_\infty\sup_{w\in\mathcal{P}_m^{(r)}}P_m\left(w,\left\{w'\in\mathcal{P}_m^{(r)},\|w'-F(w)\|>\varepsilon\right\}\right)+\eta$$

where $\varepsilon$ is associated to $\eta$ by the uniform continuity of $g$ on $\mathcal{P}_m^{(r)2}$. By Lemma 1, this is less than $2\eta$ for $m$ sufficiently large. Thus, as $m$ tends to $+\infty$,

$$\mathbb{E}\left(g(W_0^m,W_1^m)\right)\to g(w_0,w_1).$$

□

### 2.1.2 Convergence of invariant measures

Let, for $m \in \mathbb{N}$, $\pi_m$ be the stationary distribution of $(W_n^m)_{n\in\mathbb{N}}$. Define $P$ as the transition on $\mathcal{P}^{(r)}$ given by $P(w,.) = \delta_{F(w)}$.

**Proposition 2.** *Any limiting point $\pi$ of $(\pi_m)_{m\in\mathbb{N}}$ is a probability measure on $\mathcal{P}^{(r)}$ which is invariant for $P$ i.e. that satisfies $F(\pi) = \pi$.*

*Démonstration.* A classical result states that, if $P$ and $P_m$, $m \in \mathbb{N}$, are transition kernels on some metric space $E$ such that, for any bounded continuous $f$ on $E$, $Pf$ is continuous and $P_m f$ converges to $Pf$ uniformly on $E$ then, for any sequence $(\pi_m)$ of probability measures such that $\pi_m$ is invariant under $P_m$, any limiting point of $\pi_m$ is invariant under $P$. Indeed, for any $m$ and any bounded continuous $f$, $\pi_m P_m f = \pi_m f$. If a subsequence $(\pi_{m_p})$ converges weakly to $\pi$, then $\pi_{m_p} f$ converges to $\pi f$. Writing $\pi_{m_p} P_{m_p} f = \pi_{m_p} Pf + \pi_{m_p}(P_{m_p} f - Pf)$, since $Pf$ continuous (and bounded since $f$ is), the first term $\pi_{m_p} Pf$ converges to $\pi Pf$ and the second term tends to 0 by uniform convergence of $P_m f$ to $Pf$. Equation $\pi_{m_p} P_{m_p} f = \pi_{m_p} f$ thus gives, in the limit, $\pi Pf = \pi f$ for any bounded continuous $f$.

Here the difficulty is that the $P_m$'s and $P$ are transitions on $\mathcal{P}_m^{(r)}$ and $\mathcal{P}^{(r)}$, which are in general disjoint. To solve this difficulty, extend artificially $P_m$ and $P$ to $\mathcal{P}$ by setting :

$$\begin{aligned} P_m(w,.) &= \delta_{F(w)} \quad \text{for } w \in \mathcal{P} \setminus \mathcal{P}_m^{(r)} \\ P(w,.) &= \delta_{F(w)} \quad \text{for } w \in \mathcal{P} \setminus \mathcal{P}^{(r)}. \end{aligned}$$

The proposition is then deduced from the classical result if we prove that, for each $f$ continuous on $\mathcal{P}$ (notice that then $Pf = f \circ F$ is continuous),

$$\sup_{w\in\mathcal{P}_m^{(r)}} |P_m f(w) - f(F(w))| \underset{m\to\infty}{\longrightarrow} 0,$$

which is straightforward from Lemma 1. The fact that the support of $\pi$ is in $\mathcal{P}^{(r)}$ is deduced from the portmanteau theorem (see Billingsley [14] p.16) using the sequence of closed sets

$$\mathcal{P}^{(r),n} = \left\{ w \in \mathcal{P}, r \le \sum_{i=1}^{C} w(i) \le r + \frac{1}{n} \right\}.$$

□

The fixed points of the dynamical system are the probability measures $w$ on $\mathcal{P}^{(r)}$ such that

$$w = F(w) = T_C(s(w) * \mathcal{P}_{\lambda(w)})$$

where $\lambda(w) = \log(1 + w(1)/(1-r))$. This is exactly the invariant measure equation for the number of customers just after completion times in an $M/G/1/C$ queue with arrival rate $\lambda(w)$ and service times 1, so that it is equivalent to

$$w = \mu_{\lambda(w)} \tag{2.13}$$

where $\mu_\lambda$ (respectively $\nu_\lambda$) is the limiting distribution of the process of the number of customers in an $M/G/1/C$ (respectively $M/G/1/\infty$) queue with arrival rate $\lambda$ and service times 1.

Indeed, it is well-known that this queue has a limiting distribution for $\lambda \in \mathbb{R}^+$ (respectively $0 \le \lambda < 1$) which is the invariant probability measure of the embedded Markov chain of the number

of customers just after completion times. The balance equations here reduce to a recursion system, so that, even when $\lambda \geq 1$, $\nu_\lambda$ is well defined up to a multiplicative constant (which can not be normalized into a probability measure in this case). Moreover, $\nu_\lambda$ is given by the Pollaczek-Khintchine formula for its generating function :

$$\sum_{n\in\mathbb{N}} \nu_\lambda(n)u^n = \nu_\lambda(0)\frac{g_\lambda(u)(u-1)}{u-g_\lambda(u)}, \quad \text{for } |u|<1 \tag{2.14}$$

where $g_\lambda(u) = e^{-\lambda(1-u)}$ and for $\lambda < 1$, $\nu_\lambda(0) = 1-\lambda$ (see for example Robert [68] p176-177). Notice that $\nu_\lambda(n)$ ($n \in \mathbb{N}$) has no closed form. For example, the expressions of the first terms are

$$\begin{aligned}
\nu_\lambda(1) &= \nu_\lambda(0)(e^\lambda - 1),\\
\nu_\lambda(2) &= \nu_\lambda(0)e^\lambda(e^\lambda - 1 - \lambda),\\
\nu_\lambda(3) &= \nu_\lambda(0)e^\lambda\left(\frac{\lambda(\lambda+2)}{2} - (1+2\lambda)e^\lambda + e^{2\lambda}\right)
\end{aligned} \tag{2.15}$$

where $\nu_\lambda(0) = 1-\lambda$ if $\lambda < 1$. For the $M/G/1/C$ queue,

$$\mu_\lambda(i) = \frac{\nu_\lambda(i)}{\sum_{l=0}^{C}\nu_\lambda(l)}, \ i \in \{0,\dots,C\}. \tag{2.16}$$

The following proposition characterizes the fixed points of $F$.

**Proposition 3.** *$F$ defined by (2.2) has one unique fixed point, denoted by $\overline{w}$, in $\mathcal{P}^{(r)}$ given by the limiting distribution $\mu_{\overline{\lambda}}$ of the number of customers in an $M/G/1/C$ queue with arrival rate $\overline{\lambda}$ and service times 1, where $\overline{\lambda}$ is determined by the implicit equation $\mu_{\overline{\lambda}}(0) = 1-r$ which is equivalent to*

$$\overline{\lambda} = \log\left(1 + \frac{\mu_{\overline{\lambda}}(1)}{1-r}\right), \tag{2.17}$$

*where $\mu_\lambda$ is given by (2.16) and $\nu_\lambda$ by the Pollaczek-Kintchine formula (2.14).*

Notice that, moreover,

$$r \leq \overline{\lambda} \leq -\log(1-r).$$

The upper bound on $\overline{\lambda}$, obtained from equation (2.17) using $\mu_{\overline{\lambda}}(1) \leq r$ just says that the stationary mean number of balls between two shifts is less than the mean number of balls thrown until the first shift (starting with empty urns). Moreover $\overline{\lambda} \geq r$, which is clear if $\overline{\lambda} \geq 1$, and obtained if $\overline{\lambda} < 1$ writing (2.16) for $i = 0$ and using $\sum_{l=0}^{C}\nu_{\overline{\lambda}}(l) \leq 1$ in this equation. This is exactly the fact that the asymptotic stationary mean number of balls $\overline{\lambda}m$ arriving between two shift times is greater than the number of removed balls at each shift, which is $\lceil rm \rceil$. It is due to the losses under the capacity limit $C$.

*Démonstration.* Only the existence and uniqueness result remains to prove. According to (2.13), $w$ is some fixed point if and only if it is a fixed point of the function

$$\begin{aligned}
\mathcal{P}^{(r)} &\longrightarrow \mathcal{P}^{(r)}\\
w &\longmapsto \mu_{\lambda(w)}
\end{aligned}$$

with $\lambda(w) = \log(1 + w(1)/(1-r))$. This function being continuous on the convex compact set $\mathcal{P}^{(r)}$, by Brouwer's theorem, it has a fixed point. To prove uniqueness, let $w$ and $w'$ be two fixed points of $F$ in $\mathcal{P}^{(r)}$. By definition of $\mathcal{P}^{(r)}$,

$$\mu_{\lambda(w)}(0) = \mu_{\lambda(w')}(0) = 1 - r. \tag{2.18}$$

A coupling argument shows that, if $\lambda \leq \lambda'$ then $\mu_\lambda$ is stochastically dominated by $\mu_{\lambda'}$, and in particular,

$$\mu_\lambda(0) + \mu_\lambda(1) \geq \mu_{\lambda'}(0) + \mu_{\lambda'}(1). \tag{2.19}$$

It can then be deduced that $\lambda(w) = \lambda(w')$. Indeed, if for example $\lambda(w) < \lambda(w')$, by equations (2.18) and (2.19),

$$\mu_{\lambda(w)}(1) \geq \mu_{\lambda(w')}(1).$$

thus, using (2.15) together with (2.18),

$$\lambda(w) = \log\left(1 + \frac{\mu_{\lambda(w)}(1)}{1-r}\right) \geq \lambda(w') = \log\left(1 + \frac{\mu_{\lambda(w')}(1)}{1-r}\right)$$

which contradicts $\lambda(w) < \lambda(w')$. One finally gets $\lambda(w) = \lambda(w')$, and then by equation (2.13), $w = w'$. □

A Lyapunov function for the dynamical system given by $F$ on $\mathcal{P}^{(r)}$ is a function $g \geq 0$ on $\mathcal{P}^{(r)}$ such that, for each $w \in \mathcal{P}^{(r)}$, $g(F(w)) \leq g(w)$ with equality if and only if $w$ is the fixed point of $F$. In the particular case $C = 2$, a Lyapunov function can be exhibited, resulting from a contracting property of $F$ in this case.

Indeed, restricted to $\mathcal{P}^{(r)}$, $F$ is here given by :

$$w = (1-r, w(1), w(2) = r - w(1)) \longmapsto F(w) = \left(1-r, (1-r)\left[\log\left(1 + \frac{w(1)}{1-r}\right) + \frac{r - w(1)}{1-r+w(1)}\right],\right.$$
$$\left. 1 - (1-r)\left[\log\left(1 + \frac{w(1)}{1-r}\right) + \frac{1}{1-r+w(1)}\right]\right).$$

$\mathcal{P}^{(r)}$ is some one dimensional subvariety of $\mathbb{R}^3$, so that any $w \in \mathcal{P}^{(r)}$ can be identified with its second coordinate $w(1) \in [0, r]$, or equivalently with $\lambda(w) = \log(1 + w(1)/(1-r)) \in [0, \log(1/(1-r))]$.

Using this last parametrization of $\mathcal{P}^{(r)}$, it is easy to show that $F$ rewrites as $G$, mapping the interval $I = [0, \log(1/(1-r))]$ to itself and defined, for $\lambda \in I$, by $G(\lambda) = \log\left(\lambda + \frac{e^{-\lambda}}{1-r}\right)$.

An elementary computation shows that $G$ has derivative on $I$ taking values in the interval $]-1, 0]$, which gives the already known existence and uniqueness of a fixed point $\overline{\lambda}$ for $G$ (or $F$, both assertions being equivalent, and $\overline{\lambda}$ being equal to $\lambda(\overline{w})$). Moreover, the following inequality holds for $\lambda \in I$ :

$$|G(\lambda) - \overline{\lambda}| \leq |\lambda - \overline{\lambda}|,$$

equality occurring only at $\lambda = \overline{\lambda}$. As a result, $g$ defined on $\mathcal{P}^{(r)}$ by

$$g(w) = |\lambda(w) - \overline{\lambda}| = |\lambda(w) - \lambda(\overline{w})| = \left|\log \frac{1-r+w(1)}{1-r+\overline{w}(1)}\right|,$$

is a Lyapunov function for the dynamical system defined by $F$.

For $C > 2$, we conjecture the existence of such a $g$.

**Theorem 1.** *Assume that a Lyapunov function exists for the dynamical system given by $F$ on $\mathcal{P}^{(r)}$ then, as $m$ tends to $+\infty$, the invariant measure of $(W_n^m)_{n\in\mathbb{N}}$ converges to $\delta_{\bar{w}}$ where $\bar{w}$ is the unique fixed point of $F$. Thus the following diagram commutes,*

$$\begin{array}{ccc} (W_n^m)_{n\in\mathbb{N}} & \xrightarrow[n\to+\infty]{(d)} & W_\infty^m \\ {\scriptstyle m\to+\infty}\downarrow{\scriptstyle (d)} & & \downarrow{\scriptstyle (d)} \\ (w_n)_{n\in\mathbb{N}} & \longrightarrow & \bar{w} \end{array}$$

*Démonstration.* We prove that $\delta_{\bar{w}}$ is the unique invariant measure $\pi$ of $P$ with support in $\mathcal{P}^{(r)}$. Let $g$ be the Lyapunov function for $F$ on $\mathcal{P}^{(r)}$. $\pi$ is $P$-invariant, thus $\pi P = \pi$ and $\pi P g = \pi g$ which can be rewritten $\int (g\circ F - g)d\pi = 0$. This implies that $g = g\circ F$ holds $\pi$ almost surely because $g - g\circ F \geq 0$. Equality being only true at $\bar{w}$, $\pi$ has support in $\{\bar{w}\}$. □

## 2.2 A more general model

### 2.2.1 Mice with general size distribution

Let $(W_n^m)_{n\in\mathbb{N}}$ be the sequence of vectors giving the proportions of urns at $0,\ldots,C$ just before the $n$th shift time in a model where balls are thrown by batches. The balls in a batch are thrown together in a unique urn chosen at random among the $m$ urns. The $i$th batch is composed with $S_i$ balls and $(S_i)_{i\in\mathbb{N}}$ is a sequence of i.i.d. random variables distributed as a random variable $S$ on $\mathbb{N}^*$ with support containing 1. Let $\phi$ be the generating function of $S$. The quantity $S_i$ is called the size of batch $i$. The dynamics is the same : If, before the $n$th shift time, the state is $w \in \mathcal{P}_m^{(r)}$, it first becomes $s(w)$ and then a number $\tau_n^m$ defined by (2.1) of successive batches are thrown in urns until $\lceil rm \rceil$ urns are non empty. The model generalizes the previous one obtained for $S = 1$.

Let $F$ be defined on $\mathcal{P}$ by

$$F(w) = T_c(s(w) * C_{\lambda(w),S}) \tag{2.20}$$

where $T_C$, $\lambda$ and $S$ are already defined and $C_{\lambda(w),S}$ is a compound Poisson distribution i.e. the distribution of the random variable

$$Y = \sum_{i=1}^{X} S_i \tag{2.21}$$

where $X$ is independent of $(S_i)_{i\in\mathbb{N}}$ with Poisson distribution of parameter $\lambda(w)$.

We mimic the arguments in Section 2.1 to obtain the convergence of the stationary distribution of the ergodic Markov chain $(W_n^m)_{n\in\mathbb{N}}$ as $m$ tends to $+\infty$ to a Dirac measure at the unique fixed point of $F$. Propositions 1 and 2 hold. The fixed points of $F$ are described in the following proposition.

**Proposition 4.** *$F$ defined for $w \in \mathcal{P}$ by*

$$F(w) = T_c(s(w) * C_{\lambda(w),S})$$

*has a unique fixed point on $\mathcal{P}^{(r)}$ which is exactly the invariant measure $\mu_{\bar{\lambda}}$ of the number of customers in a $M/G/1/C$ queue with batches of customers arriving according to a Poisson process*

*with intensity* $\bar{\lambda}$, *batch sizes being i.i.d. distributed as* $S$ *with generating function* $\phi$ *and service times* 1, *where* $\bar{\lambda}$ *is determined by the implicit equation*

$$\mu_{\bar{\lambda}}(0) = 1 - r$$

*which is equivalent to*

$$\bar{\lambda} = \log\left(1 + \frac{\mu_{\bar{\lambda}}(1)}{1-r}\right)$$

*where for* $i \in \{0, \ldots, C\}$,

$$\mu_\lambda(i) = \frac{\nu_\lambda(i)}{\sum_{l=0}^{C} \nu_\lambda(l)}$$

*and* $\nu_\lambda$ *is given by*

$$\sum_{n=0}^{+\infty} \nu_\lambda(n) u^n = \nu_\lambda(0) \frac{g \circ \phi(u)(u-1)}{u - g \circ \phi(u)}, \quad |u| < 1$$

*where* $g_\lambda(u) = e^{-\lambda(1-u)}$ *and* $\nu_\lambda(0) = 1 - \lambda \mathbb{E}(S)$ *when* $\lambda < 1$.

Recall that the first terms of $\nu_\lambda$ are given by

$$\nu_\lambda(1) = \nu_\lambda(0)(e^\lambda - 1)$$
$$\text{and } \nu_\lambda(2) = \nu_\lambda(0) e^\lambda (e^\lambda - 1 - \lambda \mathbb{P}(S = 1))$$

where $\nu_\lambda(0) = 1 - \lambda \mathbb{E}(S)$ when $\lambda < 1$, which generalizes the previous expressions. For $C = 2$, the Lyapunov function defined when $S = 1$ still works. Furthermore, for $C > 2$, we assume the existence of a Lyapunov function for $F$. Theorem 1 still holds.

### 2.2.2 A multi-stage filter

The filter is previously supposed to have only one stage. Let now assume that the filter has $k$ stages of $m$ counters each. The natural model then consists of $k$ sets of $m$ urns where, when a ball is thrown, $k$ copies of this ball are sent simultaneously and independently into the $k$ stages, each falling at random in one of the $m$ urns of its set. If some ball hits an urn with $C$ balls, then it is rejected. Moreover, when the proportion of non-empty urns in the *whole* filter reaches $r$, then one ball is removed from each non-empty urn : This is called a shift.

The previous analysis extends with one main difference : When the system is initialized at some state $(w_j(i), 1 \le j \le k, 0 \le i \le C)$, where $w_j(i)$ is the proportion of urns with $i$ balls in stage $j$, the number $\tau_1^m$ of balls thrown in each stage before the next shift is now asymptotically equivalent to $\lambda(w)m$, where $w = (w(i), 0 \le i \le C)$ here gives the *global* proportions of urns in each possible state in the whole filter, that is $w(i) = \frac{1}{k}\sum_{j=1}^{k} w_j(i)$ for $0 \le i \le C$. Thus $\lambda$ is the same function as for the one-filter case, here evaluated at the global proportions :

$$\lambda(w) = \log\left(1 + \frac{\sum_{j=1}^{k} w_j(1)}{k(1-r)}\right).$$

The one-stage proof of (3) is however not reproducible here, due to the lack of a representation of $\tau_1^m$ analogous to (1) of Section 1.

Another proof can be written. The alternative argument is provided by noticing that when $\alpha m$ balls per stage are thrown, with $\alpha \notin [\lambda(w) - \varepsilon, \lambda(w) + \varepsilon]$ (using the same arguments (i) and

(ii) as in Section 1), the empirical distributions of the urns at each stage are precisely known (for large $m$) and do not correspond to the global proportion $r$ of non-empty urns.

Once the (uniform) convergence of $\tau_1^m/m$ is established, the proof then proceeds along the same lines as for $k=1$ (the same reasoning holding for each stage).

Notice however that the Markov property does not hold for the process of global proportions at shift times, so that convergence in distribution is proved for the process of proportions detailed by stage, then inducing convergence.

## 2.3 Discussions

### 2.3.1 Synthesis : false positives and false negatives

From a practical point of view, the main results are Propositions 3 and 4. Given some size distribution for the flows (the generating function $\phi$ of Section 2.2), these propositions show how the values of the counters can be computed from the different parameters of the algorithm, since these values are encoded by the fixed point $\overline{w}$ of $F$ : according to Theorem 1, $\overline{w}$ is the state reached in the stationary regime when there is one stage and also when there are several stages (see Subsection 2.2.2 : one has $\sum_{j=1}^{k} \overline{w}_j(C) = k\overline{w}(C)$). Moreover, the convergence is experimentally really fast (see the remark below), which ensures that in practice the algorithm lives in the stationary phase. The component $\overline{w}(i)$ of $\overline{w}$ gives the approximate proportion of counters having value $i$ in the whole Bloom filter. $\overline{\lambda}$ is the number of packets that arrive between two shifts. $\overline{w}$ and $\overline{\lambda}$ are respectively related to the number of false positives and to the number of false negatives :

The probability that a packet is a false positive is less than $(\overline{w}(C-1)+\overline{w}(C))^k$ since, in stage $j$, the probability to hit a counter at height $C$ is at most $\overline{w}_j(C-1)+\overline{w}_j(C)$ and

$$\prod_{j=1}^{k}(\overline{w}_j(C-1)+\overline{w}_j(C)) \leq (\overline{w}(C-1)+\overline{w}(C))^k.$$

The quantity $m\overline{\lambda}$ is the time (number of packets) between two shifts, which is connected to the number of false negatives according to the discussion “False positive and false negative” of the Introduction.

### 2.3.2 Implementation and tests

The algorithm has been implemented with an improvement called the *min-rule* already proposed in [34]. Instead of increasing the $k$ counters, an arriving packet is incrementing only the counters among $k$ having the minimum value. Analytically more difficult to study, the algorithm should perform better : Heuristically, more flows are needed to reach high values of the counters inducing fewer false positives ; moreover, the time between two shifts is longer and hence the number of false negatives is decreased. It has been tested against on two ADSL traffic traces from France Telecom, involving millions of flows. The performance of the algorithm is evaluated comparing the real number of elephants with the value estimated by the algorithm. Even under the min-rule, the algorithm performs well only if $r$ stays under a critical value $r_c$, closely dependent on the mice distribution.

Simulations have been processed with a one-stage filter with flows of size 1 to evaluate the transient phase duration. It appears that the number of shifts to reach the stationary phase is not much greater than $C$. Such a result on the speed of convergence seems however theoretically out of reach.

# 3

# Analysis of a Bloom filter algorithm via the supermarket model

**Contents**



This chapter also deals with the problem of identifying elephants in the Internet Traffic. It is devoted to a further analysis of the algorithm based on Bloom Filter. This algorithm uses a so-called min-rule which can be described as in the supermarket model. This model consists of joining the shortest queue among $k$ queues selected at random in a large number of $m$ queues. In case of equality, one of the shortest queues is chosen at random. An analysis of a simplified model gives an insight into the error generated by the algorithm for the estimation of the number of the elephants. The same arguments are extended. Even if some results are for the moment out of reach, one could conjecture the convergence of the empirical distribution of the filter counters to a deterministic limit. Its characterization is analytically more complicated and numerically more difficult to obtain.

## 3.1 Introduction

Chapter 2 presents a theoretical analysis of the algorithm proposed by Azzana in [6]. The objective is to estimate the error generated by the algorithm for the estimation of the number of elephants. The analytical study does not take into account the min-rule consisting in incrementing only the counters among $k$ having the minimum value, for an arriving packet.

In this chapter, we focus on the analysis of the min-rule. For this purpose, in order to obtain a model more standard to analyze, the algorithm proposed by Azzana in has been slightly modified. We consider now just one filter and $k$ hashing functions. (Keep in mind the case $k = 2$). An arriving packet increments the smallest counter among the $k$ associated counters. In case of equality, only one counter is incremented at random. In this way, every packet increments exactly one counter. Note that an elephant is here defined as a flow with at least $K$ packets where $K$ is in pratice equal to 20. A flow is declared as an elephant when its smallest associated counter reaches $C = K/k$. The same refreshment mechanism is maintained with a threshold $r$ of about 50%. The basic idea is that when the filter is not overloaded, in general, for each arriving packet of a given flow, one of the $k$ counters will be incremented in an alternative way. It means that the $k$ counters will have almost the same values and when the smallest one reaches $C$, the corresponding flow has a total size of about $K = kC$ packets ($C$ packets hitting each counter).

The advantage of this new algorithm is that each arriving packet increments exactly one counter. In this case, the way to increment the counters is exactly, in a system of $m$ queues, the way a customer joins the shortest queue among $k$ queues chosen at random, ties being solved at random. This model, called *supermarket model* by Mitzenmacher [62] and Luczak and McDiarmid [58], also known as a load-balancing model, or model *with choice*, has been extensively studied in the literature because of its numerous useful applications. In computer science, the central result is stated in a pioneer paper by Azar *et al.* [5], then by Miztenmacher [62], and, by other arguments, by Luczak and McDiarmid [58] for a discrete time model when $n$ balls are thrown into $n$ urns with the choice. It is proved that, with probability tending to 1 as $n$ gets large, the maximum load of an urn is $\log n/\log\log n + O(1)$ when $k = 1$ and $\log\log n/\log k + O(1)$ if $k \geq 2$. Luczak and McDiarmid, in continuous time related models with the choice, explore the concentration of the maximum queue length (see [58], [57]). But the model had also already been studied in Mitzenmacher [62], Vvedenskaya *et al.* [72], Graham [47] and others for mean-field limit theorems. In [62] and [72], a functional law of large numbers is stated : The process of the vector of the tail proportions of queues converges in distribution as $m$ tends to infinity to the unique solution of a differential system. For $k \geq 2$, the differential equation has a unique equilibrium point $u^\rho(i) = \rho^{(k^i-1)/(k-1)}$ for a throughput $\rho < 1$. It means that, when $k \geq 2$, the tail probabilities of the queue length decrease drastically. In Vvedenskaya and Suhov [73], variants of the choice policy and general service time distributions are investigated. Graham in [47] proves the convergence of the invariant measures to the Dirac measure at $u^\rho$. In other words, when $m$ is large, the stationary vector of the proportions of queues with $k$ customers is essentially deterministic and given by this limit.

The aim of this chapter is to analyze the min-rule via the supermarket model and to evaluate the performance of the new proposed algorithm. In particular, we want to calculate the error generated by the algorithm on the estimation of number of elephants. Notice that this error is due to both false negatives (missed elephants) and false positives (mice considered as elephants). There is a trade-off between the proportion of false positives and false negatives. Moreover this trade-off depends on the target applications. For example, for attacks detection, it will be better to have false positives, say suspicious flows, than to miss an elephant, i.e. an attack. The operator can further easily make the difference between anomalies and attacks. On the contrary, in order

to count elephants, it is reasonable to avoid having too much false positives because mice are numerous and a small proportion of false positives implies a non negligible error on the number of elephants. Let us first focus on false positives. To be declared as an elephant, a mouse must be hashed to one among $k$ counters greater than $C$ after this operation. So the proportion of such counters is a good parameter to investigate in order to evaluate false positives.

The most part of the chapter is the analysis of a simple model where the flows are mice of size one. It is relevant because most of the flows are mice so collisions between flows are mainly due to collisions between mice. It turns out that the probability that a mouse is detected as an elephant is bounded by the probability that a given counter is greater than $C$ just before a refreshment time. Thus the problem reduces to analyze the behavior of the model at the refreshment times. Moreover the transition phase is very short thus the study of the stationary behavior is pertinent.

The key idea of the study is to use the Markovian framework in order to rigorously establish limit theorems and analytical expressions in the stationary regime. The main result is that, as $m$ tends to infinity, the evolution of the model is characterized by a dynamical system which has a unique fixed point $\bar{w}$. The interpretation of $\bar{w}$ as a key quantity in a supermarket model with deterministic service times is discussed. Analytical expressions are given in [19] for $k = 1$ but are more complicated to obtain here for $k \geq 2$.

An objective would be to prove the convergence of the invariant measure of the Markov chain as $m$ tends to $+\infty$ to the Dirac measure $\delta_{\bar{w}}$ at the fixed point $\bar{w}$. In practice, such a result is completely crucial. If it is not true, if the sequence of invariant measures do not converge, the system oscillates with long periods of transition between different configurations (metastability phenomenon). So even if the algorithm performs well during a while, it can reach another state where it can give bad results. This question is partially addressed here. The convergence of the invariant measures is conjectured, due to simulations of the algorithm where such a phenomenon has not been observed. For such a result, a possible technique is the existence of a Lyapounov function which both proves the convergence of the dynamical system to its unique fixed point $\bar{w}$ and the convergence of the sequence of invariant measures to $\delta_{\bar{w}}$. Such a function is exhibited in [19] for $k = 1$ and $C = 2$.

A simulation of the limit distribution $\bar{w}$ is done for a uniform general mice size distribution which is non analytically tractable for $k \geq 2$. Is is compared to the experiment on a real trace. This one hour trace is commercial traffic provided by France Telecom. Results are very close. This trace is used as a validation test and no algorithm parameter need to be changed to fit to some characteristics of this particular trace. Experiments have three other goals. First to compare the original version with the min rule to the modified version of the algorithm introduced here. It appears that the latter version exhibits performances as good as the original one. Second, the time between two refreshments is plotted. This quantity is crucial for the trade-off between false positives and false negatives. It depends on the value of the threshold $r$. If $r$ is high, the counters are high due to collisions, involving many false positives. But refreshment times occur less often so less packets are lost, inducing less false negatives. The analysis shows such a behavior. Experiments validate the analysis. Third, the time to reach the stationary phase is also discussed.

The organization of the chapter is as follows : Section II presents the analytical results for the simple model defined to study the question of false positives. Section III is devoted to experiments. Section IV mainly gives a discussion of the way to choose the parameter $r$ in order to have an algorithm which performs well.

## 3.2 The Markovian urn and ball model

### 3.2.1 Description of the model

In this section, the question of false positives is addressed : The target quantity is the probability for a mouse to be detected by the algorithm as an elephant.

The problem is studied in a simple framework, where flows are reduced to mice of size one. Thus the model can be described as a urn and ball model because one size flows hashed in a filter with $m$ counters can be viewed as balls thrown into $m$ urns with capacity $C$ under the supermarket rule : For each ball, a subset of $k$ urns is chosen at random and the ball is put in the least loaded urn, ties being resolved uniformly. Balls overflowing the capacity $C$ are rejected. Moreover if, after putting the ball, the number of non empty urns exceeds $rm$, then one ball is removed from every non empty urn.

The probability of a flow to be detected as an elephant is reduced to the probability that, after the ball arrival, all the $k$ chosen urns have $C$ balls. It is bounded by the probability that, just before a refreshment time, after putting the last ball in its urn, all the $k$ urns chosen for that contain $C$ balls. The bound is more convenient to study. Let us focus on the embedded model just before the refreshment times.

### 3.2.2 A Markovian framework

For fixed $C$, let us consider the sequence $(W_n^m)_{n\in\mathbb{N}}$, where $W_n^m$ denotes the vector of the proportions of urns with $0,\dots,C$ balls just before the $n$th refreshment time. For $m \geq 1$, $(W_n^m)_{n\in\mathbb{N}}$ is an ergodic Markov chain on the finite state space

$$\mathcal{P}_m^{(r)} = \{w \in \left(\frac{\mathbb{N}}{m}\right)^{C+1}, \sum_{i=0}^{C} w(i) = 1,\ \sum_{i=1}^{C} w(i) = \frac{\lceil rm \rceil}{m}\},$$

(where $\lceil rm \rceil$ denotes the smallest integer larger than $rm$). Thus it has a unique invariant measure $\pi_m$.

The problem is that this quantity is combinatorically intractable. Even the transition probability $P_m$ of the Markov chain is awfully difficult to write. Nevertheless, one could expect an asymptotic of this quantity when $m$ gets large. In other words, the limit of the invariant measures $\pi_m$ when $m$ is large is investigated.

### 3.2.3 A dynamical system

The way which is used here to obtain limit theorems is very classical (see [30] for example). In fact, similar results for $k = 1$ can be found in Chabchoub et al [19]. The following results extend the case $k = 1$ to $k \geq 1$. Of course the motivation here is the case $k \geq 2$. The proofs must often be rewritten with new arguments and the sections which are still valid will be in general omitted.

Let us introduce some notations. Let

$$\mathcal{P} \stackrel{def}{=} \{w \in \mathbb{R}_+^{C+1}, \sum_{i=0}^{C} w(i) = 1\}$$

$$\text{and } \mathcal{P}^{(r)} \stackrel{def}{=} \{w \in \mathbb{R}_+^{C+1}, \sum_{i=0}^{C} w(i) = 1 \text{ and } \sum_{i=1}^{C} w(i) = r\}$$

be the state spaces. Let $s$ be the shift defined as

$$s : w \mapsto (w(0)+w(1), w(2), \ldots, w(C), 0) \text{ on } \mathcal{P}$$

and

$$\lambda : \mathcal{P}^{(r)} \to \mathbb{R}^+,\ w \mapsto \int_{r-w(1)}^{r} \frac{du}{1-u^k}. \tag{3.1}$$

For the vector of proportions $v \in \mathcal{P}$, it is more convenient to deal with the vector of the tail proportions $u$ defined by $u_j = \sum_{i \geq j} v_i$. $G$ is then defined on $\mathcal{P}^{(r)}$ by

$$G(w) = v(\lambda(w)) \tag{3.2}$$

where $(v(t))$ is associated to $(u(t))$ the unique solution of

$$\frac{du_j}{dt} = u_{j-1}(t)^k - u_j(t)^k,\ j \in \{1, \ldots C\},\ u_0 = 1 \tag{3.3}$$

with initial condition $u(0)$ corresponding to $v(0) = s(w)$.

The following result is that, as $m \to +\infty$, the Markov chain converges in distribution to a deterministic dynamical system which will be explicited.

**Proposition 5.** *If $W_0^m$ converges in distribution to $w \in \mathcal{P}_m^{(r)}$ then $(W_n^m)_{n\in\mathbb{N}}$ converges in distribution to the dynamical system $(w_n)_{n\in\mathbb{N}}$ given by the recursion*

$$w_{n+1} = G(w_n),\ n \in \mathbb{N}$$

*where $G$ is defined by equation* (3.2)*.*

Notice that $G$ maps, by definition of $\lambda$, $\mathcal{P}^{(r)}$ to $\mathcal{P}^{(r)}$.

*Démonstration.* The result is a consequence of the convergence of the transition $P_m$ of the Markov chain $(W_n^m)_{n\in\mathbb{N}}$ as $m$ tends to $+\infty$ to $P$ given by

$$P(w, .) = \delta_{G(w)}.$$

It means that, starting from $w$ just before a refreshment time, at the next refreshment time, the vector of the proportions of urns tend to $G(w)$ when $m$ tends to $+\infty$. The uniform convergence stated by the following lemma provides the convenient way to prove Proposition 1.

**Lemme 2.** *For $\varepsilon > 0$,*

$$\sup_{w \in \mathcal{P}_m^{(r)}} P_m(w, \{w' \in \mathcal{P}_m^{(r)} : ||w' - G(w)|| > \varepsilon\}) \underset{m\to+\infty}{\to} 0.$$

*Démonstration.* The idea of the proof is that, starting from $w$ (with $\lceil rm \rceil$ non empty urns), after refreshment, the vector of the proportions becomes $s(w)$ defined by

$$s(w) = (w(0)+w(1), w(2), \ldots, w(C), 0)$$

where the proportion of non empty urns is $r - w(1)$. Then a number $\tau_1^m$ of balls are thrown in order to reach a state $w'$ with again $\lceil rm \rceil$ non empty urns. It has to be proved that $w'$ is close to $G(w)$. There are three steps :

1) It can be proved that this number $\tau_1^m$ is deterministic at first order when $m$ is large, equivalent to $\lambda(w)m$, where $\lambda(w)$ is defined by equation (3.1). More precisely,

$$\sup_{w\in\mathcal{P}_m^{(r)}} \mathbb{P}_w\left(\left|\frac{\tau_1^m}{m}-\lambda(w)\right|>\varepsilon\right)\underset{m\to\infty}{\to} 0. \tag{3.4}$$

To see it, use that, starting from $w$, $\tau_1^m$ has an analytical expression as a sum of different numbers $Y_l$ of balls necessary to hit the $(l+1)$th non empty urn. Indeed,

$$\tau_1^m = \sum_{l=\lceil rm\rceil - w_1 m}^{\lceil rm\rceil -1} Y_l, \tag{3.5}$$

where the $Y_l$s for $l\in\mathbb{N}$ are independent random variables with geometrical distributions on $\mathbb{N}^*$ with respective parameters

$$a_l = \prod_{j=0}^{l-1}\frac{l-j}{m-j},$$

i.e. $\mathbb{P}(Y_l=n)=(l/m)^{n-1}(1-l/m)$, $n\geq 1$.

As $\mathbb{E}(Y_l)=1/(1-a_l)$, computing the mean and comparing this sum with integrals leads to

$$\sup_{w\in\mathcal{P}_m^{(r)}} \mathbb{E}_w\left(\frac{\tau_1^m}{m}\right)\underset{m\to\infty}{\to}\lambda(w). \tag{3.6}$$

At the same time, as $\mathrm{Var}(Y_l)=a_l/(1-a_l)^2$,

$$\sup_{w\in\mathcal{P}_m^{(r)}} \frac{\mathrm{Var}_w(\tau_1^m)}{m}\underset{m\to\infty}{\to}\int_{r-w(1)}^{r}\frac{dt}{(1-t^k)^2}. \tag{3.7}$$

By Bienaymé-Chebychev's inequality, using equations (3.6) and (3.7), equation (3.4) is proved.

2) From the previous fact, there is a natural coupling throwing $\tau_1^m m$ balls or $\lambda(w)m$ balls where the vectors of proportions $W_1^m$ and say $\tilde{W}_1^m$ are close to each other. Then there is also a coupling throwing $\lambda(w)m$ and a Poisson random variable with parameter $\lambda(w)m$, for which, by Chernoff's inequality, the vector of proportions $\tilde{W}_1^m$ and say $\hat{W}_1^m$ are close.

3) The vector of proportions $\hat{W}_1^m$ obtained by coupling is the vector of proportions at time $\lambda(w)$ in a queueing supermarket model without departures. The model consists of $m$ queues with capacity $C$ where customers arrive according to a Poisson process with rate $m$. At each arrival, a subset of $k$ queues is chosen and the customer joins the shortest one, ties being solved at random. Let $W^m(t)$ be the vector of the proportions of $m$ queues with $0,1,\ldots,C$ customers at time $t$. It is more convenient to deal with the tail proportions defined as

$$U_j^m(t)=\sum_{i\geq j} W_i^m(t).$$

Given $W^m(0)=s(w)$, we have that

$$\hat{W}_1^m = W^m(\lambda(w)).$$

By the convergence of the Markov process $(U^m(t))$ to the fluid limit (see Vvedenskaya et al. [72] for example), it holds that $\hat{W}_1^m$ converges in distribution to $v(\lambda(w))$ where $v$, defined by (3.3), is associated to $u$ the fluid limit of $(U^m(t))$, the unique solution of the differential system

$$\frac{du_j}{dt}=u_{j-1}(t)^k-u_j(t)^k\ (1\leq j\leq C),\ u_0=1, \tag{3.8}$$

with initial condition $u(0)$ corresponding to $v(0) = s(w)$. Moreover using the continuity of a solution of a differential equation with respect to the initial condition, for each $\varepsilon$, $t > 0$, $0 \leq j \leq C$,

$$\sup_{w \in \mathcal{P}_m^{(r)}} \mathbb{P}_w(|U_j^m(\lambda(w)) - u_j(\lambda(w))| > \varepsilon) \underset{m \to \infty}{\to} 0$$

which straightforwardly leads to

$$\sup_{w \in \mathcal{P}_m^{(r)}} \mathbb{P}_w(\| \hat{W}_1^m - G(w) \| > \varepsilon) \underset{m \to \infty}{\to} 0$$

where

$$G(w) = v(\lambda(w)).$$

It ends the proof of the lemma. □

The argument to obtain Proposition 1 from Lemma 2 is standard and detailed in [19, Proposition 1]. It is omitted here. □

### 3.2.4 Fixed point of the dynamical system

The function

$$\begin{aligned} \mathcal{P}^{(r)} &\longrightarrow \mathcal{P}^{(r)} \\ w &\longmapsto G(w) \end{aligned}$$

being continuous on the convex compact set $\mathcal{P}^{(r)}$, by Brouwer's theorem, it has a fixed point.

It remains to prove the uniqueness of the fixed point. Recall that, for $k = 1$, the proof is based on the interpretation of the fixed point equation

$$G(w) = w$$

as the equation

$$w = \mu_{\lambda(w)} \tag{3.9}$$

where $\mu_\lambda$ is the invariant measure of some ergodic Markov chain. This Markov chain is the queue length at the service completion times of a $M/G/1/C$ queue with deterministic service times equal to 1 with arrival rate $\lambda$. The proof of the uniqueness of the solution of equation (3.9) is then based on the coupling argument that, if $\lambda \leq \lambda'$ then $\mu_\lambda$ is stocastically dominated by $\mu_{\lambda'}$ (see [19] for details).

Let us try to extend the argument for the case $k \geq 2$. For that, let us consider the following system. Balls arrive according to a Poisson process with rate $\lambda m$. They are thrown into $m$ urns with capacity $C$ as follows. Each ball joins the least loaded urn among a subset of $k$ urns, chosen at random. The ties are resolved uniformly. At each unit time, one ball is removed from each non-empty urn. It can be proved as in Proposition 1 that the vector of the proportions of urns with $j$ balls just before time $n$ converges, when $m$ is large, to a dynamical system

$$w_{n+1} = H(w_n)$$

where, for $v$ defined defined by (3.3) with initial condition $v(0) = s(w)$,

$$H(w) = v(\lambda).$$

But the argument fails for $k \geq 2$. Assume that $H(w) = w$ is the invariant measure equation of some ergodic Markov chain, i.e., of type $wP = w$ for some transition matrix $P$. As $H(w) = v(\lambda)$,

$$v(\lambda) = wP.$$

From equation (3.8), $u_1(t)$ can be computed explicitly. It gives that

$$v_0(t) = 1 - F^{-1}(t + F(r - w(1))$$

where $F(x) = \int_0^x \frac{dt}{1-t^k}$ is a bijection from $[0,1[$ to its image. It means that, for $k \geq 2$, $v_0(\lambda)$ can not be written as $\sum_{j=0}^{C} P_{0,j} w(j)$.

Thus another way to prove it can be found and, at this point, the uniqueness of the fixed point is conjectured.

### 3.2.5 Identification of the fixed point

If the capacity is infinite, then the parameter $\lambda(\bar{w})$ is equal to $r$. In this case, it is simple to have the explicit expression of $\bar{w}_1$, which is a good approximation for the case $C = 20/k$, for $k$ small enough. It is the purpose of this section.

Assume that $C = +\infty$. By definition,

$$\lambda(\bar{w}) = \int_{r-\bar{w}(1)}^{r} \frac{dt}{1-t^k}$$

and $F(x) = \int_0^x \frac{dt}{1-t^k}$ defines a bijection from $[0,1[$ to its image. Using that $\lambda(\bar{w}) = r$ for $C = +\infty$, it can be rewritten

$$r = F(r) - F(r - \bar{w}(1)),$$

or

$$\bar{w}(1) = r - F^{-1}(F(r) - r). \tag{3.10}$$

Notice that for $k \in \mathbb{N}$, $F$ has an explicit expression. For $k = 1$, $F(x) = -\log(1-x)$ which leads (see [19]) to

$$\bar{w}(1) = (1-r)(e^r - 1).$$

Moreover, for $k = 2$, $F(x) = \text{argth}x$ and thus

$$\bar{w}(1) = r - \frac{r - \text{thr}}{1 - r\text{thr}}. \tag{3.11}$$

For $k \geq 4$, even if $F$ has a simple close form (for instance, for $k = 4$,

$$F(x) = (\text{argth}x + \arctan x)/2)$$

$F^{-1}$ and $\bar{w}_1$ have to be numerically computed. In Figure 13, $\bar{w}(1)$ is plotted for $k = 1$, 2 and 4.

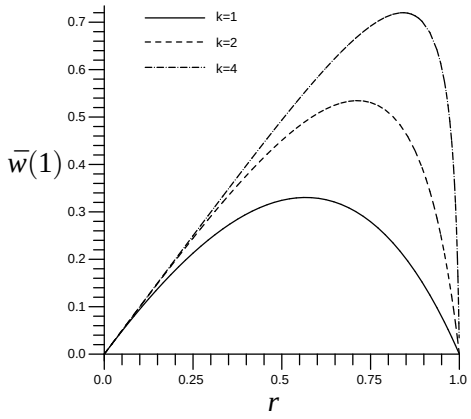



FIG. 13 – Limit proportion $\bar{w}(1)$ of counters with value 1 for $k = 1, 2, 4$.

### 3.2.6 Convergence of invariant measures

A first result is obtained. It is proved in [19, Proposition 2] and recalled here omitting the proof.

**Proposition 6.** *Let, for $m \geq 1$, $\pi_m$ be the stationary distribution of $(W_n^m)_{n\in\mathbb{N}}$. Define $P$ as the transition on $\mathcal{P}^{(r)}$ given by $P(w,.) = \delta_{G(w)}$. Any limiting point $\pi$ of $(\pi_m)_{m\in\mathbb{N}}$ is a probability measure on $\mathcal{P}^{(r)}$ which is invariant for $P$ i.e. that satisfies $G(\pi) = \pi$.*

As noticed in the introduction, the limiting point of $\pi_m$ is not necessarily unique, because there is not a unique measure $\pi$ such that $G(\pi) = \pi$. There is one because $G$ has a unique fixed point thus $G(\delta_{\bar{w}}) = \delta_{\bar{w}}$. But imagine that $G$ has cycles, i.e. that there exist $n \geq 2$ and $w_1, \ldots, w_n$ in $\mathcal{P}^{(r)}$ such that

$$G(w_j) = w_{j+1}\ (1 \leq j < n),\ G(w_n) = w_1$$

then $\pi = 1/n \sum_{j=1}^{n} \delta_{w_j}$ is invariant under $G$. It gives two different limiting points for $\pi_m$. A way to prove the convergence of $(\pi_m)$ to $\delta_{\bar{w}}$ is to find a Lyapounov function for $G$ (see [19, Theorem 1] for details). Such a Lyapounov function is exhibited in [19] for $k = 1$ and $C = 2$. It is not investigated here.

### 3.2.7 General mice size distribution

The subsection deals with the extension of the previous results to a model with general size distribution. An approximated model is taken. Indeed, as mice size are short (with mean close to some units, in real traffic traces, close to 4), an approximated model is to consider that the packets of the mice are thrown without interleaving with packets of other mice in the target counters. It means that the packets of the different mice arrive consecutively in the filter.

The model chosen is thus an urn and ball model where balls are thrown by batches. The balls in a batch are thrown successively, each ball into the least loaded urn among $k$ chosen at random in the $m$ urns. Notice that the set of $k$ urns is the same for every ball of a given batch, but the balls tend to fill these $k$ urns alternatively. The $i$th batch is composed with $S_i$ balls, where the $S_i$s are independent random variables with distribution denoted by $p$. The model generalizes the previous one obtained for mice of size one (p(1)=1).

This model, which is simple to analyze in the case $k = 1$ (see [19, Subsection 2.1]) as an extension of the model with mice of size one, becomes much more difficult in the case $k \geq 2$.

To our knowledge, the pending queueing system with batch arrivals and customers joining the shortest queue has never been studied.

## 3.3 Experiments

In this section, the proposed algorithm is tested against an ADSL commercial traffic trace from France Telecom IP backbone network. This traffic trace has been captured on a Gigabit Ethernet link in October 2003 between 9 :00 pm and 10 :00 pm. This period corresponds to a peak activity by ADSL customers (the link load was equal to 43.5%), its duration is 1 hour and contains more than 10 millions of TCP flows. Some trace characteristics are given in Table I.

| Traces | Nb. IP packets | Nb. TCP Flows | Duration |
|---|---|---|---|
| FT trace | 135 844 423 | 10 474 665 | 1 hour |

TAB. 3.1 – Characteristics of the traffic trace considered in experiments

In our experiments, the filter consists of $m = 2^{20}$ counters associated to two independent hashing functions ($k = 2$). Elephants are here defined as flows with at least 20 packets ($K = 20$).

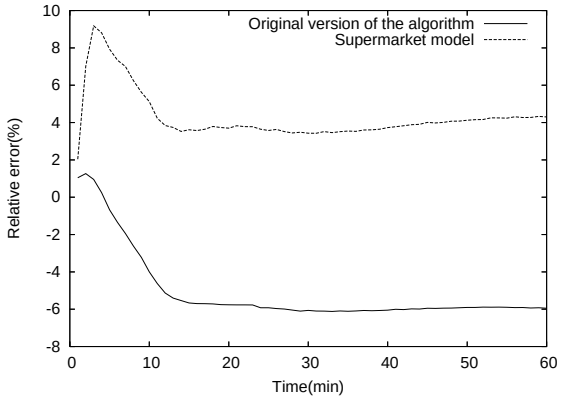



FIG. 14 – Impact of the Supermarket model on the estimation of elephants number , $r = 50\%$, France Telecom trace

The relative error on the estimated number of elephants, defined as the difference between the number of detected elephants and the exact number of elephants divided by the exact number of elephants, is plotted in Figure 14. Two different versions of the algorithm are considered : The original algorithm developed in [6] and presented in the introduction and the modified algorithm introduced here. Recall that these two algorithms use the min-rule (incrementing only smallest counters), but in a different way : In case of equality, only one counter is incremented at random in the supermarket scheme, whereas the two counters are incremented in the original version of the algorithm. The proposed alternative algorithm implements the min rule in a more standard way. It allows to exploit well-known analytical results on the supermarket model, which is very studied in the literature. In its original version, the fact that each lowest counter is incremented makes the model more difficult to analyze. Experiments show that both methods give a good estimation of number of elephants, for the whole duration of the trace. Moreover, the supermarket version has less missed elephants, which can be better for some applications.

One can notice that for the modified algorithm, the generated error is higher at the beginning (when time is less than 10 minutes). This can be explained by the fact that starting from an empty filter (with all counters at 0), it takes a long time to reach the filling up threshold (50%) to perform the first refreshment. As a consequence, counters are high compared to their values in the stationary phase. Indeed it has been checked experimentally that, just before the first refreshment, the proportion of counters at $C$ is up to 10 times higher than in the stationary phase.

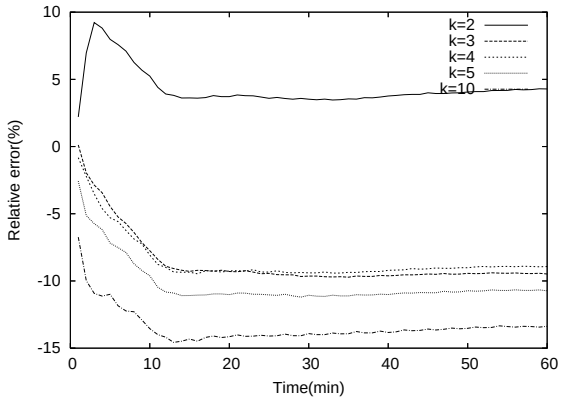



FIG. 15 – Impact of $k$ on the estimation of elephants number, $r = 50\%$, France Telecom trace

Figure 15 gives the relative error on the estimated number of elephants for different values of $k$. The first observation is that, for larger values of $k$, the algorithm remains very robust even if the following occurs : A flow is declared elephant if its $k$ associated counters reach $C = 20/k$. In a $k = 10$ version, $C$ is only 2. Nevertheless, in stationary regime, the algorithm performs better for $k = 2$. Indeed, it seems on Figure 15 that more elephants are missed for larger $k$, generating a higher negative error on the estimation of number of elephants. The explanation could be the following : Elephant packets are alternatively incrementing $k$ counters, so when a refreshment occurs, up to $k$ packets of the elephant are lost at the same time, according to the number of packets already transmitted. It implies that, if $k$ is large, then elephants with size close to 20 could be more easily lost. In the following, $k$ will remain equal to 2.

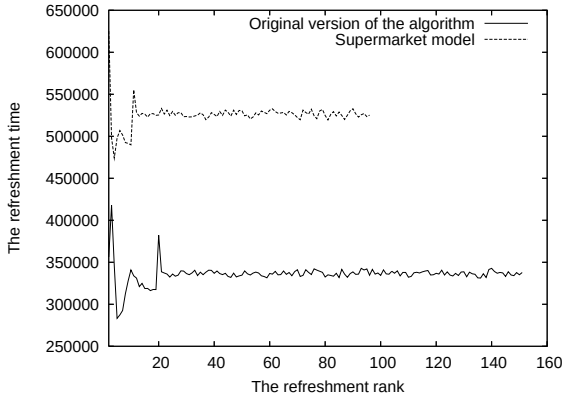



FIG. 16 – Duration of the transition and the stationary phase, $r = 50\%$, France Telecom trace

Figure 16 presents *the inter-refreshment time* (duration between two successive refreshments in terms of number of arriving packets) for the whole traffic trace. It can be noticed that the

stationary phase is reached at the $K$th refreshment time. So the transition phase seems to be rather short, according to experiments. The stationary inter-refreshment time using the algorithm based on the supermarket model is higher than the one obtained with the original version of the algorithm. This can be explained by the fact that, with the supermarket scheme, every arriving packet increments exactly one counter, whereas in the original version, if the two selected counters are equal, they are both incremented by one. In particular, when they are both null, they will be both impacted. As a consequence, the proportion of non null counters grows faster and the filling up threshold $r$ is reached more quickly.

Figure 16 gives an explanation to the behavior of the algorithms plotted in Figure 14. In the original algorithm, the inter-refreshment time is lower thus more elephants are missed. The error is thus negative. In the supermarket version, the error is positive due to false positives.

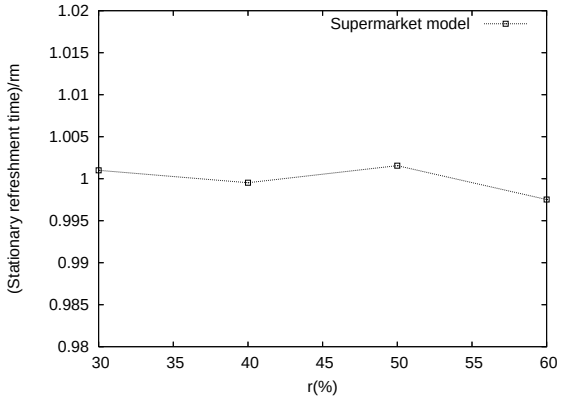



FIG. 17 – Comparison between $rm$ and the stationary inter-refreshment time $\tau_\infty^m$, France Telecom trace

In Figure 17, the impact of $r$ on the stationary inter-refreshment time $\tau_\infty^m$ is investigated. More precisely $\tau_\infty^m/rm$ is plotted for various values of $r$. According to experiments, $\tau_\infty^m$ is very close to $rm$. In fact the refreshment can be seen as removing $rm$ from the sum $S$ of all counters (decreasing by one all non null counters which are exactly $rm$ as the refreshment is performed as soon as the filling up threshold $r$ is reached). In the stationary phase, when m is large, the proportion of counters at $i$, for $i \in \{0,\ldots,C\}$, is $w_i$ at the first order. And just before the next refreshment, $rm$ packets must be inserted into the filter, to let $S$ have its former value. Packets belonging to elephants which have been detected are not taken into account. Those packets are very numerous and they are not inserted into the filter to avoid polluting it. The conclusion is that $\tau_\infty^m/m$ is close to $r$ even for $C = 10$.

## 3.4 Discussion

The performance of the algorithm clearly depends on the filling up $r$. To have a good estimation of the number of elephants, $r$ must be around 50%. When $r$ has higher values, elephants number will be largely overestimated due to false positives. The key quantity is $\bar{w}(i)$, the stationary proportion of counters at $i$ when $m$ gets large. An explicit expression for $\bar{w}$ is not available even if a numerical value could be computed. Nevertheless, less ambitiously, one can maybe simply find the critical value of $r$ for which $\bar{w}(C)$ gets non negligible. At least, the impact of $r$ on $\bar{w}$ is shown here by simulation.

Figure 18 ($w$ is written for $\bar{w}$) is not based on a real traffic trace but on simulation. The

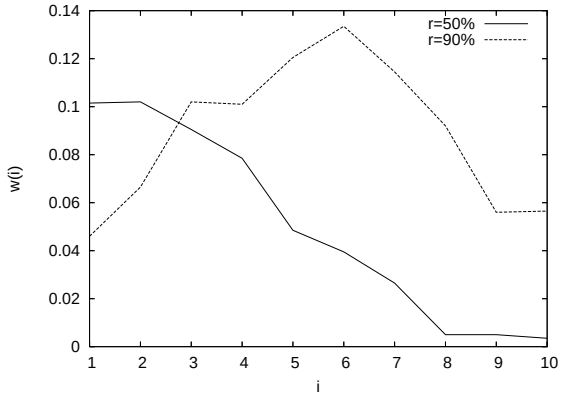



FIG. 18 – Impact of $r$ on the limit stationary distribution $\bar{w}$, simulation using only mice with uniform size distribution of mean 4, $m = 2000$

objective here is to evaluate the limit stationary distribution $\bar{w}$ if we consider a traffic composed only of mice. The mice mean size is taken equal to four to be close to the real traffic (This value is deduced from the real traffic trace). Under these conditions, we obtain a decreasing limit stationary distribution of $\bar{w}$, when $r$ equals 50%. For a filling up threshold of 90%, counters are very likely to be higher. We can notice that the main part of counters values is around six and there are many counters at $C$. This explains the fact that, with a filling up threshold around 50%, the algorithm performs better.

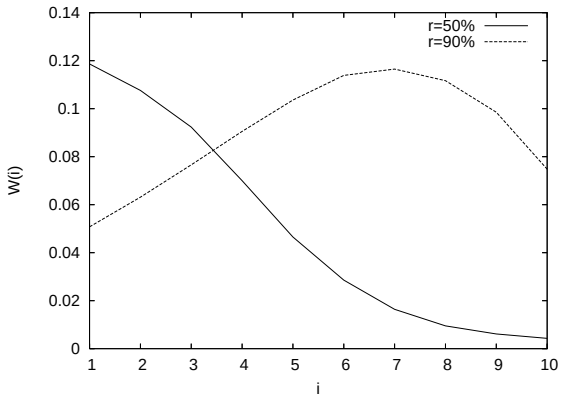



FIG. 19 – Impact of $r$ on the limit stationary distribution $\bar{w}$, France Telecom trace

The same experiments are now performed on the real traffic trace. Notice that results plotted on figure 19 are very close to those obtained by simulation on figure 18. One can deduce that the elephants present in the real traffic trace do not impact too much the limit stationary distribution $\bar{w}$. As a consequence, to analyze the performance of the algorithm, in particular its generated false positives, flows can be reduced to mice. This is in agreement with the model considered in part II.

## 3.5 Conclusion

In this chapter, a new algorithm catching on-line elephants in the Internet is analyzed. This algorithm is based on Bloom filters with a refreshment mechanism that depends on the current

traffic intensity. It also uses a conservative way to update counters, called the min-rule. This latter is exactly to increment the lowest counter among a set of $k$ chosen at random, ties being solved at random, as in the supermarket model which provides a much lower tail distribution for the counter values. For a model involving just mice, limit theorems investigate the existence of a deterministic limit for the empirical distribution of counters values, when the filter size gets large. This limit can be exploited to adjust the parameters for the algorithm to perform well. The accuracy of the algorithm and some theoretical results are tested against a traffic trace from France Telecom and by simulations.

## Deuxième partie

# Internet traffic modeling and inference of flows characteristics via sampling

# Introduction

Packet sampling is an efficient method of reducing the amount of data to retrieve and to analyze in order to study the characteristics of IP traffic (cf. the drafts of IPFIX [50] and PSAMP [51] working groups at the IETF). The simplest approach to packet sampling is certainly the so-called 1-out-of-$k$ sampling technique, which consists of capturing and analyzing one packet every other $k$ packets with $k = 100, 500 or 1000$ in practice. This method will be referred to in the following as deterministic sampling, which has been implemented, for instance, in CISCO routers (NetFlow facility [24]) and is widely used in today's operational networks, even if it suffers from several shortcomings identified in [33]. Deterministic sampling is not easy to model as it can introduce some bias in sampled data. That is why theoretical studies rely on an other sampling method called probabilistic or random sampling. It consists in selecting with a probability $p$.

Packet sampling reduces certainly the amount of data but causes also a great loss of information about flows. In particular, the majority of small flows are not sampled. Thus recovering original flow statistics from sampled data is a difficult task (see [29] for instance). Different solutions have been introduced to overcome these limitations (e.g., the "sample and hold" technique by Estan and Varghese [34], adaptive sampling [23, 33], etc.).

The main objective of this part is to propose some algorithms inferring flow characteristics from sampled traffic. Note that a flow is here defined as the set of those packets sharing common characteristics (same source and destination IP addresses and port numbers together with the same protocol type).

This part is organized as follows : In chapter 4, we prove, under some assumptions, the equivalence of deterministic and probabilistic sampling methods in term of composition of sampled traffic in flows. We also show that the tail distribution of flow size can be inferred from the distribution of the number of its sampled packets. In chapters 5 and 6, we propose a new method inferring the total number of large flows and their size distribution, from sampled traffic. These results are validated on several traces from different types of IP networks.

# 4

# Deterministic versus probabilistic packet sampling

**Contents**



Under the assumption that packets are sufficiently interleaved and the sampling rate is small, we show in this chapter that those characteristics of flows like the number of packets, volume, etc. obtained through deterministic 1-out-of-$k$ packet sampling is equivalent to random packet sampling with rate $p = 1/k$. In particular, it is shown that under mild assumptions, the tail distribution of the total number of packets in a given flow can be estimated from the distribution of the number of sampled packets. Explicit theoretical bounds are then derived by using technical tools relying on bounds of Poisson approximation (Le Cam's Inequality) and refinements of the central limit theorem (Berry-Essen bounds). Experimental results from an ADSL trace show a good agreement with the theoretical results established in this chapter.

## 4.1 Introduction

Because deterministic sampling may introduce some synchronization and then some bias in sampled data, which bias is not easy to determine because it depends upon the realization of flows (i.e., the relative position of packets between each other), several studies and IETF drafts [75] recommend probabilistic sampling. In its basic version, random sampling consists of picking up a packet, independently from other packets, with a given probability $p$. The major advantage is that random sampling provides isolation between flows : the selection of a packet does not depend upon the relative position of flows between each other.

In this chapter, it is shown that if packets are sufficiently interleaved (which is definitely the case on a transmission link of a backbone network), then 1-out-of-$k$ deterministic sampling is equivalent to random sampling with $p = 1/k$. More precisely, an explicit estimation of the distance (for the total variation norm) between the distributions of the numbers of packets in a flow sampled with the two sampling techniques is obtained.

On the basis of this result, bounds on the difference between the distributions of the original flow size and of the sampled flow size rescaled by the sampling factor are established. If the estimation of the size of a flow with the number of sampled packets scaled by the sampling factor is natural and frequently used in the literature, it is not always accurate and can be wrong sometimes. A bound to estimate the accuracy of this estimation is therefore important in practice. Provided that the flow size is sufficiently heavy tailed, it can be shown that the original size of a flow can be indeed estimated from the number of sampled packets.

The different theoretical results obtained in this chapter are illustrated on a traffic trace from the France Telecom backbone network carrying ADSL traffic. For this purpose, we introduce a flow decomposition technique based on an ad-hoc mouse/elephant dichotomy. The theoretical results are applied to elephants. Mice appear as background noise in sampled data and their flow size distribution is of less interest, since their volume represents only a small fraction of global traffic. Experimental data show good agreement with theoretical results.

The chapter is organized as follows : In Section 4.2, we describe the traffic analysis methodology. The comparison between deterministic sampling and random sampling is discussed in Section 4.3 and results on random sampling are then established. These results are compared in Section 4.4 against experimental results. Concluding remarks are presented in Section 4.5.

## 4.2 Traffic analysis methodology

Let us consider a high speed transmission link carrying Internet traffic and let us divide time into slots of length $\Delta$. The constant $\Delta$ may range from a few seconds to several tens of minutes (say, from one to two hours).

In this chapter, we are interested in the characteristics of TCP traffic since it still represents today 95 % of the total amount of traffic in IP networks, even though the proportion of UDP traffic is growing with the development of streaming applications (VoIP, video, peer-to-peer streaming, etc.). In the literature on Internet traffic characterization, it is well known that all flows are not all equivalent : there are flows with many packets, possibly transmitted in bursts, and small flows comprising only a few packets. Many small flows are composed of single SYN segments corresponding to unsuccessful TCP connection establishments attempts.

The elephant/mouse dichotomy used in this chapter corresponds more or less to the definition introduced by Paxson and Floyd [67], even if clear definitions for mice and elephants do not exist

(see the discussion in [65]). To be more specific, we shall use the following definitions :

**Definition 2** (Mouse/Elephant)**.** *A mouse is a flow with less than $b$ packets in a time window of length $\Delta$. An elephant is a flow with at least $b$ packets in a time window of length $\Delta$.*

We do not claim that the above definitions should be *the* definitions for mice and elephants ; they are introduced for convenience to split the flow population into two distinct sets. In particular, they depend upon the length $\Delta$ of the measurement window and the threshold $b$. In previous studies (see [8] for instance), a threshold $b = 20$ packets yields a neat delineation between mice and elephants when dealing with ADSL traffic even for large observation windows. This is the value which is chosen in the whole chapter.

To illustrate the above definition, we consider a traffic trace from the France Telecom IP backbone network carrying ADSL traffic. This traffic trace has been captured on a Gigabit Ethernet link in October 2003 between 9 :00 pm and 11 :00 pm (this time period corresponding to the peak activity by ADSL customers) ; the link load was equal to 43.5%. The complementary cumulative distribution function (ccdf) of the number $N_{\text{mice}}$ of packets in mice is displayed in Figures 4.20(a) and 4.20(b) for $\Delta = 5$ seconds and $\Delta = 3200$ seconds, respectively. We see that for $\Delta = 5$ seconds, the distribution of the random variable $N_{\text{mice}}$ can reasonably be approximated by a geometric distribution (i.e., $\mathbb{P}(N_{\text{mice}} > n) \approx r_1^n$). By using a standard Maximum Likelihood Estimation (MLE) procedure, we find $r_1 = 0.75$. For $\Delta = 3200$ seconds, only the tail of the distribution can be approximated by a geometric distribution ; experimental results give $\mathbb{P}(N_{\text{mice}} > n) \approx c_2 r_2^n$ for large $n$, with $c_2 = .1$ and $r_2 = .6$.

The distribution of the number $N_{\text{eleph}}$ of packets in elephants is displayed in Figure 4.20(c) and 4.20(d) for $\Delta = 5$ and $\Delta = 3200$ seconds, respectively. Now, we see that elephants clearly exhibit a behavior, which is significantly different from that of mice. The random variable $N_{\text{eleph}}$ has a slowly decreasing distribution, which can reasonably be approximated by a Pareto distribution, at least for moderate values of $N_{\text{eleph}}$ for $\Delta = 5$ seconds.

We specifically have $\mathbb{P}(N_{\text{eleph}} > n) \approx (b/n)^a$ for $n \geq b = 20$. For $\Delta = 5$ seconds, we find by means of a standard MLE procedure $a = 1.95$. When $\Delta = 3200$ seconds, the distribution of $N_{\text{eleph}}$ is more complicated and can be approximated by two Pareto distributions, namely $\mathbb{P}(N_{\text{eleph}} > n) \approx (20/n)^{a_2}$ for $20 \leq n \leq 2000$ with $a_2 = .55$, and $\mathbb{P}(N_{\text{eleph}} > n) \approx (600/n)^{a_2'}$ for $n \geq 2000$ with $a_2' = 1.8$.

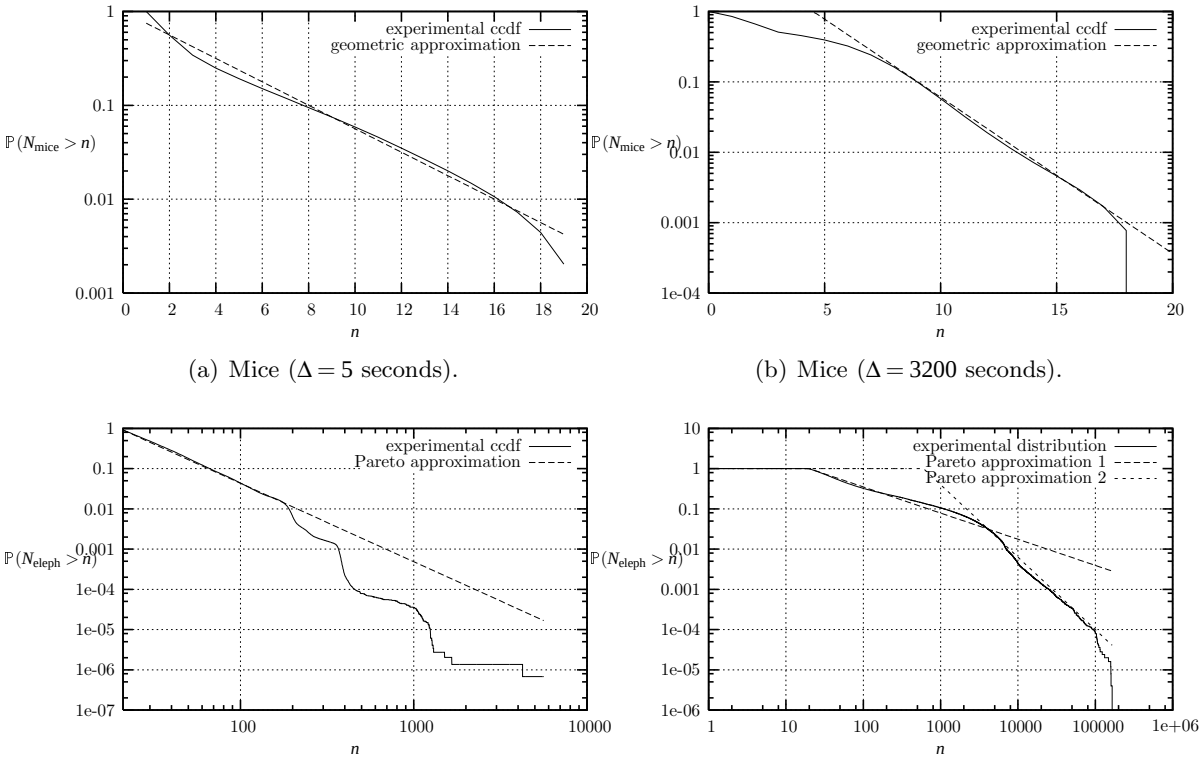
**Remark**. It turns out that taking only a limited time window for the statistics of the duration of a flow gives a much more robust statistical description of the traffic. Additional works has to be done to recover the full information on the duration of the flows.

In this chapter, we are interested in comparing the random variables describing the number of packets in a sampled flow, when deterministic or random sampling is performed.

## 4.3 Properties of random and deterministic sampling

### 4.3.1 Deterministic sampling

In the case of deterministic sampling, one packet is selected every other $1/p$ (integer) packets, where $p$ is the sampling rate. If packets of flows are back to back, then there is little chance of seeing flows more than once if their number of packets is not significantly larger than the sampling coefficient $1/p$. Fortunately, on a high speed backbone link, the number of simultaneous flows is very large and packets of the different competing flows are highly interleaved. Hence,




(a) Mice ($\Delta = 5$ seconds).

(b) Mice ($\Delta = 3200$ seconds).

(c) Elephants ($\Delta = 5$ seconds).

(d) Elephants ($\Delta = 3200$ seconds).

FIG. 20 – Ccdf of the number of packets in mice and elephants for $\Delta$=5 seconds and $\Delta = 3200$ seconds ($b = 20$ packets).

consecutive packets of a given flow are separated by many packets of other flows. This introduces some randomness in the selection of packets of a given flow.

More precisely, we consider a model where all flows are permanent in a time window of length $\Delta$. This means that, during all this time window, each flow $f$ is active and has a constant rate $v_f/\Delta$, where $v_f$ is the volume of the flow. Under this assumption, deterministic sampling consists of drawing $\lfloor pM(\Delta) \rfloor$ packets out of the total number $M(\Delta)$ of packets in the time window. If packets are sufficiently interleaved, a sampled packet belongs to a given flow $f$ with probability $v_f/M(\Delta)$. Hence, the number of sampled packets from flow $f$ is $\hat{v}_f = B_1^f + B_2^f + \cdots + B_{pM(\Delta)}^f$, where the quantities $B_j^f$ are independent Bernoulli random variables equal to one if the $j$th sampled packet is from flow $f$. Note that if $f$ and $g$ are distinct flows, then the variables $B_j^f$ and $B_j^g$ are *not* independent.
This model can be also seen as an urn and ball scheme with replacement : An urn contains a random number of balls with different colors. We draw a small fraction $p$ of the total number of balls. A ball which has been drawn is replaced into the urn.

The assumption of permanent flows is reasonable, when the observation window length $\Delta$ is small. When $\Delta$ is large, however, flows may be bursty and alternate between on and off periods. This phenomenon has been observed in particular when analyzing elephants in ADSL traffic [8]. Heuristically, at the first order, everything occurs as if the flows are permanent. It has been investigated both theoretically and by simulation. This is the object of Subsection 4.3.3. This assumption allows us, for the moment, to avoid such a discussion.

### 4.3.2 Probabilistic sampling

It is assumed in this section that random sampling is performed : each packet of a given flow $f$ with $v_f$ packets is taken with a probability $p$ and the number of packets in the sampled flow is exactly given by $\tilde{v}_f = \widetilde{B}_1^f + \widetilde{B}_2^f + \cdots + \widetilde{B}_{v_f}^f$, where the random variables $\widetilde{B}_i^f$ are iid with Bernoulli distribution with mean $p$. The key property of this sampling mode is that it provides isolation between flows. Mathematically, it amounts to the fact that the Bernoulli variables $\widetilde{B}_i^f$ and $\widetilde{B}_i^g$ are independent for distinct flows $f$ and $g$.

The comparison between the two sampling methods is done through the estimation of the *total variation distance* between the distributions of $\hat{v}_f$ and $\tilde{v}_f$,

$$\left\| \mathbb{P}(\hat{v}_f \in \cdot) - \mathbb{P}(\tilde{v}_f \in \cdot) \right\|_{tv} \stackrel{\text{def.}}{=} \sup_{A \subset \mathbb{N}} \left| \mathbb{P}(\hat{v}_f \in A) - \mathbb{P}(\tilde{v}_f \in A) \right|.$$

**Proposition 7** (Probabilistic vs. Deterministic Sampling)**.** *Under the above assumptions, for a flow $f$ with $v_f$ packets with $\mathbb{E}(v_f^2) < +\infty$, the relation*

$$\|\mathbb{P}(\hat{v}_f \in \cdot) - \mathbb{P}(\tilde{v}_f \in \cdot)\|_{tv} \leq p \frac{\mathbb{E}(v_f^2)}{M(\Delta)} + p^2 \mathbb{E}(v_f) \tag{4.1}$$

*holds. Moreover, as $M(\Delta)$ goes to infinity, the number of sampled packets $\hat{v}_f$ converges in distribution to $\mathbb{Q}$ defined by*

$$\mathbb{Q}(k) = \frac{p^k}{k!} \mathbb{E}\left( v_f^k e^{-p v_f} \right).$$

*Démonstration.* The proof relies on Le Cam's inequality conditionally on the value of $v_f$, see Chapter 1 of Barbour [11]. If $\mathsf{Pois}(\lambda)$ denotes the Poisson distribution with parameter $\lambda$, then

$$\|\mathbb{P}(\hat{v}_f \in \cdot \mid v_f) - \mathsf{Pois}(p v_f)\|_{tv} \leq p v_f^2 / M(\Delta). \tag{4.2}$$

By integrating this relation, we obtain $\|\mathbb{P}(\hat{v}_f \in \cdot) - \mathbb{Q}\|_{tv} \leq p\mathbb{E}(v_f^2)/M(\Delta)$. Similarly for $\tilde{v}_f$, with similar arguments, we have $\|\mathbb{P}(\tilde{v}_f \in \cdot) - \mathbb{Q}\|_{tv} \leq p^2 \mathbb{E}(v_f)$. Relation (4.1) is proved. The convergence in distribution is a direct consequence of Inequality (4.2). More details about Le Cam's inequality are given in chapter 5. □

Equation (4.1) implies that when the sampling parameter $p$ is small, the distribution of the number of sampled packets of a given flow is close to the analogue quantity obtained by probabilistic sampling.

Considering that if we deal with an elephant, the number of packets of the flow is quite large, the law of large numbers would suggest the following approximation $\widetilde{B}_1^f + \widetilde{B}_2^f + \cdots + \widetilde{B}_{v_f}^f \stackrel{\text{dist.}}{\sim} p v_f$, so that the total number of packets of a flow can be recovered from the number of sampled packets. In spite of the fact that this approximation is quite appealing and natural, it turns out that it has to be handled with care. Indeed, if $v_f$ is geometrically distributed, then it is easily checked that the above approximation is wrong. The fact that $v_f$ is, very likely, heavy tailed helps to establish such an approximation. This is the subject of the rest of the section. The following result is a first step in this direction.

**Proposition 8.** *If $h_k(x) = x^2/4p^2 \left( \sqrt{1 + 4kp/x^2} - 1 \right)^2$ $x \in \mathbb{R}$, $k > 0$, and the random variables $B_i$ are Bernoulli with mean $p$, then*

$$\left| \mathbb{P}\left( \sum_{i=1}^{v_f} B_i \geq k \right) - \mathbb{P}\left[ v_f \geq h_k\left( \sqrt{p(1-p)}\mathcal{G} \right) \vee k \right] \right| \leq c\mathbb{E}\left( \frac{1}{\sqrt{v_f}} \mathbb{1}_{\{v_f \geq k\}} \right),$$

*where $\mathcal{G}$ is a standard Gaussian random variable, for real numbers $a \vee b = \max(a,b)$, and $c = 3(p^2+(1-p)^2)/\sqrt{p(1-p)}$.*

*Démonstration.* Let $\sigma^2 = \mathrm{Var}(B) = p(1-p)$, $S_n = B_1 + \cdots + B_n$, $\bar{S_n} = S_n/n$ and $\hat{S_n} = \sqrt{n}(\bar{S_n} - np)/\sigma$. By Berry-Essen's theorem [39], for each $n \in \mathbb{N}$ and $k > 0$,

$$\left| \mathbb{P}\left(\hat{S_n} \geq \frac{k-pn}{\sigma\sqrt{n}}\right) - \mathbb{P}\left(\mathcal{G} \geq \frac{k-pn}{\sigma\sqrt{n}}\right)\right| \leq \frac{c}{\sqrt{n}}$$

where $c = 3E((p-B)^3)/\sigma^3 = 3(p^2+(1-p)^2)/\sqrt{p(1-p)}$. Thus, multiplying by $\mathbb{1}_{\{n \geq k\}}$, using the independence of $S_n$ and $v_f$ and Fubini's theorem, noticing that $\mathbb{P}(\hat{S}_N \geq (k-pv_f)/\sqrt{v_f}) = \mathbb{P}(S_{v_f} \geq k)$ and that, if $S_{v_f} \geq k$ then $v_f \geq k$, we obtain

$$\left| \mathbb{P}\left(\hat{S}_N \geq \frac{k-pv_f}{\sigma\sqrt{v_f}}\right) - \mathbb{P}\left(\mathcal{G} \geq \frac{k-pv_f}{\sigma\sqrt{v_f}}, v_f \geq k\right)\right| \leq c\mathbb{E}\left(\frac{1}{\sqrt{v_f}}\mathbb{1}_{\{v_f \geq k\}}\right).$$

Now, we prove that

$$\mathbb{P}\left(\mathcal{G} \geq \frac{k-pv_f}{\sigma\sqrt{v_f}}, v_f \geq k\right) = \mathbb{P}(pv_f + \sqrt{v_f}\sigma\mathcal{G} \geq k, v_f \geq k) = \mathbb{P}(v_f \geq f_k(\sigma\mathcal{G}) \vee k).$$

Indeed, denoting $z = \sqrt{y}$, the equation $pz^2 + zx - k = 0$ has two roots in $\mathbb{R}$, equal to $z_1 = (-x - \sqrt{x^2+4pk})/2p < 0$ and $z_2 = (-x + \sqrt{x^2+4pk})/2p > 0$. Thus, for every $x \in \mathbb{R}$, $pz^2 + zx - k \geq 0, z \geq 0$ is equivalent to $z \geq z_2$, i.e., $y \geq h_k(x)$. The result then readily follows. □

From the above result, under mild assumptions on the distribution of $v_f$, the tail distribution of $B_1 + B_2 + \cdots + B_{v_f}$ is related to the tail distribution of $v_f$. In particular, if $v_f$ has a Pareto distribution, we have the following result.

**Corollaire 1.** *If the random variable $v_f$ has a Pareto distribution, i.e. for some $b > 0$ and $a > 1$, $\mathbb{P}(v_f \geq k) = (b/k)^a$, and if the random variables $B_i$ are Bernoulli with mean $p$, then*

$$\lim_{k \to +\infty} \frac{\mathbb{P}\left(B_1 + B_2 + \cdots + B_{v_f} \geq k\right)}{\mathbb{P}(v_f \geq k/p)} = 1.$$

*Démonstration.* We have

$$\mathbb{P}(v_f \geq h_k(\sqrt{p(1-p)}\,\mathcal{G}) \vee l) = \mathbb{E}((b/(h_k(\sqrt{p(1-p)}\,\mathcal{G}) \vee k))^a) \sim (bp/k)^a,$$

since $h_k(x) = k/p(1 + O(\frac{1}{\sqrt{k}}))$ for large $k$. □

The above asymptotic results have been established for a random variable $v_f$, which has a Pareto distribution. But it is straightforwardly checked that similar results hold, when only the tail of $v_f$ is Pareto as for the traffic trace described in Section 4.2. To conclude the comparison between the original flow size distribution and the rescaled sampled size distribution, let us mention that Berry-Essen bound based on the normal approximation is accurate only around the mean value. To obtain a tighter bound on the tail of the distribution, it is possible to establish the following result (see [17] for details).

**Theorem 2.** *For $\alpha \in (1/2,1)$, there exist positive constants $C_0$ and $C_1$ such that for any $p \in (0,1)$ and $\ell \geq 1/p$,*

$$\left| \frac{\mathbb{P}\left(\sum_{i=1}^{v_f} B_i \geq \ell\right)}{\mathbb{P}(v_f \geq \ell/p)} - 1 \right| \leq \sup_{-C_1 \leq u \leq C_1} 3 \left| \frac{\mathbb{P}\left(v_f \geq \frac{\ell}{p} + u\left(\frac{\ell}{p}\right)^{\alpha}\right)}{\mathbb{P}(v_f \geq \ell/p)} - 1 \right| + \frac{C_0}{\mathbb{P}(v_f \geq \ell/p)} \exp\left(-\frac{p}{4(1-p)} \ell^{2\alpha-1}\right)$$

From the above result, we see that the quantity $\mathbb{P}\left(\sum_{i=1}^{v_f} B_i \geq \ell\right)$ related to the probability that a sampled flow contains at least $\ell$ packets is exponentially close for sufficiently large $\ell$ to $\mathbb{P}(v_f \geq \ell/p)$. In Section 4.4, the above theoretical results are used to interpret the experimental results when performing deterministic and random sampling on the France Telecom ADSL traffic traces.

### 4.3.3 Refinements

To prove Proposition 7, it has been assumed that flows are permanent. This assumption is reasonable, when the observation window length $\Delta$ is small. When $\Delta$ is large, however, flows may be bursty and alternate between on and off periods. To take into account this phenomenon, convergence to Poisson distributions as in Proposition 7 can be proved, when flows have different transmission rates on simple model designed for that purpose.

More precisely, let us assume that there are $L$ classes of flows. For a class $\ell \in \{1,\ldots,L\}$, $r_\ell(u)$ is the transmission rate of a flow of class $\ell$ at time $u$. The quantity $C_\ell = \frac{1}{\Delta}\int_0^\Delta r_\ell(u)\,du$ is the average transmission rate of a flow on $[0,\Delta]$. Flows are assumed to arrive uniformly in $[0,\Delta]$. Consequently, for each flow $f$ in class $\ell$, the number of packets transmitted up to time $t \in [0,\Delta]$ is $v_f(t) = \int_0^t r_\ell((u-\tau_f) \mod \Delta)\,du$, where the $\tau_f$'s are independent and uniformly distributed in $[0,\Delta]$. It follows that the different processes $v_f(t)$ for flows $f$ in class $\ell$ have the same distribution.

For $\ell \in \{1,\ldots,L\}$, let $K_\ell$ be the number of flows of class $\ell$ in $[0,\Delta]$ and $K = \sum_\ell K_\ell$ . The total number of transmitted packets up to time $u$ is denoted by $M(u) = \sum_{i=1}^K N_i(u)$. Let $pM(\Delta)$ be the number of sampling times between 0 and $\Delta$, $p$ denoting the sampling rate. When $K$ becomes large, assume that for every $\ell$, $K_\ell/K$ tends to a constant $\alpha_\ell$. By the law of large numbers, $M(u)/K$ converges almost surely to $C = \sum_{\ell=1}^L \alpha_\ell C_\ell$ for all $u \in [0,\Delta]$. The numbers of packets $\hat{v}_f$ in the sampled flows $f$ of class $\ell$ have the same distribution. We have the following result, whose proof is given in Appendix 4.6.

**Proposition 9.** *If $pM(\Delta)/K \to x$, the distribution of the number $n(\ell)$ of packets in a sampled flow of class $\ell$ converge to a Poisson distribution with parameter $xC_k/C$.*

The above proposition shows that the distribution of the number of sampled packets of a flow in class $\ell$ depends only on the ratio of the average rate of class $\ell$ to the total average rate in the observation window. This indicates that we could have considered the flows permanent at the average rate in the observation window.

## 4.4 Experimental results

In this section, we consider the traffic trace from the France Telecom backbone network described in Section 4.2 and we fix the length of the observation window equal to $\Delta = 3200$ seconds and the sampling rate $p = 1/100$. The complementary cumulative distribution functions (ccdf) of the number of packets in original mice and elephants are displayed in Figure 4.20(b)

and 4.20(d), respectively. In the original trace, there were 252,854 elephants and in the sampled trace, we found 132,352 and 132,185 of the original elephants with deterministic and random sampling, respectively.

From the above experimental results, we see that the probabilities of seeing elephants after sampling in the different cases are very close one to each other, about 0.523.

If $v_f$ has a Pareto distribution of the form $\mathbb{P}(v_f > k) = (b/k)^a \mathbb{1}_{\{k \geq b\}}$, the probability of seeing an elephant by random sampling is

$$\mathbb{P}\left(\sum_{i=1}^{v_f} B_i > 0\right) = 1-(1-p)^b + p\sum_{k=b}^{\infty}(1-p)^k \mathbb{P}(v_f > k) \sim bp + (bp)^a \Gamma(1-a, bp),$$

when $p$ is small. With $a = a_2 = 0.55$, $b = 20$, and $p = 1/100$, we find that the probability of seeing an elephant is approximately equal to 55 %, which is very close to the experimental value. Hence, by estimating the exponent $a$ of the Pareto distribution allows us to estimate the probability of seeing an elephant. This quantity is critical for the estimation of the parameters of flows. For instance, for estimating the original duration of flows, a method is presented in [9], but the estimation of $\nu$, the probability of seeing an elephant, is critical because it relies on the tails of some probability distributions. The method based on the estimation of the exponent of the Pareto distribution is more reliable.

The major difficulty for exploiting the sampled trace comes from the fact that we do not know if a sampled flow is really an elephant or not. If we had adopted the convention that a sampled flow corresponds to an elephant as soon as it is composed of at least two packets, we would have found 143,517 and 144,000 elephants with deterministic and random sampling, respectively. We see that this convention leads to slightly overestimating the number of elephants.

Figure 21 represents the ccdf of the number of packets in elephants after probabilistic and deterministic sampling, along with the rescaled original distribution $\mathbb{P}(N > k/p)/\nu$, where $\nu$ is the probability of seeing an elephant. We can observe that the three curves coincide, which is in agreement with the results obtained in Section 4.3. By using Proposition 8 and Theorem 2 and assuming that random and deterministic sampling are sufficiently close one to each other, we can recover the distribution of the original elephants from the distribution of sampled elephants with known bounds.

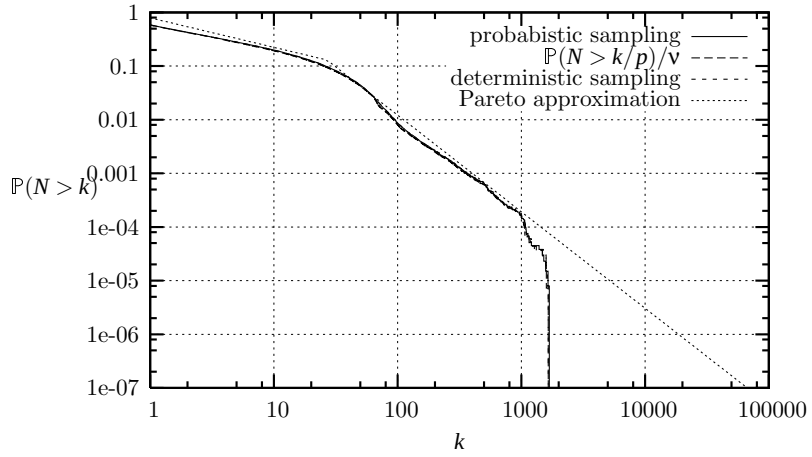
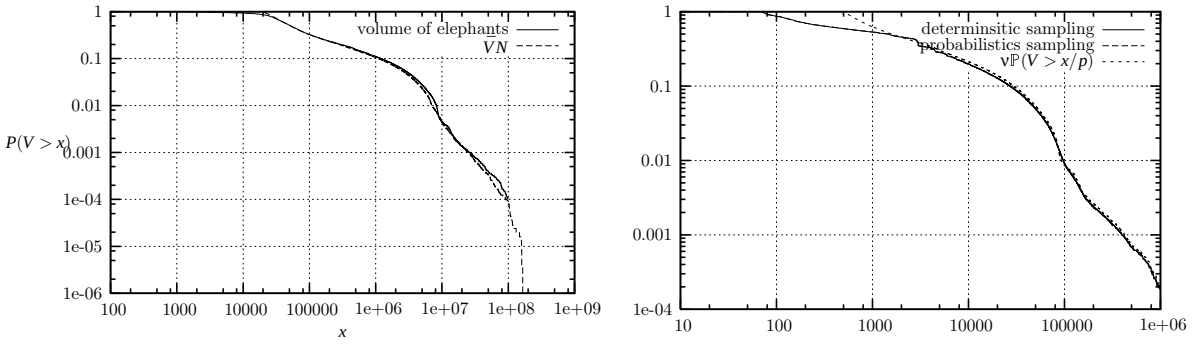



FIG. 21 – Number of packets in elephants after sampling and comparison with the rescaled original size $\mathbb{P}(N > k/p)/\nu$ along with the Pareto approximation.

For the volume $V$ (expressed in bytes) of elephants, we can first compute the mean number of bytes in packets. For instance, for the traffic trace considered in this chapter, the mean number of bytes in packets of elephants is equal to $\overline{V} = 1000$. Then, we can verify that multiplying the number of packets in elephants by the mean number $\overline{V}$ of bytes in packets give a fair estimate of the volume of elephants, as illustrated in Figure 4.22(a). From the results established for the number of packets in elephants and under the assumption that random sampling is sufficiently close to deterministic sampling, we can estimate the volume of original elephants with known bounds; Figure 4.22(b) shows that the rescaled distribution $\mathbb{P}(V > x/p)/\nu$ is close to the distribution of the volume of sampled elephants.




(a) Experimental curves

(b) Experimental curves

FIG. 22 – Volume (in bytes) of elephants after deterministic and probabilistic sampling and comparison with the rescaled original volume $\mathbb{P}(V > x/p)/\nu$.

## 4.5 Conclusion

We have shown in this chapter that as far as the volume and the number of packets in elephants are concerned, random and deterministic sampling are very close to each other, when the sampling rate becomes small. Several results for the number of packets contained in randomly sampled flows have been established. In particular, bounds between the distribution of the number of packets in a randomly sampled elephant and the rescaled original distribution have been established. Experimental results obtained by using a traffic trace from the France Telecom IP backbone network show good agreement with theoretical results.

## 4.6 Appendix : Proof of Proposition 9

Let $(t_j)_{1\le j\le pM(\Delta)}$ be the sequence of the $pM(\Delta)$ sampling times in $[0,\Delta]$. We have for any flow in class $\ell$, say, flow $i$

$$\mathbb{P}(\hat{v}_i = 0) = \mathbb{E}\left(\prod_{j=1}^{pM(\Delta)}\left(1-\frac{v_i(t_j)}{M(t_j)}\right)\right) = \mathbb{E}\left(e^{\sum_{j=1}^{pM(\Delta)}\log(1-v_i(t_j)/M(t_j))}\right),$$

where $\hat{v}_i$ is the number of packets in the sampled flow $i$. First, note that

$$\sum_{j=1}^{pM(\Delta)}\log\left(1-\frac{v_i(t_j)}{M(t_j)}\right) = -\sum_{j=1}^{pM(\Delta)}\frac{v_i(t_j)}{M(t_j)} + O\left(\frac{1}{K}\right).$$

Second, if $f$ is a twice continuously differentiable function in $[0,\Delta]$, we have

$$\sum_{j=1}^{pM(\Delta)} f(t_j) = \frac{pM(\Delta)}{\Delta}\int_0^\Delta f(u)\,du + \frac{f(\Delta)-f(0)}{2} + O\left(\frac{1}{pM(\Delta)}\right),$$

since the points $(t_j)$ are distributed more or less uniformly in $[0,\Delta]$. Hence, we have

$$\sum_{j=1}^{pM(\Delta)} \log\left(1-\frac{v_i(t_j)}{M(t_j)}\right) = -\frac{pM(\Delta)}{\Delta}\int_0^\Delta \frac{v_i(u)}{M(u)}\,du + \frac{1}{2}\frac{v_i(\Delta)}{M(\Delta)} + O(\frac{1}{K}).$$

The first term of the right-hand side is equal to $-\frac{x}{\Delta}\int_0^\Delta \frac{r_\ell(u-\tau_i)}{M(u)/K}\,du$, which converges a.s. to $-\frac{x}{C\Delta}\int_0^\Delta r_\ell(u-\tau_i)\,du = -xC_\ell/C$, when $K$ tends to $+\infty$. It follows that, when $K$ tends to $+\infty$,

$$\mathbb{P}(\hat{v}_i = 0) \to \exp\left(\frac{-xC_\ell}{C}\right). \tag{4.3}$$

For $k \in \mathbb{N}$,

$$\begin{aligned}\mathbb{P}(\hat{v}_i = k) &= \mathbb{E}\left(\sum_{i_1<\ldots<i_k}\prod_{m=1}^{k}\frac{v_i(t_{i_m})}{M(t_{i_m})}\prod_{j\notin\{i_1<\ldots<i_k\}}\left(1-\frac{v_i(t_j)}{M(t_j)}\right)\right)\\ &= \mathbb{E}\left(\prod_{j=1}^{pM(\Delta)}\left(1-\frac{v_i(t_j)}{M(t_j)}\right)\Sigma_k(g_i(t_1),\ldots,g_i(t_{pM(\Delta)}))\right),\end{aligned}$$

where $g_i(u) = v_i(u)/(M(u)-v_i(u))$ and $\Sigma_k = \sum_{i_1<\ldots<i_k}\prod_{j=1}^k X_{i_j}$ is the symmetric homogeneous polynomial of degree $k$. Denoting $S_i = \sum_{j=1}^{pM(\Delta)} X_j^i$ for $i>1$, Newton's formula

$$(-1)^k k\Sigma_k + \sum_{p=0}^{k-1}(-1)^p\Sigma_p S_{k-p} = 0 \quad (1 \le k \le pM(\Delta))$$

establishes that $\Sigma_k$ can be expressed as a function of $S_1,\ldots,S_k$. It is clear that $S_q(g_i(t_1),\ldots,g_i(t_{pM(\Delta)}))$ is the Riemann sum with $pM(\Delta)$ terms associated to $g_i^q$ on $[0,\Delta]$. Using Newton's formula, it can be proved that when $K$ tends to $+\infty$,

$$\Sigma_k(g_i(t_1),\ldots,g_i(t_{pM(\Delta)})) \sim \frac{\left(\sum_{j=1}^{pM(\Delta)} g_i(t_j)\right)^k}{k!}.$$

Taking into account approximation (4.3), we obtain that, for flow $i$ of class $\ell$,

$$\mathbb{P}(\hat{v}_i = k) \to e^{-x\frac{C_l}{C}}\frac{1}{k!}\mathbb{E}\left(\left(\frac{x}{C\Delta}\int_0^\Delta r_l(u-\tau_i)\,du\right)^k\right) = \frac{e^{-x\frac{C_\ell}{C}}}{k!}\left(\frac{xC_\ell}{C}\right)^k$$

and Proposition 9 follows.

# 5

# On the Statistical Characterization of Flows in Internet Traffic with Application to Sampling

**Contents**



A new method of estimating some statistical characteristics of TCP flows in the Internet is developed in this chapter. For this purpose, a new set of random variables (referred to as observables) is defined. When dealing with sampled traffic, these observables can easily be computed from sampled data. By adopting a convenient mouse/elephant dichotomy also *dependent on traffic*, it is shown how these variables give a reliable statistical representation of the number of packets transmitted by large flows during successive time intervals with an appropriate duration. A mathematical framework is developed to estimate the accuracy of the method. As an application, it is shown how one can estimate the number of large TCP flows when only sampled traffic is available. The algorithm proposed is tested against experimental data collected from different types of IP networks.

## 5.1 Introduction

We investigate in this chapter how to characterize the statistical properties of the sizes of large flows (notably their number of packets) in Internet traffic. It is commonly observed in the technical literature and in real experiments that the total size (in packets or bytes) of such flows has a heavy tailed distribution. In practice, however, this characterization holds only for very large values of the flow size. Consequently, in order to accurately estimate the tail of the size probability distribution, a large number of large flows is necessary. To increase the sample size when empirically estimating probability distribution tails, one is led to increase the length of the observation period. But the counterpart is that the distribution of the flow size can no more be described in terms of simple probability distributions, of the Pareto type for example. This is due to the fact that traffic is not stationary over long time periods, for instance because of daily variations of interactive services (video, web, etc.).

Actually, numerous approaches have been proposed in the technical literature in order to model large flows as well as their superposition properties. One can roughly classify them in two categories : signal processing models and statistical models. Using ideas from signal processing, Abry and Veitch [2], see also Feldman *et al.* [37, 38] and Crovella and Bestravos [25], describe the spectral properties of the time series associated with IP traffic by using wavelets. In this way, a characterization of long range dependence (the Hurst parameter for example) can be provided. Straight lines in the log-log plot of the power spectrum support some of the "fractal" properties of the IP traffic, even if they may simply be due to packet bursts in data flows. See Rolland *et al.* [69].

Signal processing tools provide information on aggregated traffic but not on characteristics on individual TCP flows, like the number of packets or their transmission time. For statistical models, a representation with Poisson shot noise processes (and therefore some independence properties) has been used to describe the dynamics of IP traffic, see Hohn and Veitch [49], Duffield *et al.* [28], Gong *et al.* [46], Barakat *et al.* [10] and Krunz and Makowski [53] for example. In Ben Azzouna *et al.* [9], Loiseau *et al.* [56, 55] and Gong *et al.* [46], the distribution of the size of large flows is represented by a Pareto distribution, i.e. a probability distribution whose tail decays on a polynomial scale. ,

The starting point of some of these analyses is the need for understanding the relation between the distribution of the number $\hat{v}$ of sampled packets when performing packet sampling and the distribution of the flow size $v$. The problem can be described as follows : $\mathbb{P}(\hat{v} = j) = Q(\mathbb{P}(v = \cdot), j),\ j \geq 1$, with

$$Q(\phi, i) = p^j \sum_{\ell=j}^{+\infty} \binom{\ell}{j} (1-p)^{\ell-j} \phi(\ell).$$

The problem then consists of finding a distribution $\phi_0$ maximizing some functional $\mathcal{L}(\phi)$ so that the relation $\mathbb{P}(\hat{v} = j) = Q(\phi, j)$ holds. See Loiseau *et al.* [55] for an extensive discussion of the current literature where our algorithm is called "stochastic counting". As it will be seen in the following, we will not rely on the maximum likelihood ratio of distributions in our approach but on estimations of some averages to estimate some key parameters.

**Statistical Characterization Method** We develop in this chapter an alternative method of obtaining a statistical description of the size of large flows in IP traffic by means of a Pareto distribution : Statistics are collected during successive time windows of limited length (instead of one single time window for the whole trace). It must be emphasized that this characterization

in terms of a Pareto distribution does not rely on the asymptotic behavior of the tail distribution but only on statistics on some range of values for the sizes of flows.

The advantage of the proposed method is that with a careful procedure, a simple statistical characterization is possible and seems to be quite reliable as shown by our experiments for various sets of traffic traces. The intuitive reason for considering short time periods is that on such times scales, flows exhibit only one major statistical mode (typically a Pareto behavior). In larger time windows, different modes due to the wide variety of flows and non-stationarity in IP traffic necessarily appear. (See Feldman *et al.* [38].) This approach allows us to establish a reliable statistical characterization of flows which is used to infer information from sampled traffic as it will be seen. The counterpart of that the distribution of the *total* size of a large flow (obtained when considering the complete traffic trace) cannot be obtained directly in this way since the trace is cut into small pieces.

An algorithm is proposed to obtain the statistical representation of large flows when all the packets of the trace are available. The constants used in our algorithms are explicitly expressed as either universal constants (independent of traffic) or constants depending on traffic : Length of the observation window, definition of TCP flows referred to as large flows, etc. The procedure invoked to estimate flow statistics should not depend on some hidden pre-processing of the trace. Our algorithms determine *on-line* the constants depending on the traffic. This is, in our view, one important aspect which is sometimes neglected in the technical literature.

**Application to Sampled Traffic** The basic motivation for developing a flow characterization method is to infer flow characteristics from sampled data. This is notably the case for sampling processes such as the 1-out-of-$k$ sampling scheme implemented by CISCO's NetFlow [24], which greatly degrades information on flows. What we advocate in this chapter is that it is still possible to infer relevant characteristics on flows from sampled data if some characteristics of the flow size can be confidently described by means of a simple Pareto distribution. By using the statistical representation described above, we propose a method of inferring the number of large flows from sampled traffic.

The proposed method relies on a new set of random variables, referred to as observables and computed in successive time intervals with fixed length. Specifically, these random variables count the number of flows sampled once, twice or more in the successive observation windows. The properties of these variables can be obtained through simple characteristics, in particular *mean values* of variables instead of *remote quantiles* of the tail distribution, which are much more difficult to accurately estimate. By developing a convenient mathematical setting (Poisson approximation methods), it is moreover possible to show that quantities related to the observables under consideration are close to Poisson random variables with an explicit bound on the error. This Poisson approximation is the key result to estimate the total number of large flows.

**Organization of the chapter** The organization of the chapter is as follows. A statistical description of large TCP flows is presented in Section 5.2, this representation is tested against five exhaustive sets of traffic traces : three from the France Telecom (FT) commercial IP network carrying residential ADSL traffic and two others from Abilene network. An algorithm is developed in this section to compute the characteristics of the Pareto distributions describing flows. In Section 5.3, some assumptions on sampled traffic are introduced and the observables for describing traffic are defined. The mathematical properties are analyzed in light of Poisson approximation methods in Section 5.4. The results developed in this section are crucial to infer the statistics of an IP traffic from sampled data. Experiments with the five sets of sampled traces

used in this chapter are presented and discussed in Section 5.5. Some concluding remarks are presented in Section 5.6.

## 5.2 Statistical Properties of Flows

This section is devoted to a statistical study of the size (the number of packets) of flows in a limited time window of duration $\Delta$. The goal of this section is show that a simple statistical representation of the flow size can be obtained for various sets of traffic traces.

### 5.2.1 Assumptions and Experimental Conditions

#### The sets of traces used for testing theoretical results

For the experiments carried out in the following sections, several sets of traces will be considered : Commercial IP traffic, namely ADSL traces from the France Telecom (FT) IP collect network, and traffic issued from campus networks (Abilene III traces). Their characteristics are given in Table 5.1.

TAB. 5.1 – Characteristics of traffic traces considered in experiments.

| Name | Nb. IP packets | Nb. TCP Flows | Duration |
|---|---|---|---|
| ADSL Trace A | 271 455 718 | 20 949 331 | 2 hours |
| ADSL Trace B Upstream | 54 396 226 | 2 648 193 | 2 hours |
| ADSL Trace B Downstream | 53 391 874 | 2 107 379 | 2 hours |
| Abilene III Trace A | 62 875 146 | 1 654 410 | 8 minutes |
| Abilene III Trace B | 47 706 252 | 1 826 380 | 8 minutes |

Trace ADSL A has been used in chapter 4 (captured in October 2003) The Abilene traces 20040601-193121-1.gz (trace A) and 20040601-194000-0.gz (trace B) can be found at the url http ://pma.nlanr.net/Traces/Traces/long/ipls/3/.

#### Time Windows

Traffic will be observed in successive time windows with length $\Delta$. In practice, the quantity $\Delta$ can vary from a few seconds to several minutes depending upon traffic characteristics on the link considered.

The ideal value of $\Delta$ actually depends on the targeted application. For the design of network elements considering the flow level (e.g., flow aware routers, measurement devices, etc.), it is necessary to estimate the requirements in terms of memory to store the different flow descriptors. In this context, $\Delta$ may be of the order of few seconds. The same order of magnitude is also adapted to anomaly detection, for instance for detecting a sudden increase in the number of flows. For the computation of traffic matrices, $\Delta$ can be several minutes long (typically 15 minutes). In our study, the "adequate" values for $\Delta$ are of the order of several seconds. See the discussion below.

#### Mice and Elephants

With regard to the analysis of the composition of traffic, in light of earlier studies on IP traffic (see Estan and Estan-1 [36], Papagiannaki *et al.* [66] or Ben Azzouna *et al.* [8]), two

types of flows are identified : small flows with few packets (referred to as mice) and the other flows will be referred to as elephants. In commercial IP traffic, this simple traffic decomposition can be justified by the predominance of web browsing and peer-to-peer traffic giving rise to either signaling and very small file transfers (mice) or else file downloads (elephants).

This dichotomy may be more delicate to verify in a different context than the one considered in Ben Azzouna *et al.* [8]. For LAN traffic, for example, there may be very large amounts of data transferred at very high speed. As it will be seen in the various IP traces used in our analysis, the distinction between mice and elephants has to be handled with care and in our case is dependent on the type of traffic considered. The distinction between the constants depending on the trace and "universal" constants is, in our view, a crucial issue. It amounts to precisely stating which constants are *depend* on traffic. This aspect is generally (unduly in our opinion) neglected in traffic measurement studies. In particular, the variable $\Delta$ and the dichotomy mice/elephants are dependent on the trace, as explained in the next section.

### 5.2.2 Heavy Tails

The fact that the distribution of the size $v$ of a large TCP flow is heavy tailed is well known. Experiments and theoretical results on the superposition of ON-OFF heavy tailed traffic have justified the self similar nature of IP traffic, see Crovella and Bestravos [25]. Although the heavy tailed property of the size of large flows is commonly admitted, little attention has been paid to identify properly a class of heavy tailed distributions so that the corresponding parameters can be estimated for an arbitrary traffic trace with a significant duration.

One of the reasons for this situation is that the most common heavy tailed distributions $G(x) = \mathbb{P}(v \geq x)$ (e.g., Pareto, i.e., $G(x) = C/x^{\alpha}$ for $x \geq b$ and some $\alpha > 0$, or Weibull, i.e., $G(x) = \exp(-\nu x^{\beta})$ for some $\beta > 0$ and $\nu > 0$) have a very small number of parameters and consequently a limited of number of possible degrees of freedom for describing the distribution of the sizes of flows. For this reason, such a distribution can rarely represent the statistics of the total number of packets transmitted by a flow in a trace of arbitrary duration.

As a matter of fact, if a traffic trace is sufficiently long, some non stationary phenomena may arise and the diversity of file sizes may not be captured by one or two parameters. For example, with a Pareto distribution, the function $x \to G(x)$ in a log-log scale should be a straight line. The statistics of the file sizes in the traces used in our experiments are depicted in Figure 23 and 24 for an ADSL traffic trace from the France Telecom backbone IP collect network and for a traffic trace from Abilene network, respectively.

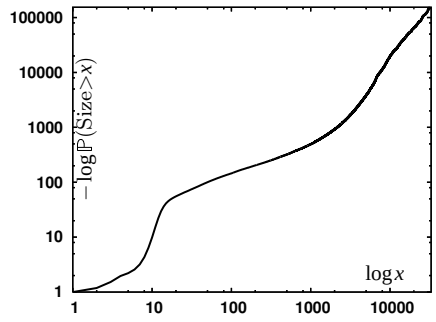



FIG. 23 – Statistics of the number of packets $v$ of a flow for ADSL A (2 hours) : the quantity $-\log(\mathbb{P}(v > x))$ as a function of $\log(x)$.

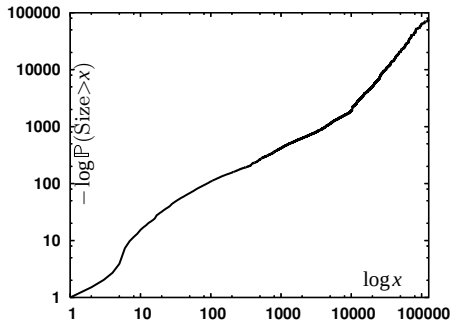



FIG. 24 – Statistics of the number of packets $v$ of a flow for ABILENE A trace (8 minutes) : the quantity $-\log(\mathbb{P}(v > x))$ as a function of $\log(x)$.

Figure 23 and 24 clearly show that for the two traffic traces considered, the file size exhibits a multimodal behavior : At least *several* straight lines should be necessary to properly describe these distributions. These figures also exhibit the (intuitive) fact that has been noticed in earlier experiments : The longer the trace is, the more marked is the multimodal phenomenon. (See Ben Azzouna *et al.* [9] for a discussion.)

The *key observation* when characterizing a traffic trace is the fact that if the duration $\Delta$ of the successive time intervals used for computing traffic parameters is appropriately chosen, then the distribution of the size of the main contributing flows in the time interval can be represented by a Pareto distribution. More precisely, there exist $\Delta$, $B_{min}$, $B_{max}$ and $a > 0$ such that if $v$ is the number of packets transmitted by a flow in $\Delta$ time units, then $\mathbb{P}(v \geq x \mid v \geq B_{min}) \sim P_\alpha(x)$ for $B_{min} \leq x \leq B_{max}$ with

$$P_\alpha(x) \stackrel{\text{def.}}{=} \left(\frac{B_{min}}{x}\right)^a, \text{ for } x \geq B_{min}, \tag{5.1}$$

and furthermore the proportion of large flows with size greater than $B_{max}$ is less than 5%. The parameter $B_{min}$ is usually referred to as the location parameter and $a$ as the shape parameter.

In other words, if the time interval is sufficiently small then the distribution of the number of packets transmitted by a large flow has one dominant Pareto mode and therefore can confidently be characterized by a unique Pareto distribution. The algorithm used to validate this result is described in Table 5.2. It is run from the beginning of the trace ; in practice a couple of minutes is sufficient to obtain results for the constants $\Delta$, $B_{min}$, $B_{max}$. The algorithm is of course valid when the total trace is available for at least an interval of several minutes. In the case of sampled traffic for which this algorithm cannot be used, another method will be proposed in Section 5.3.

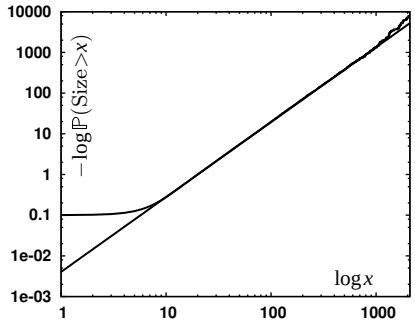
The quantity $B_{min}$ defines the boundary between mice and *elephants* in the trace. A *mouse* is a flow with a number of packets less than $B_{min}$. An elephant is a flow such that its number of packets during a time interval of length $\Delta$ is greater than or equal to $B_{min}$. By definition of $B_{max}$, flows whose size is greater than $B_{max}$ represent a small fraction of the elephants.

### 5.2.3 Experiments with Synthetic and Real Traffic Traces

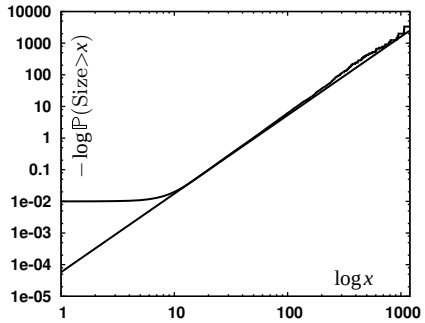
Some experiments have been done using artificial traces with a real Pareto distribution. For these traces, the algorithm described in Table 5.2 has been used without any modification : A time window is defined when at least 1000 flows of size greater than 20 packets are detected. As it can be seen, the identification of the exponent $a$ is quite good. Note that, because only Pareto distributed flows are present the minimal size $B_{min}$ of elephants is smaller than in real traffic.

TAB. 5.2 – Algorithm for Identifying $\Delta$ and the Pareto Distribution.

— $\Delta$ is fixed so that at least 1000 flows have more than 20 packets.
— $B_{max}$ is defined as the smallest integer such that less than 5% of the flows have a size greater than $B_{max}$.
— A Least Square Method, see Deuflhard and Hohmann [27] for example, is performed to get a linear interpolation in a log-log scale of the distribution of sizes between $B_{min}$ and $B_{max}$. The constant $B_{min}$ is chosen as the smallest integer such that the $L_2$-distance in the sense of least square method with the approximating straight line is less than $2.10^{-3}$. The slope of the line gives the value of the parameter $a$.




(a) Pareto $a = 1.85$. Estimation : $\hat{a} = 1.84$, $B_{min} = 9$, $B_{max} = 100$




(b) Pareto $a = 2.5$. Estimation : $\hat{a} = 2.48$, $B_{min} = 11$, $B_{max} = 65$

FIG. 25 – Synthetic traces with $10^6$ flows with a Pareto distribution

Experimental results with real traces, for the ADSL A and Abilene A traffic traces, are displayed in Figures 26 and 27, respectively. The same algorithm has been run for the ADSL trace B Upstream and Downstream as well as for the Abilene III B trace. The benefit of the algorithm is that the distribution of the number of packets in elephants can always be represented by a unimodal Pareto distribution if the duration of $\Delta$ is adequately chosen by using the algorithm given in Table 5.2. Results are summarized in Table 5.3.

TAB. 5.3 – Statistics of the elephants for the different traffic traces.

| | ADSL A | ADSL B Up | ADSL B Down | Abilene A | Abilene B |
|---|---|---|---|---|---|
| $\Delta$ (sec) | 5 | 15 | 15 | 2 | 2 |
| $B_{min}$ | 20 | 29 | 39 | 89 | 79 |
| $B_{max}$ | 94 | 154 | 128 | 324 | 312 |
| $a$ | 1.85 | 1.97 | 1.50 | 1.30 | 1.28 |

### 5.2.4 On the choice of parameters

We discuss in this section the various parameters used by the algorithm.

#### Fixed parameters and parameters depending on traffic

There are four basic parameters for the model which are determined by the trace : $\Delta$ (duration of time window for statistics), the range of values $[B_{min}, B_{max}]$ for the Pareto distribution and the exponent $a$ of this distribution. These parameters are discussed below.

Additionally there are "universal" (i.e. independent of the trace) : the minimal number of flows to make statistics, set to 1000 here, the proportion, 5%, of flows of size $\geq B_{max}$, and the level of accuracy, $2.10^{-3}$ here, of the least square method to determine $B_{min}$ and $B_{max}$.

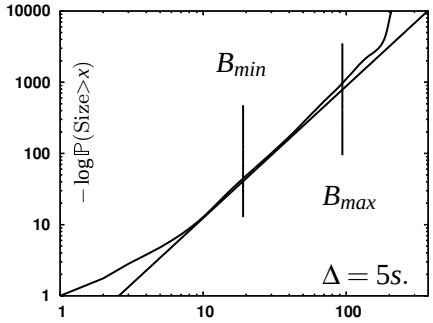
#### Parameter $B_{min}$

It turns out that for commercial (ADSL) traffic, the value of $B_{min}$ is close to 20. This value is fairly common in earlier studies for classifying ADSL traffic. It should be noted that this value is not at all universal since, in our view, it does depend on traffic. The examples with Abilene traces, see below, which contain significantly bigger elephants, shows that the corresponding values should be higher than 20 (around 80 in our example).

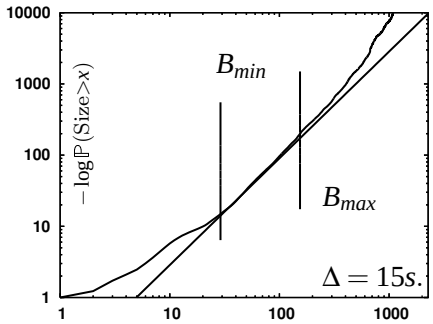
The two types of traffic are intrinsically different : ADSL traffic is mainly composed of peer to peer traffic (with a huge number of small flows and a few file transfers of limited size because of the segmentation of large files into chunks), while Abilene traffic comprises large file transfers issued from campus networks. In order to maximize the range for the Pareto description, the variable $B_{min}$ is defined as the smallest value for which the linear representation (in the log scale) holds.

#### Parameter $\Delta$

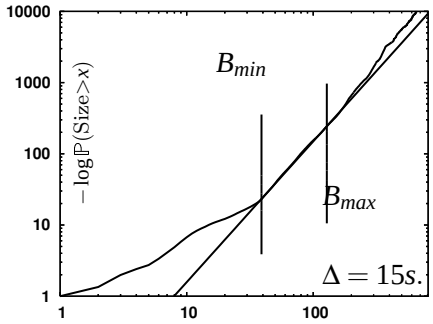
This parameter $\Delta$ is determined in a simple way by our algorithm. According to the various experiments, the parameter $\Delta$ can be taken in some range of values where the Pareto representation still holds. On the one hand, $\Delta$ has to be taken large enough so that sufficiently many




(a) ADSL A trace – $\Delta = 5$s

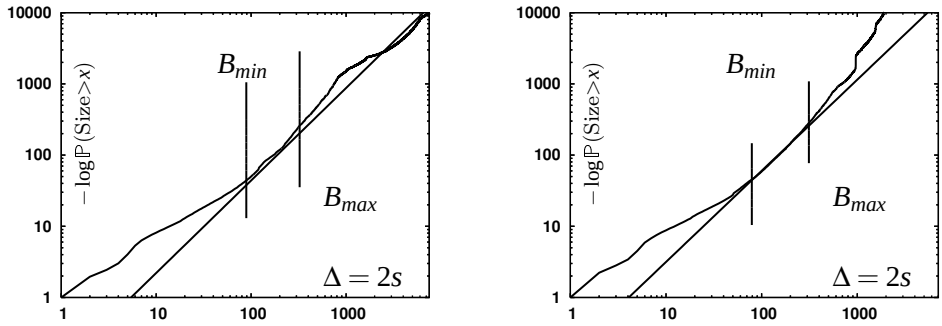



(b) ADSL B Down trace – $\Delta = 15$ seconds




(c) ADSL B Up trace – $\Delta = 15$ s

FIG. 26 – Statistics of the flow size (number of packets) in a time interval of length $\Delta = 15$




(a) Abilene A trace – $\Delta = 2$ s (b) Abilene B trace – $\Delta = 2$ s

FIG. 27 – Statistics of the flow size (number of packets) in a time interval of length $\Delta$ for the Abilene traces.

packets arrive in time intervals of duration $\Delta$ to derive reliable estimations of the Pareto distribution. An experiment with ADSL A trace with $\Delta = 1s$ gives only 63 flows of size more than 20 which is not enough to obtain reliable statistics. A "correct" value in this case is $5s$. Experiments show that higher values (like $10s$) do not change significantly the Pareto property observed in this case.

On the other hand, $\Delta$ should not be too large so that the statistical properties (a Pareto distribution in our case) can be identified, i.e., so that the statistics are unimodal. See Figures 23 and 24 which illustrate situations where statistics are done on the complete trace, i.e. when $\Delta$ is taken equal to the total duration of the trace. In these examples, the piecewise linear aspect of the curves suggests, for both cases, there is at least a bi-modal Pareto behavior.

### 5.2.5 Discussion

As it will be seen in the following, the above statistical model gives interesting results to extract information from sampled traffic. It has nevertheless some shortcomings which are now discussed.

#### A partial information when $\Delta$ is small

It should be noted that the parameters computed in a time window of length $\Delta$ do not give a complete description of the distribution of the size of a large flow, since statistics are done over a limited time horizon. The procedure provides therefore a fragmented information.

To obtain a complete description of the statistics of the size of flows, it would be necessary to relate the statistics from successive time windows of length $\Delta$. We do not know how to do that yet. Nevertheless, as it will be seen in the following, this fragmented information can be recovered from sampled traffic and it will be used to give a good estimation on the number of active large flows at a given time. This incomplete but useful description of the statistics is, in some sense, the price to pay to have a simple estimation of the statistics of flows.

#### An incomplete description of large flows in a time window of size $\Delta$

The representation with a Pareto distribution is for elephants (with size greater than $B_{min}$) whose size is less than $B_{max}$. In particular, it does not give any information on the statistics of

flows with size greater than $B_{max}$. But note that, by definition, less that 5% of the total number of flows have a size greater than $B_{max}$. This is however a source of errors when, as in Section 5.4, the Pareto representation is used on the interval $[B_{min}, +\infty]$ instead of $[B_{min}, B_{max}]$.

## 5.3 Sampled Traffic : Assumptions and Definition of Observables

In the previous section, an algorithm to describe the distribution of large flows by means of a unimodal distribution has been introduced. Now, it is shown how to exploit this algorithm in the context of packet sampling in the Internet. Packet sampling is a crucial issue when performing traffic measurements in high speed backbone networks. As a matter of fact, a fundamental problem related to the computation of flow statistics from traffic crossing very high speed transmission links is that, due to the enormous number of packets handled by routers, only a reduced amount of information can be available to the network operator.

We describe in this section the different assumptions made on traffic in order to develop an analytical evaluation of our method of inferring flow statistics. Throughout this chapter, high speed transmission links (at least 1 Gbit/s) will be considered.

### 5.3.1 Mixing condition

When observing traffic, packets are assumed to be sufficiently interleaved so that those packets of a same flow are not back-to-back but mixed with packets of other flows. This introduces some randomness in the selection of packets when performing sampling. In particular, when $K$ flows are active in a given time window and if the $i$th flow comprises $v_i$ packets during that period, then the probability of selecting a packet of the $i$th flow is assumed to be equal $v_i/(v_1+v_2+\cdots+v_K)$. This property will be referred to as *mixing condition* in the following and is formally defined as follows. A variant of this property is, implicitly at least, assumed in the existing literature. See, e.g. Duffield *et al.* [29] and Chabchoub *et al.* [18]. It has also been discussed in Chapter 4, Section 4.3.1 as the permanent flow property.

**Definition 3** (Mixing Condition)**.** *If $K$ TCP flows are active during a time interval of duration $\Delta$, traffic is said to be mixing if for all $i$, $1 \leq i \leq K$, the total number $\hat{v}_i$ of packets sampled from the $i$th flow during that time interval has the same distribution as the analog variable in the following scenario : at each sampling instant a packet of the $i$th flow is chosen with probability $v_i/V$ where $v_i$ is the number of packets of the $i$th flow and $V = v_1 + \cdots + v_K$.*

This amounts to claim that with regard to sampling, the probability of selecting a packet of a given flow is proportional to the total number of packets of this flow.

One alternative would consist of assuming that the probability of selecting a packet of the $i$th flow is $1/K$, the inverse of the total number of flows. This assumption, however, does not take into account the respective contributions of the different flows to the total volume and thus may be inaccurate. If all $K$ flows had the same distribution with a small variance, then this assumption would not much differ from the mixing condition. Note however that the variance of Pareto distributions can be infinite if the shape parameter $a$ is less than 2. Hence, this leads us to suppose that the mixing condition holds and that the probability of selecting a packet from flow $i$ is indeed $v_i/V$.

### 5.3.2 Negligibility assumption

We consider traffic on very high speed links and it then seems reasonable to assume that no flows contribute a significant proportion of global traffic. In other words, we suppose that the contribution of a given flow to global traffic is negligible. In the following, we go one step further by assuming that in any time window, the number of packets of a given flow is negligible when compared to the total number of packets in the observation window. By using the notation of the previous section, this amounts to assuming that for any flow $i$, the number of packet $v_i$ is much less than $V$. Furthermore, we even impose that the squared value of $v_i$ is much less than $V$. We specifically formulate the above assumptions as follows.

**Definition 4** (Negligibility condition)**.** *In any window of length $\Delta$, the square of the number of packets of every flow is negligible when compared to the total number of packets $V$ in the observation window. There specifically exists some $0 < \varepsilon \ll 1$ such that for all $i = 1, \ldots, K$, $v_i^2/V \leq \varepsilon$.*

The above assumption implies that no flows are dominating when observing traffic on a high speed transmission link. Table 5.4 shows that this is the case for the traces used in our experiments. There is thus no bias in the sampling process, which may be caused by the fact that some flows are oversampled because they contribute a significant part of traffic. This assumption is reasonable for commercial ADSL traffic because access links are often the bottlenecks in the network. For instance, ADSL users may have access rates of a few Mbit/s, which are negligible when compared against backbone links of 1 to 10 Gbit/s. Moreover, the bit rate achievable by an individual flow rarely exceeds a few hundreds of Kbit/s. In the case of transit networks carrying campus traffic, the above assumption may be more questionable since bulk data transfers may take place in Ethernet local area networks and individual flows may achieve bit rates of several Mbit/s.

TAB. 5.4 – The quantity $\mathbb{E}(v_1^2)/\mathbb{E}(V)$ for traffic traces considered in experiments.

| Trace | $\Delta = 5$sec | $\Delta = 10$sec | $\Delta = 15$sec |
|---|---|---|---|
| ADSL A | 0.000146 | 0.000159 | 0.000168 |
| ADSL B up | 0.001100 | — | 0.001335 |
| ADSL B Down | 0.002199 | 0.002543 | 0.002732 |

| Trace | $\Delta = 1$sec | $\Delta = 2$sec | $\Delta = 3$sec | $\Delta = 5$sec |
|---|---|---|---|---|
| Abilene A | 0.055001 | 0.068833 | 0.064813 | 0.072768 |

| Trace | $\Delta = 1$sec | $\Delta = 2$sec |
|---|---|---|
| Abilene B | 0.011786 | 0.013804 |

### 5.3.3 The Observables

We now introduce the different variables used to infer flow characteristics. These variables are based only upon sampled data ; they can be evaluated when analyzing NetFlow records sent by routers of an IP network. For this reason, these variables are referred to as observables. Because

of packet sampling, recall that the original characteristics of flows (for instance their duration or their original number of packets) cannot be directly observed.

The observables considered in this chapter to infer flow characteristics are the random variables $W_j,\ j \geq 1$, where $W_j$ is the number of flows sampled $j$ times during a time interval of duration $\Delta$. The averages of the random variables $W_j$ are in fact the key quantities used to infer the characteristics of flows from sampled data.

The random variables $W_j,\ j \geq 1$ are formally defined as follows : Consider a time interval of length $\Delta$ and let $K$ be the total number of large flows present in this time interval. Each flow $i \in \{1,\ldots,K\}$ is composed of $v_i$ packets in this time interval. Let denote by $\hat{v}_i$ the number of times that flow $i$ is sampled. The random variable $W_j$ is simply defined by

$$W_j = \mathbb{1}_{\hat{v}_1=j} + \mathbb{1}_{\hat{v}_2=j} + \cdots + \mathbb{1}_{\hat{v}_K=j}. \tag{5.2}$$

In practice, if $\Delta$ is not too large, the data structures used to compute the variables $W_j$ are reasonably simple. Moreover, as it will be seen in the following, provided that $\Delta$ is appropriately chosen, the statistics of the number of packets transmitted by elephants during successive time windows with duration $\Delta$ are quite robust. Consequently, the variables $W_j$ inherit also this property. When the number of large flows is large, the estimation of the asymptotics of their averages from the sampled traffic is easy in practice. Theoretical results on these variables are derived in the next section.

## 5.4 Mathematical Properties of the Observables

### 5.4.1 Definitions and Le Cam's inequality

For $j \geq 0$, the variable $W_j$ defined by Equation (5.2) is a sum of Bernoulli random variables, namely

$$W_j = \mathbb{1}_{\{\hat{v}_1=j\}} + \mathbb{1}_{\{\hat{v}_2=j\}} + \cdots + \mathbb{1}_{\{\hat{v}_K=j\}},$$

where $\hat{v}_i$ is the number of times that the $i$th flow has been sampled. If these indicator functions were independent, by assuming that $K$ is large, one could use to estimate the distribution of $W_j$ either via a Poisson approximation (in a rare event setting) or via a central limit theorem (in a law of large numbers context). Since the total number of samples is known, the sum of the random variables $\hat{v}_i$ for $i = 1,\ldots,K$ is known and then, the Bernoulli variables defining $W_j$ are *not* independent.

To overcome this problem, we make use of general results on the sum of Bernoulli random variables. Let us consider a sequence $(I_i)$ of Bernoulli random variables, i.e. $I_i \in \{0,1\}$. The distance in total variation between the distribution of $X = I_1 + \cdots + I_i + \cdots$ and a Poisson distribution with parameter $\delta > 0$ is defined by

$$\|\mathbb{P}(X \in \cdot) - \mathbb{P}(Q_\delta \in \cdot)\|_{tv} \overset{\text{def.}}{=} \sup_{A \subset \mathbb{N}} |\mathbb{P}(X \in A) - \mathbb{P}(Q_\delta \in A)| = \frac{1}{2}\sum_{n \geq 0} \left| \mathbb{P}(X = n) - \frac{\delta^n}{n!} e^{-\delta} \right|.$$

The Poisson distribution $Q_\delta$ with mean $\delta$ is such that

$$\mathbb{P}(Q_\delta = n) = \frac{\delta^n}{n!} \exp(-\delta).$$

Note that the total variation distance is a strong distance since it is uniform with respect to all events, i.e., for all subset s $A$ of $\mathbb{N}$,

$$|\mathbb{P}(X \in A) - \mathbb{P}(Q_\delta \in A)| \leq \|\mathbb{P}(X \in \cdot) - \mathbb{P}(Q_\delta \in \cdot)\|_{tv}.$$

The following result (see Barbour *et al.* [11]) gives a tight bound on the total variation distance between the distribution of $X$ and the Poisson distribution with the same expected value when the Bernoulli variables are independent. In spite of the fact that this result is not directly applicable in our case, we shall show in the following how to use it to obtain information on the distributions of the observables $W_j$.

**Theorem 3** (Le Cam's Inequality)**.** *If the random variables* $(I_i)$ *are independent and if* $X = \sum_i I_i$*, then*

$$\|\mathbb{P}(X \in \cdot) - \mathbb{P}(Q_{\mathbb{E}(X)} \in \cdot)\|_{tv} \leq \sum_i \mathbb{P}(I_i = 1)^2 \leq \mathbb{E}(X)^2 = \mathbb{E}(X) - \mathrm{Var}(X) \quad (5.3)$$

If $X$ is a Poisson distribution then $\mathrm{Var}(X) = \mathbb{E}(X)$, the above relation shows that to prove the convergence to a Poisson distribution one has only to prove that the expectation of the random variable is arbitrarily close to its variance.

### 5.4.2 Estimation of the mean value of the observables

We consider the 1-out-of-$k$ deterministic sampling technique, where one packet is selected every other $k$ packets. In addition, we suppose that traffic on the observed link is sufficiently mixed so that the mixing condition given by Definition 3 holds and that there are no dominating flows in traffic so that the negligibility condition (Definition 4) also pertains.

It is assumed that during a time interval of length $\Delta$, there are $K$ flows composed of at least $B_{min}$ packets, where $B_{min}$ is defined in Section 5.2. It has been seen that the number of packets in these flows follows a Pareto distribution defined by Relation (5.1) for some exponent $a$ and parameters $B_{min}$ and $B_{max}$. Let $v$ be a random variable whose distribution is given by Relation (5.1) for all $x \geq B_{min}$. From our experiments, $v$ is the size of a "typical" flow whose size is in the interval $[B_{min}, B_{max}]$. See the discussion at the end of Section 5.2 for the flows of size greater than $B_{max}$. Of course the sizes of mice are not represented by this random variable. The variable $V$ denotes the total number of packets in the observation window, note that it includes not only the elephants but also the mice.

Note that $V$ is the sum of the number of packets in elephants and mice. If $v_i$ is the number of packet in the $i$th elephant, then $v_i$ has the same Pareto distribution as $v$ (i.e., $v_i \stackrel{\text{dist.}}{=} v$) and $V \geq v_1 + v_2 + \cdots + v_K$. The difference $V - v_1 - v_2 - \cdots - v_K$ is the number of packets of mice.

**Proposition 10** (Mean Value of the Observables)**.** *If* $K$ *elephants are active in a time window of length* $\Delta$*, the mean number* $\mathbb{E}(W_j)$ *of flows sampled* $j$ *times,* $j \geq 1$*, satisfies the relation*

$$\left| \frac{\mathbb{E}(W_j)}{K} - \mathbb{Q}_j \right| \leq p\mathbb{E}\left(\frac{v^2}{V}\right), \quad (5.4)$$

*where* $\mathbb{Q}$ *is the probability distribution defined by*

$$\mathbb{P}(\mathbb{Q} = j) \stackrel{def}{=} \mathbb{Q}_j = \mathbb{E}\left(\frac{(pv)^j}{j!} e^{-pv}\right),$$

*and* $p = 1/k$ *is the sampling rate.*

From Equation (5.4) one gets that the larger the total volume $V$ of packets is, the better is the approximation of $\mathbb{E}(W_j)/K$ by $\mathbb{Q}_j$.

*Démonstration.* The number of times $\hat{v}_i$ that the $i$th flow is sampled in the time interval is given by

$$\hat{v}_i = B_1^i + B_2^i + \cdots + B_{pV}^i,$$

where, due to the mixing condition, $B_\ell^i$ is equal to one if the $\ell$th sampled packet is from the $i$th flow, which event occurs with probability $v_i/V$. Note that the total number of sampled packets is $pV$.

Conditionally on the values of the set $\mathcal{F} = \{v_1, \ldots, v_K\}$, the variables $(B_\ell^i, \ell \geq 1)$ are independent Bernoulli variables. For $1 \leq i \leq K$, Le Cam's Inequality (5.3) gives therefore the relation

$$\|\mathbb{P}(\hat{v}_i \in \cdot \mid \mathcal{F}) - Q_{pv_i}\|_{tv} \leq p\frac{v_i^2}{V}.$$

By integrating with respect to the variables $v_1, \ldots, v_K$, this gives the relation

$$\|\mathbb{P}(\hat{v}_i \in \cdot) - \mathbb{Q}\|_{tv} \leq p\mathbb{E}\left(\frac{v_i^2}{V}\right).$$

In particular, for $j \in \mathbb{N}$, $\left|\mathbb{P}(\hat{v}_i = j) - \mathbb{Q}_j\right| \leq p\mathbb{E}\left(v^2/V\right)$. Since

$$\mathbb{E}(W_j) = \sum_{i=1}^{K} \mathbb{P}(\hat{v}_i = j),$$

by summing on $i = 1, \ldots, K$, one gets

$$\left|\mathbb{E}(W_j) - K\mathbb{Q}_j\right| \leq pK\mathbb{E}\left(\frac{v^2}{V}\right)$$

and the result follows. □

If the number of packets per flow were constant, then $\mathbb{Q}$ would be a Poisson distribution with parameter $pv$, the variable $v$ being in this case a constant. The above inequality shows that at the first order the expected value of $W_j$ is $p\mathbb{E}(v)$. The expression of $\mathbb{Q}$, however, indicates that higher order moments of $v$ play a significant role. For example, if the variable $v$ has a significant variance, then the classical rough reduction, which consists of assuming that the size of a sampled elephant is $pv$, is no longer valid for estimating the original size of the elephant.

Under the negligibility condition, we deduce that

$$\left|\frac{\mathbb{E}(W_j)}{K} - \mathbb{Q}_j\right| \leq p\varepsilon,$$

where $\varepsilon$ appears in Definition 4 and is assumed to much less than 1. This implies that Inequality (5.4) is tight and the quantity $\mathbb{E}(W_j)/K$ can accurately be approximated by the quantity $\mathbb{Q}_j$, when no flows are dominating in traffic.

We are now ready to state the main result needed for estimating the number $K$ of elephants from sampled data.

**Proposition 11** (Asymptotic Mean Values)**.** *Under the same assumptions as those of Proposition 10,*

$$\lim_{K \to +\infty} \frac{\mathbb{E}(W_{j+1})}{\mathbb{E}(W_j)} \sim 1 - \frac{a+1}{j+1} \tag{5.5}$$

*and*

$$\lim_{K\to+\infty} \frac{\mathbb{E}(W_j)}{K} \sim a(pB_{min})^a \frac{\Gamma(j-a)}{j!}, \tag{5.6}$$

*if* $B_{max} >> 1$ *and* $pB_{min} << 1$, *where* $\Gamma$ *is the classical Gamma function defined by*

$$\Gamma(x) = \int_0^{+\infty} u^{x-1} e^{-u}\, du, \quad x > 0.$$

*Démonstration.* For $j \geq 1$,

$$\mathbb{Q}_j = \mathbb{E}\left(\frac{(pv)^j}{j!} e^{-pv}\right) \sim aB_{min}^a \frac{p^{a+1}}{j!} \int_{B_{min}}^{+\infty} (pu)^{j-a-1} e^{-pu}\, du$$

and then

$$\mathbb{Q}_j \sim aB_{min}^a \frac{p^a}{j!} \int_{pB_{min}}^{+\infty} u^{j-a-1} e^{-u}\, du \sim a(pB_{min})^a \frac{\Gamma(j-a)}{j!},$$

since $pB_{min} \sim 0$. Therefore, by using the relation $\Gamma(x+1) = x\Gamma(x)$ we obtain the equivalence

$$\frac{\mathbb{Q}_{j+1}}{\mathbb{Q}_j} \sim \frac{j-a}{j+1}.$$

The proposition follows by using the fact that the upper bound of Equation (5.4) of Proposition 10 goes to 0 by the law of large numbers. □

As it will be seen later in the next section, Relation (5.5) is used to estimate the exponent $a$ of the Pareto distribution of the number of packets of elephants, the quantities $\mathbb{E}(W_j)$ and $\mathbb{E}(W_{j+1})$ being easily derived from sampled traffic. The quantity $K$ will be estimated from Relation (5.6). The estimation of the parameter $B_{min}$ from sampled traffic as well as the correct choice of the integer $j$ will be discussed in the next section.

## 5.5 Applications

### 5.5.1 Traffic parameter inference algorithm

In this section, it is assumed that only sampled traffic is available. The methods described in Section 5.2 to infer the statistical properties of the flows cannot be applied and another algorithm has to be defined. For the experiments carried out in the present section, the sampling factor $p = 1/k$ has been taken equal to 1/100. To infer flow characteristics, we have to give the proper definition of the mouse and elephant dichotomy (the parameter $B_{min}$) and to estimate the coefficient of the corresponding Pareto distribution (the parameter $a$ in Relation (5.1)).

Relation (5.5) gives the following equivalence, for $j \geq 1$ sufficiently large so that the impact of mice on $\mathbb{E}(W_j)$ is negligible,

$$a \sim a(j) \stackrel{\text{def.}}{=} (j+1)\left(1 - \frac{\mathbb{E}(W_{j+1})}{\mathbb{E}(W_j)}\right) - 1, \tag{5.7}$$

and Relation (5.6) yields an estimate of the number of elephants, i.e. the number of flows with a number of packets greater than or equal to $B_{min}$ ; we specifically have

$$K \sim K(j) \stackrel{\text{def.}}{=} \frac{j!\,\mathbb{E}(W_j)}{a(j)(pB_{min})^{a(j)}\Gamma(j-a(j))}. \tag{5.8}$$

These estimations greatly depend on some of the key parameters used to obtain a convenient and confident Pareto representation of the size of the flows, in particular the size of the time window $\Delta$ and the lower bound $B_{min}$ for the elephants. The variable $\Delta$ is chosen so that

1. the number of flows sampled twice is sufficiently large in order to obtain a significant number of samples so that the estimation of the mean values of the random variables $W_j$ for $j \geq 2$ is accurate ; this requires that $\Delta$ should not be too small,
2. $\Delta$ is not too large in order to preserve the unimodal Pareto representation (see Section 4.4 for a discussion).

To count the average number of flows sampled $j$ times, the parameter $j$ should be chosen as large as possible in order to neglect the impact of mice (for which the Pareto representation does not hold) but not too large so that the statistics are robust to compute the mean value $\mathbb{E}(W_j)$.

In the experimental work reported below, special attention has been paid to the choice of the *universal* constants, i.e., those constants used in the analysis of sampled data, that do not depend on the traffic trace considered. In our opinion, this is a crucial in an accurate inference of traffic parameters from sampled data. These constants are defined in the algorithm given in Table 5.5.

TAB. 5.5 – Algorithm used to identify $\Delta$ and the Pareto parameter from sampled traffic.

— Choose $\Delta$ so that $80 \leq \mathbb{E}[W_2] \leq 100$ ;
— Choose $j$ so that $|a(j) - a(j+1)|$ computed with Equation (5.7) is minimized with for all $j$ such that $\mathbb{E}[W_j] \geq 5$.
— $B_{min}$ is the smallest integer so that the probability that a flow of size greater than $B_{min}$ is sampled more than $j$ times is greater than $p/10$ ;

### 5.5.2 Experimental results

Concerning the estimation of the constants $B_{min}$, the numerical results obtained by using the algorithm given in Table 5.5 are presented in Table 5.6, where the values of the different $B_{min}$ estimated by the algorithm are compared against the values given in Section 5.2. As it can be observed, the proposed algorithm yields a rather conservative definition of elephants (i.e., flows of size greater than or equal to $B_{min}$).

TAB. 5.6 – Elephants for the France Telecom ADSL and the Abilene traffic traces.

| | ADSL A | ADSL B Up | ADSL B Down | Abilene A | Abilene B |
|---|---|---|---|---|---|
| $B_{min}$ | 20 | 29 | 39 | 89 | 79 |
| estimated $B_{min}$ | 21 | 45 | 45 | 77 | 77 |

The main results are gathered in Table 5.7 giving the quantities $K$ and $a$ estimated by using Equations (5.7) and (5.8) for different values of the parameters $j$. These values are compared against the experimental values $a_{exp}$ and $K_{exp}$, referred to as the "real" $a$ and $K$ obtained from the complete traffic traces in Section 5.2. The accuracy of the estimation of $K$ is generally quite good except for the Abilene A trace where the error is significant although not out of bound. A look at the corresponding figure in Section 5.2 gives a plausible explanation for this discrepancy : For this trace, the Pareto representation is not very precise.

Finally, it is worth noting from Table 5.7 that the estimation of the important parameter $a$ describing the statistics of flows is also quite accurate. The error in this table is defined as

$$\frac{K(j) - K_{exp}}{K_{exp}}.$$

TAB. 5.7 – Estimations of the Number of Elephants from Sampled traffic

| Trace | $\Delta$ | $j$ | $\mathbb{E}(W_j)$ | $\mathbb{E}(W_{j+1})$ | $a_{exp}$ | $a(j)$ | $K_{exp}$ | $K(j)$ | Error |
|---|---|---|---|---|---|---|---|---|---|
| ADSL A | 5s | 3 | 12.89 | 3.33 | 1.85 | 1.95 | 943.71 | 1031.04 | 9.25% |
| ADSL B Do | 15s | 4 | 9.7 | 4.75 | 1.49 | 1.55 | 414.90 | 404.13 | 2.59% |
| ADSL B Up | 15s | 4 | 7.46 | 2.97 | 1.97 | 2.00 | 453.01 | 462.68 | 2.13% |
| ABILENE A | 1s | 5 | 6.04 | 3.21 | 1.38 | 1.81 | 217.44 | 270.79 | 24.53% |
| ABILENE B | 1s | 5 | 6.1 | 3.7 | 1.36 | 1.51 | 209.12 | 197.12 | 5.74% |

**Remark.** As pointed out by Loiseau *et al.* [55], the determination of $\Delta$ is crucial. Recall it is determined explicitly by the first step of our algorithm, see Table 5.5.

## 5.6 Conclusion

We have developed in this chapter one method of characterizing flows in IP traffic by a few parameters and another one of inferring these parameters from sampled data obtained via deterministic 1-out-of-$k$ sampling. For this purpose, we have made some restrictive assumptions, which are in our opinion essential in order to establish an accurate characterization of flows. The basic principle we have adopted consists of describing flows in successive observation windows of limited length, which has to satisfy two contradicting requirements. On the one hand, observation windows shall not to be too large in order to preserve a description of flow statistics as simple as possible, for instance their size by means of a simple Pareto distribution.

On the other hand, a sufficiently large number of packets has to be present in each observation window in order to be able of computing flow characteristics with sufficient accuracy, in particular the tail of the distribution of the flow size. By assuming that large flows (elephants) have a size which is Pareto distributed, we have developed an algorithm to determine the optimal observation window length together with the parameters of the Pareto distribution. The location parameter $B_{min}$ (see Equation (5.1)) leads to a natural division of the total flow population into two sets : those flows with at least $B_{min}$ packets, referred to as elephants, and those flows with less than $B_{min}$ packets,called mice. This method of characterizing flows has been tested against traffic traces from the France Telecom and Abilene networks carrying completely different types of traffic.

For interpreting sampled data, we have made assumptions on the sampling process. We have specifically supposed that flows are sufficiently interleaved in order to introduce some randomness in the packet selection process (mixing condition) and that there are no dominating flows so that there is no bias with regard to the probability of sampling a flow (negligibility condition). These two assumptions allows us to establish rigorous results for the number of times an elephant is sampled, in particular for the mean values of the random variables $W_j$, $j \geq 1$.

Of course, when analyzing sampled data, the original flow statistics are not known. In particular, the length of the observation window necessary to characterize the flow size by means

of a unique Pareto distribution is unknown. To overcome this problem, we have proposed an algorithm to fix the observation window length and the minimal length of elephants. Then, by choosing the index $j$ sufficiently large so as to neglect the impact of mice, the theoretical results are used to complete the flow parameter inference. This method has been tested against Abilene and France Telecom traffic traces and yields satisfactory results.

# 6

# Inference of flow statistics via packet sampling in the Internet

Contents



We show in this chapter that by deterministic or probabilistic packet sampling, the tail of the distribution of the sampled flow size can be obtained by rescaling that of the original flow size. To recover information on the flow size distribution lost through packet sampling, we propose some heuristics based on measurements from different backbone IP networks.

## 6.1 Introduction

The basic problem of packet sampling is that it is difficult to infer the original flow statistics from sampled data. But this task is however fundamental for charging, monitoring and traffic characterization in operational networks.

Flow statistics inference from sampled data has been addressed in previous studies. Duffield *et al* [29, 28] study the accuracy of different estimators based on multiplying the sampled flow size by the sampling factor $k$, but their method does not apply to the complete range of the flow size. Hohn and Veitch [48] use generating function techniques to invert the flow size distribution but the proposed procedure is numerically unstable. Mori *et al* [64] use a Bayesian approach to inferring the characteristics of long flows.

In this chapter, we develop a probabilistic approach to inverting sampled traffic together with some heuristic arguments. First, we note that when observing sampled traffic, we can only compute the distribution of the random variable $\tilde{\nu}$ describing the number of packets in sampled flows and $K_s$ the number of sampled flows.

Under some reasonable assumptions on the sampling process, we show in this chapter that the tail of the original flow size distribution can be obtained by rescaling that of the sampled flow size. It is however much more difficult to totally recover the original distribution because information on small or moderate flow sizes is lost through sampling. To overcome this problem, we propose some heuristic arguments based on measurements and exploiting an a priori information on flows. We consider, as in previous chapters, TCP traffic only.

The rest of this note is organized as follows : In Section 6.2, we make some reasonable assumptions on the sampling process. In Section 6.3, we prove that the tail of the original flow size can be obtained by rescaling that of the sampled flow size. In Section 6.4, we present some heuristic arguments to recover the total flow size distribution. Concluding remarks are presented in Section 6.5.

## 6.2 Assumptions on the sampling process

When observing in a time window of length $\Delta$ traffic on a high speed link, one may reasonably assume that the packets of the different active flows are sufficiently interleaved. Hence, one may suppose the selection of packets among active flows at a sampling time is random.

Moreover, in a time window of length $\Delta$, flows start and finish and some of them may be silent (for instance in the case of flows alternating between On and Off periods). In [18, Section 3.3], it is shown that these fluctuations may be neglected at the first order (i.e., when computing mean values) and it can be assumed that flows are permanent. Under the two above assumptions, we then suppose that the probability of selecting a packet of a given flow, say, flow $i$, is equal to $\nu_i/V_i$, where $\nu_i$ is the size of flow $i$ and $V_i$ is the total number of packets arrived when flow $i$ is active.

## 6.3 Tail of the sampled flow size

Let $W_j$ denote the number of flows sampled $j$ times.

**Proposition 12.** *If $K$ flows are active during a time window of length $\Delta$, the mean value $\mathbb{E}(W_j)$*

*satisfies*

$$\left|\mathbb{E}(W_j) - K\mathbb{Q}_j\right| \leq p\sum_{i=1}^{K}\mathbb{E}\left(v_i^2/V_i\right), \tag{6.1}$$

*where* $p = 1/k$, $v_i$ *is the random number of packets in the* $i$*th flow, and* $\mathbb{Q}$ *is the probability distribution defined for* $j \geq 0$ *by*

$$\mathbb{Q}_j = \mathbb{E}\left(\frac{(pv)^j}{j!}e^{-pv}\right). \tag{6.2}$$

*Démonstration.* Let us condition on the values of the set $\mathcal{F} = \{v_1, \ldots, v_K, V_1, \ldots, V_K\}$. Under the assumptions of Section 6.2, the number of times that the $i$th flow is sampled is

$$\tilde{v}_i = B_1^i + B_2^i + \cdots + B_{pV_i}^i,$$

where $B_\ell^i$ is equal to one if the $\ell$th sampled packet is from the $i$th flow, which event occurs with probability $v_i/V_i$. The random variables $(B_\ell^i, \ell \geq 1)$ are i.i.d. Bernoulli random variables and Le Cam's Inequality [11] then states

$$\left\|\mathbb{P}(\tilde{v}_i \in \cdot) - \mathbb{P}\left(Q_{\mathbb{E}(\tilde{v}_i)} \in \cdot\right)\right\|_{tv} \leq \sum_{\ell=1}^{pV_i}\mathbb{P}(B_\ell^i = 1)^2,$$

where $\|.\|_{tv}$ is the total variation norm and $Q_{\mathbb{E}(\tilde{v}_i)}$ is a Poisson random variable with mean $\mathbb{E}(\tilde{v}_i)$. By deconditioning with respect to the set $\mathcal{F}$, we have by using the distribution $\mathbb{Q}$

$$\left\|\mathbb{P}(\tilde{v}_i \in \cdot) - \mathbb{Q}\right\|_{tv} \leq p\mathbb{E}\left(v_i^2/V_i\right). \tag{6.3}$$

In particular, for $j \in \mathbb{N}$, $\left|\mathbb{P}(\tilde{v}_i = j) - \mathbb{Q}_j\right| \leq p\mathbb{E}\left(v_i^2/V_i\right)$. Since $\mathbb{E}(W_j) = \sum_{i=1}^{K}\mathbb{P}(\tilde{v}_i = j)$, summing on $i$ yields Equation (6.1). □

If $K$ is sufficiently large, from Equation (6.1), we have for $j \geq 1$

$$\mathbb{P}(\tilde{v} = j) = \mathbb{E}\left(\frac{1}{K_s}W_j\right) \sim \frac{1}{\nu}\mathbb{Q}_j, \tag{6.4}$$

where $\nu = K_s/K$ is the probability of sampling a flow.

**Proposition 13.** *If all flows have a negligible contribution to the total volume of traffic (i.e.,* $\mathbb{E}(v_i^2/V_i) \ll 1$ *for all* $i = 1, \ldots, K$*), if* $K$ *is sufficiently large, and if the flow size distribution satisfies the following conditions : There exists some* $0 < \alpha < 1/2$ *and* $\gamma > 0$ *such that, for any* $T > 0$*,*

$$\lim_{x\to+\infty}\frac{\mathbb{P}(v \geq x)}{\mathbb{P}(v \geq x + Tx^{1/2+\gamma})} = 1 \text{ and } \lim_{x\to+\infty}\frac{1}{\mathbb{P}(v \geq x)}e^{-\gamma x^\alpha} = 0.$$

*then, when* $j \to \infty$*,*

$$\mathbb{P}(\tilde{v} \geq j) \sim \mathbb{P}\left(v \geq \frac{j}{p}\right)/\nu. \tag{6.5}$$

This assumption on the flow size distribution is true for any Pareto and for some Weibull distributions.

*Démonstration.* From Equation (6.4)

$$\nu\mathbb{P}(\tilde{v} \geq j) \sim A_j = \mathbb{P}\left(N[0, p\nu] \geq j\right), \tag{6.6}$$

where $N$ is a Poisson process with parameter 1. We denote by

$$V = p\nu \text{ and } \eta(x) = \frac{N[0,x] - x}{\sqrt{x}}.$$

We can prove that there exists some $C_0 < +\infty$ such that for $a > 0$, and for all $x \geq 1$,

$$\mathbb{P}(|\eta(x)| \geq a) \leq C_0 \exp(-\delta a).$$

For $\alpha > 0$,

$$A_j = \mathbb{P}\left(V + \eta(V)\sqrt{V} - j \geq 0, |\eta(V)| \leq j^\alpha\right) + O(\mathbb{P}(|\eta(V)| \geq j^\alpha)).$$

Since $\mathbb{P}(|\eta(V)| \geq j^\alpha)/\mathbb{P}(V \geq j) = o(1)$ by assumption, it is enough to prove the equivalence (6.6), replacing $A_j$ by the following term

$$B_j = \mathbb{P}\left(\sqrt{V} \geq \frac{-\eta(V) + \sqrt{\eta(V)^2 + 4j}}{2}, |\eta(V)| \leq j^\alpha\right) = \mathbb{P}\left(V \geq j + F(\eta(V), j), |\eta(V)| \leq j^\alpha\right)$$

where

$$F(u,x) = \left(\frac{-u + \sqrt{u^2 + 4x}}{2}\right)^2 - x.$$

There exist $0 < \alpha < 1/2$ and $T > 0$ such that, for $u < x^\alpha$, one has $F(u,x) \leq T x^{\alpha + 1/2}$. It is then easy to conclude. □

## 6.4 Heuristics for the total flow size distribution

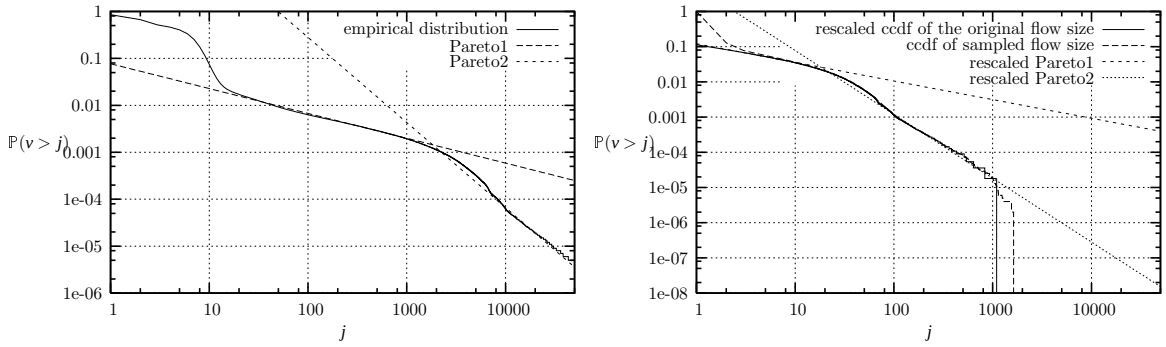
Proposition 13 shows that the tail of the complementary cumulative distribution function (ccdf) of the original flow size can be obtained by rescaling that of the sampled flow size. We can however verify through examples that information on that distribution for small or moderate flow size values is lost.

We exemplify this phenomenon by considering a 2 hour long real traffic trace from a 1 Gbit/s transmission link of the France Telecom IP backbone network carrying ADSL traffic. The original flow size is depicted in Figure 6.28(a) and the deterministically sampled flow size in Figure 6.28(b), which exhibits good agreement with the rescaled distribution $\mathbb{P}(v \geq j/p)/\nu$ for sufficiently large $j$ as predicted by Proposition 13. But all information for moderate values of the flow size is contained in a few values, in this case $\mathbb{P}(\tilde{v} \geq j)$ for $j = 2, 3$. The same phenomenon (see [19]) has been observed for an Abilene traffic trace available at
http ://pma.nlanr.net/Traces/Traces/long/ipls/3/.

In fact, through numerous experiments with real traffic traces, it has been observed in [19] that $\mathbb{P}(v \geq j/p)$ can be approximated by $\nu\mathbb{P}(\tilde{v} \geq j)$ when $j \geq j_0$ for some $j_0 > 0$. The problem is then to estimate the quantities $\mathbb{P}(v = j)$ for $j = 1, \ldots, j_0/p - 1$.

We have from Equation (6.4)

$$\mathbb{P}(\tilde{v} = j) \sim \frac{1}{\nu} \sum_{\ell=1}^{\infty} \frac{(p\ell)^j}{j!} e^{-p\ell} \mathbb{P}(v = \ell) \tag{6.7}$$




(a) Original size.

(b) Sampled size ($p = 1/100$).

FIG. 28 – Flow size distribution in the France Telecom ADSL trace.

and we know by Proposition 13 that for $j \geq j_0$, this equation is equivalent to $\mathbb{P}(\tilde{v} = j) \sim \mathbb{P}(v = j/p)/(vp)$. It follows that for determining the $(j_0/p - 1)$ quantities $\mathbb{P}(v = \ell)$ for $\ell = 1, \ldots, j_0/p - 1$, we have only $j_0$ equations. The problem is hence clearly under-determined. Some heuristics are needed to recover the complete flow size distribution.

It has been observed in [19] that depending on the size of the observation window $\Delta$, the sampled flow size distribution can locally be approximated by means of Pareto distributions. This leads us to make the following assumption.

**Assumption 1.** *There exist some $m > 0$ and some integers $j_0 < j_1 < \ldots < j_m = \infty$ such that for $\ell = 1, \ldots, m$ and $j \in [j_{\ell-1}, j_\ell]$, $\tilde{v}$ has a Pareto distribution of the form*

$$\mathbb{P}(\tilde{v} \geq j) = \mathbb{P}(\tilde{v} \geq j_{\ell-1}) \, (j_{\ell-1}/j)^{a_\ell}$$

*for some shape parameter $a_\ell > 0$.*

When $\Delta$ is adequately chosen, the tail may be uni-modular (i.e., $m = 1$), but when $\Delta$ is too large, we can have $m > 1$. For the above France Telecom trace ($\Delta = 2$ hours), $m = 2$ as shown in Figure 6.28(b).

By using Proposition 13, we deduce that for $\frac{j_{m-1}}{p} \leq j \leq \frac{j_m}{p}$

$$\mathbb{P}(v \geq j) \sim v\mathbb{P}\left(\tilde{v} \geq j_{m-1}\right) \left(j_{m-1}/(pj)\right)^{a_m}, \tag{6.8}$$

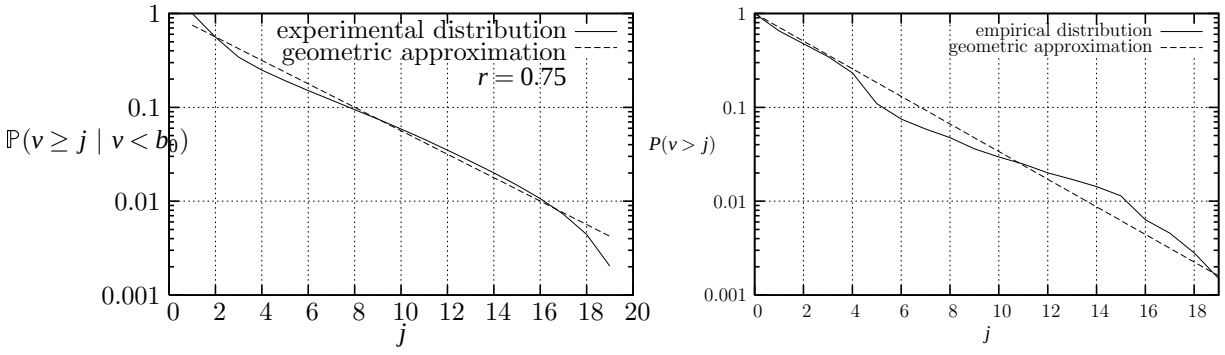
The above equation implies that $\mathbb{P}(v \geq j)$ can locally be approximated by a Pareto distribution with shape parameter $a_m$, as shown in Figure 6.28(a).

For inferring the quantities $\mathbb{P}(v = j)$ for $j = 1, \ldots, j_0/p - 1$, we need more assumptions. Numerous experiments [19] have shown that when $j < b_0$ for some $b_0 > 0$, $\mathbb{P}(v = j)$ follows a geometric distribution.

**Assumption 2.** *There exists some $b_0 > 0$ such that for $1 \leq j < b_0$, $\mathbb{P}(v = j) = (1-r)r^{j-1}/(1-r^{b_0})$ for some $r > 0$.*

The above assumption is supported by experiments, as shown in Figure 6.29(a) and 6.29(b) for the France Telecom and Abilene traffic traces, respectively. The value $b_0 = 20$ has been successfully tested in numerous experiments.

By using Equation (6.8) and Assumption 2, we have the form of the distribution for $j \leq b_0$ and $j \geq j_0/p$. To fill the gap, we use the following heuristic : $\mathbb{P}(v \geq j)$ for $b_0 \leq j \leq j_1/p$ has the




(a) ADSL trace (b) Abilene trace

FIG. 29 – Ccdf of the number of packets in flows with less than $b_0 = 20$ packets.

same form as in Equation (6.8), namely $\mathbb{P}(v \geq j) = \mathbb{P}(v \geq b_0)(b_0/j)^{a_1}$. Equation (6.7) can then be rewritten as

$$\mathbb{P}(\tilde{v} = j) \sim \frac{\mathbb{P}(v < b_0)}{\nu} \sum_{\ell=1}^{\infty} (1 - r) r^{\ell} \frac{(p\ell)^j}{j!} e^{-p\ell} + \frac{1}{\nu} \sum_{\ell=b_0}^{\infty} \frac{(p\ell)^j}{j!} e^{-p\ell} \mathbb{P}(v = \ell). \quad (6.9)$$

The shape parameters $a_\ell$ for $1 \leq \ell \leq m$ are determined from the sampled flow size distribution by using standard Maximum Likelihood Estimation (MLE) procedures. The parameter $b_0$ is set equal to 20 ; this value is purely phenomenological but roughly corresponds to the number of packets needed to leave the slow start regime with a maximum window size of 32 Kbytes. The parameter $\mathbb{P}(v \geq b_0)/\nu$ is obtained by using Proposition 13, namely by computing the ratio $\eta \stackrel{def}{=} \mathbb{P}(\tilde{v} \geq j)/(b_0 p/j)^{a_1}$ for $j \in \{j_0, \ldots, j_1\}$, which is by assumption independent of $j$. The number of flows with at least $b_0$ packets is $K_0^+ = \eta K_s$. Equation (6.9) multiplied by $K_s$ for $j = 1, 2$ is then used to compute the parameter $r$ and the number $K_0^-$ of flows with less than $b_0$ packets. The total number of flows is then $K = K_0^+ + K_0^-$ and the probability of sampling a flow is estimated by the ratio $K_s/K$.

By using the above method for the France Telecom ADSL trace with $p = 1/100$, we find $j_0 = 3$ and the estimated shape parameters $\hat{a}_1 = .54$ and $\hat{a}_2 = 1.81$, which are close to the experimental values $a_1 = .52$ and $a_2 = 1.81$ for the original flow size. We then find $\mathbb{P}(v \geq b_0)/\nu = .3$ and since $K_s = 1,120,546$, we obtain the estimate $\hat{K}_0^+ = 336,163$, while the actual value is $K_0^+ = 343,004$. By neglecting the term due to flows with at least 20 packets in Equation (6.9), we then find the estimate $\hat{r} = 0.84$ while the actual experimental value is $r = .75$. This yields a number of flows with less than $b_0$ packets $\hat{K}_0^- \sim 20.1e6$ while the actual value is $K_0^- \approx 19.8e6$. Finally, the estimated total number of flows is $\hat{K} = 20.4e6$ while the actual value is $K = 20.1e6$ and we find the estimate $\hat{\nu} = .054$ for the probability of sampling a flow while the experimental value is $\nu = 0.057$.

## 6.5 Conclusion

The method of inverting sampled traffic presented in this note is based on a prior knowledge of the form of the flow size distribution (geometric distribution of small flows and Pareto for the others). The method has successfully been tested on sampled ADSL traffic from the France

Telecom collect IP network. The method is robust as long as the sampling factor $p$ is not too small so that the Pareto distributions can be recognized in the sampled flow size distribution.

# Conclusion

Dans cette thèse, nous avons étudié la composition du trafic Internet à l'échelle des flots. Un flot est un ensemble de paquets IP ayant certains champs communs. L'analyse des flots fournit des informations utiles sur les clients comme le nombre de clients actifs, leur débits, leurs comportements... Aujourd'hui, l'accès à de telles informations en temps réel est une tâche difficile vu le très haut débit du trafic et la masse gigantesque de données à analyser. Pour surmonter ce problème, deux axes ont été explorés : le hachage des flots utilisant les filtres de Bloom, et l'échantillonnage du trafic.

Dans le premier chapitre, nous nous sommes intéressés à l'étude d'un algorithme d'identification et de comptage des grands flots basé sur les filtres de Bloom. L'idée-clé de l'algorithme est de rafraîchir les filtres avec une fréquence qui dépend du débit instantané du trafic, de telle façon à garantir l'efficacité du comptage. Le but de notre étude analytique est d'estimer l'erreur générée par cet algorithme sur le comptage des grands flots (les éléphants). Nous nous sommes particulièrement intéressés aux faux positifs, c'est-à-dire aux petits flots qui sont considérés à tort comme des éléphants par l'algorithme. Partant d'un modèle simplifié où le trafic est uniquement composé de flots de taille 1 paquet, nous avons montré que le problème peut être décrit à l'aide d'une file d'attente M/G/1/C, ce qui permet d'exprimer le nombre de faux positifs en fonction des différents paramètres de l'algorithme. L'analyse a été ensuite généralisée à des souris de taille quelconque.

Dans le deuxième chapitre, nous avons proposé et expérimenté d'autres versions de l'algorithme en jouant sur le critère de rafraîchissement des filtres. Le but est d'éviter la saturation des filtres en cas de trafic intense. Pour décrire l'état des filtres, deux paramètres ont été comparés : le taux de remplissage (proportion de compteurs non nuls) et la valeur moyenne des compteurs. Les expérimentations montrent que le deuxième critère est particulièrement intéressant dans le cas de trafic avec des souris de grandes tailles. L'algorithme a été ensuite adapté au problème de détection des attaques par déni de service où l'attaque peut se traduire au niveau du trafic par l'apparition d'un grand flot (en utilisant une définition adaptée du flot). Le rafraîchissement des filtres est plus agressif dans ce cas, car le but est d'éliminer au plus vite tous les flots légitimes et ne garder dans les filtres que les flots suspicieux. Les expérimentations montrent que l'algorithme ainsi conçu permet de détecter la majorité des attaques avec un temps de réponse assez court (de l'ordre de la minute).

Le troisième chapitre traite une autre composante de l'algorithme, à savoir la méthode de remplissage des filtres et d'incrémentation des compteurs. Pour atténuer les collisions entre les flots, l'algorithme utilise l'incrémentation conservative dont le principe est le suivant : un paquet est associé à $k$ compteurs, et seulement les compteurs ayant la plus petite valeur sont incrémentés de 1. C'est ce qu'on appelle dans ce manuscrit "la règle du min". Pour des raisons de

simplification, l'analyse présentée dans le premier chapitre ne tient pas compte de cette règle de min. L'objectif du troisième chapitre est d'étudier analytiquement l'impact de la règle du min sur les performances de l'algorithme. Pour pouvoir modéliser le problème, nous avons introduit pour l'algorithme la modification suivante : on n'incrémente pas tous les compteurs réalisant le minimum, mais seulement l'un de ces compteurs choisi au hasard. On retrouve ainsi le modèle du supermarché où un client choisi la file la plus courte parmi $k$ files sélectionnées au hasard parmi $m$. En cas d'égalité, une file est choisie au hasard parmi les files les plus courtes. Ce modèle étant connu dans la littérature, il nous a permis d'aboutir à des résultats théoriques en rapport avec les performances de l'algorithme.

La deuxième partie de la thèse est consacrée à l'étude de l'échantillonnage et des méthodes d'inférence des paramètres du trafic réel à partir des informations réduites contenues dans le trafic échantillonné. Dans le chapitre quatre, nous avons comparé les deux méthodes d'échantillonnage suivantes : échantillonnage déterministe et probabiliste. Nous avons montré que si les flots sont suffisamment mélangés, ces deux méthodes sont équivalentes du point de vue composition du trafic échantillonné. Nous avons aussi montré que la queue de distribution de la taille des flots peut être facilement inférée par échantillonnage à condition qu'elle soit lourde.

Dans les deux derniers chapitres, nous avons montré que la distribution de la taille des flots sur une durée convenablement choisie, peut être approchée par une loi de Pareto. Nous avons établi une méthode permettant de trouver les paramètres de cette loi pour une trace de trafic quelconque. Cette méthode ne nécessite aucune information à priori sur la trace considérée. Nous avons ensuite montré qu'on peut aussi inférer les paramètres de la loi de Pareto ainsi que le nombre total de flots en examinant juste le trafic échantillonné. Ces méthodes ont été testées et validées sur des traces de trafic issues du réseau de France Télécom et du réseau Abilène.

# Bibliographie


[1] http ://www.ietf.org/html.charters/psamp-charter.html.

[2] P. Abry and D. Veitch, *Wavelet analysis of long range dependent traffic*, IEEE Trans. Information Theory **44** (1998), no. 1, 2–15.

[3] N. Antunes, Y. Chabchoub, C. Fricker, F. Guillemin, and P. Robert, *A new method of inferring flow statistics from sampled data in the Internet*, Submitted for publication.

[4] N. Antunes, C. Fricker, P. Robert, and D. Tibi, *Stochastic networks with multiple stable points*, Annals of Probability **36** (2008), no. 1, 255–278.

[5] Y. Azar, A.Z. Broder, and .R. Karlin, *On-line load balancing*, Theoret. Comput. Sci. **130** (1994), no. 1, 73–84. MR MR1287132 (95f :68026)

[6] Y. Azzana, *Mesures de la topologie et du trafic internet*, Ph.D. thesis, Université Pierre et Marie Curie, july 2006.

[7] N. Ben Azzouna, *Etude des méthodes d'échantillonnage des flux pour la mesure dans les réseaux large bande*, Ph.D. thesis, Université Pierre et Marie Curie - Paris VI, Septembre 2004.

[8] N. Ben Azzouna, F. Clérot, C. Fricker, and F. Guillemin, *A flow-based approach to modeling ADSL traffic on an IP backbone link*, Annals of Telecommunications **59** (2004), no. 11-12, 1260–1299.

[9] N. Ben Azzouna, F. Guillemin, S. Poisson, P. Robert, C. Fricker, and N. Antunes, *Inverting sampled ADSL traffic*, Proc. ICC 2005 (Seoul, Korea), May 2005.

[10] C. Barakat, P. Thiran, G. Iannaccone, C. Diot, and P. Owezarski, *A flow-based model for internet backbone traffic*, Proc. ACM SIGCOMM Internet Measurement Workshop (Marseille), November 2002.

[11] A. D. Barbour, Lars Holst, and Svante Janson, *Poisson approximation*, The Clarendon Press Oxford University Press, New York, 1992, Oxford Science Publications.

[12] P. Barford, J. Kline, D. Plonka, and A. Ron, *A signal analysis of network traffic anomalies*, ACM/SIGCOMM IMW, 2002.

[13] B. Benmammar, C. Lévy-Leduc, and F. Roueff, *Algorithme de détection des attaques de type syn flooding*, 20ème colloque GRETSI, Septembre 2007.

[14] P. Billingsley, *Convergence of probability measures*, second ed., Wiley Series in Probability and Statistics : Probability and Statistics, John Wiley & Sons Inc., New York, 1999, A Wiley-Interscience Publication. MR MR1700749 (2000e :60008)

[15] B.H. Bloom, *Space/time trade-offs in hash coding with allowable errors*, CACM **13** (1970), 422–426.

[16] A. Broder and M. Mitzenmacher, *Network applications of bloom filters : A survey*, Internet Mathematics **1** (2004), no. 4, 485–509.

[17] Y. Chabchoub, C. Fricker, F. Guillemin, and P. Robert, *Bounds for packet sampling in the Internet*, In Preparation.

[18] ———, *Deterministic versus probabilistic packet sampling in the Internet*, Proceedings of ITC'20, June 2007.

[19] Y. Chabchoub, C. Fricker, F. Meunier, and D. Tibi, *Analysis of an algorithm catching elephants on the internet*, Fifth Colloquium on Mathematics and Computer Science, DMTCS Proceedings Series, september 2008, pp. 299–314.

[20] Y. Chabchoub, C. Fricker, and H. Mohamed, *Analysis of a Bloom filter algorithm via the supermarket model*, Proceedings of itc21, september 2009.

[21] F. Chatelain, P. Borgnat, J.Y. Tourneret, and P. Abry, *Parameter estimation for sums of correlated gamma random variables. Application to anomaly detection in internet traffic*, Proc IEEE Int. Conf. on Acoust., Speech and Signal Proc. ICASSP-08, (Las Vegas (NV)), 2008.

[22] B.Y. Choi, J. Park, and Z.L. Zhang, *Adaptative random sampling for load change detection*, SIGMETRICS '02 : Proceedings of the 2002 ACM SIGMETRICS international conference on mesurement and modeling of computer systems (2002), 272–273.

[23] ———, *Adaptive packet sampling for accurate and scalable flow measurement*, Proc. Globecom'04 (Dallas, TX), December 2004.

[24] CISCO, http ://www.cisco.com/warp/public/netflow/index.html.

[25] M. Crovella and A. Bestravos, *Self-similarity in world wide web. Evidence and possible causes*, IEEE/ACM Trans. on Networking (1997), 835–846.

[26] E. D. Demaine, A. López-Ortiz, and J. I. Munro, *Frequency estimation, of internet packet streams with limited space*, Proceedings of the 10th Annual European Symposium on Algorithms (ESA 2002) (Rome, Italy), September 2002, pp. 348–360.

[27] P. Deuflhard and A. Hohmann, *Numerical analysis in modern scientific computing : an introduction*, Texts in Applied Mathematics, Springer, 2003.

[28] N. Duffield, C.L., and M. Thorup, *Estimating flow distributions from sampled flow statistics*, SIGCOMM'03, vol. August, 2003, pp. 25–29.

[29] N. Duffield, C.L., and Mikkel Thorup, *Properties and prediction of flow statistics*, ACM SIGCOMM Internet Measurement Workshop, November 2002, pp. 6–8.

[30] V. Dumas, F. Guillemin, and P. Robert, *A Markovian analysis of additive-increase multiplicative-decrease algorithms*, Adv. in Appl. Probab. **34** (2002), no. 1, 85–111. MR 1 895 332

[31] M. Durand and P. Flajolet, *Loglog counting of large cardinalities*, 2003.

[32] H.Wang D.Zhang and K.Shin, *Detecting syn flooding attacks*, IEEE Infocom'02 (2002).

[33] C. Estan, K. Keys, D. Moore, and G. Varghese, *Building a better NetFlow*, Proc. ACM Sigcomm'04 (Portland, Oregon, USA), August 30 – September 3 2004.

[34] C. Estan and G. Varghese, *New directions in traffic measurement and accounting*, Proceedings of the ACM SIGCOMM 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM-02) (New York) (John Wroclawski, ed.), Computer Communication Review, vol. 32, 4, ACM Press, August 19–23 2002, pp. 323–338.

[35] ———, *Bitmap algorithms for counting active flows on high speed links*, IMC '03 : Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement (2003), 153–166.

[36] ———, *New directions in traffic measurement and accounting : Focusing on the elephants, ignoring the mice*, ACM Trans. Comput. Syst **21** (2003), no. 3, 270–313.

[37] A. Feldmann, A. C. Gilbert, W. Willinger, and T. G. Kurtz, *The changing nature of network traffic : scaling phenomena*, SIGCOMM Comput. Commun. Rev. **28** (1998), no. 2, 5–29.

[38] A. Feldmann, J. Rexford, and R. Cáceres, *Efficient policies for carrying web traffic over flow-switched networks*, IEEE/ACM Trans. Netw. **6** (1998), no. 6, 673–685.

[39] W. Feller, *An introduction to probability theory and its applications*, John Wiley and Sons, 1996.

[40] P. Flajolet, *On adaptative sampling*, Computing (1990), 391–400.

[41] P. Flajolet, E. Fusy, O. Gandouet, and F. Meunier, *Hyperloglog : the analysis of a near-optimal cardinality estimation algorithm*, Proceedings of the 13th conference on analysis of algorithm (AofA 07) (Juan-les-Pins, France), 2007, pp. 127–146.

[42] P. Flajolet and G.N. Martin, *Probabilistic counting algorithms for data base applications*, Journal of Computer and System Sciences 31 **2** (1985), 182–209.

[43] O. Gandouet and A. Jean-Marie, *Loglog counting for the estimation of ip traffic*, Proceedings of the 4th Colloquium on Mathematics and Computer Science Algorithms, Trees, Combinatorics and Probabilities (Nancy, France), 2006.

[44] F. Giroire, *Order statistics and estimating cardinalities of massive data sets*, Discrete Applied Mathematics (to appear).

[45] F. Giroire and E. Fusy, *Estimating the number of active flows in a data stream over a sliding window*, Proceedings of the Fourth Workshop on Analytic Algorithmics and Combinatorics (ANALC0) (New Orleans) (David Applegate, ed.), SIAM, January 2007, pp. 223–231.

[46] W. Gong, Y. Liu, V. Misra, and D. Towsley, *On the tails of web file size distributions*, Proceedings of 39th Allerton Conference on Communication, Control, and Computing, 2001.

[47] C. Graham, *Chaoticity results for "join the shortest queue"*, Council for African American Researchers in the Mathematical Sciences, Vol. III (Baltimore, MD, 1997/Ann Arbor, MI, 1999), Contemp. Math., vol. 275, Amer. Math. Soc., Providence, RI, 2001, pp. 53–68. MR MR1827335 (2002f :60176)

[48] N. Hohn and D. Veitch, *Inverting sampled traffic*, Internet Measurement Conference, October 2003, pp. 27–29.

[49] ———, *Inverting sampled traffic*, IEEE/ACM Trans. Netw. **14** (2006), no. 1, 68–80.

[50] IETF, IPFIX Working Group, *IP flow information export*, For information, see the url http ://www.ietf.org/html.charters/ipfix-charter.html.

[51] IETF, PSAMP Working Group, *Packet sampling working group*, See the url https ://ops.ietf.org/lists/psamp.

[52] A.Hussain J.Heidmann and C.Papadopoulos, *A framework for classifying denial of service attacks*, ACM/SIGCOMM (Aout 2003).

[53] M.M. Krunz and A.M. Makowski, *Modeling video traffic using m/g/∞ input processes : acompromise between markovian and lrd models*, IEEE Journal on Selected Areas in Communications **16** (1998), no. 5, 733–748.

[54] A. Lakhina, M. Crovella, and C. Diot., *Detecting distributed attacks using network-wide flow data*, Proc. FloCon 2005 Analysis Workshop (New Orleans), September 2005.

[55] P. Loiseau, P. Gonçalves, S. Girard, and F. Forbesand P. Primet Vicat-Blanc, *Maximum likelihood estimation of the flow size distribution tail index from sampled packet data*, ACM Sigmetrics Conference (Seattle, WA, USA), June 2009.

[56] P. Loiseau, P. Gonçalves, and P. Primet Vicat-Blanc, *A comparative study of different heavy tail index estimators of the flow size from sampled data*, GridNets '07 : Proceedings of the first international conference on Networks for grid applications (ICST, Brussels, Belgium, Belgium), ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007, pp. 1–8.

[57] M. J. Luczak and C. McDiarmid, *On the power of two choices : balls and bins in continuous time*, Ann. Appl. Probab. **15** (2005), no. 3, 1733–1764. MR MR2152243 (2006j :60014)

[58] ———, *On the maximum queue length in the supermarket model*, Ann. Probab. **34** (2006), no. 2, 493–527. MR MR2223949 (2007g :60110)

[59] N. Duffield C. Lund and M. Thorup, *Charging from sampled network usage*, IMW '01 : Proceedings of the 1st ACM SIGCOMM Workshop on Internet Mesurement (2001), 245–256.

[60] G. S. Manku and R. Motwani, *Approximate frequency counts over data streams*, Proceedings of the 28th VLDB Conference (Hong Kong, China), 2002, pp. 346–357.

[61] A.Lakhina M.Crovella and C.Diot, *Characterization of netwok-wide anomalies in traffic flows*, Proceedings of the ACM/SIGCOMM Internet Measurement Conference. (2004), 201–206.

[62] M. Mitzenmacher, *The power of two choices in randomized load balancing*, Ph.D. thesis, Berkeley, 1996.

[63] ———, *Dynamic models for file sizes and double pareto distributions*, Internet Mathematics **1** (2002), no. 3, 301–33.

[64] T. Mori, M. Uchida, and R. Kawahara, *Identifying elephant flows through periodically sampled packets*, Internet Measurement Conference (Taormina, Italy), 2004.

[65] K. Papagiannaki, N. Taft, S. Bhattacharrya, P. Thiran, K. Salamatian, and Christophe Diot, *On the feasibility of identifying elephants in internet backbone traffic*, Sprint ATL Research Report RR01-ATL-110918, Sprint ATL, November 2001.

[66] K. Papagiannaki, N. Taft, S. Bhattacharyya, P. Thiran, K. Salamatian, and C. Diot, *A pragmatic definition of elephants in internet backbone traffic*, Internet Measurement Workshop, ACM, 2002, pp. 175–176.

[67] V. Paxson and S. Floyd, *Wide area traffic : The failure of the Poisson assumption*, IEEE/ACM Trans. on Networking (1995), 226–244.

[68] Philippe R., *Stochastic networks and queues*, Applications of Mathematics, vol. 52, Springer-Verlag, Berlin, 2003, Stochastic Modelling and Applied Probability. MR 1 996 883

[69] C. Rolland, J. Ridoux, and B. Baynat, *Using LitGen, a realistic IP traffic model, to evaluate the impact of burstiness on performance*, Proc. SIMUTools 2008 (Marseille, France), 2008.

[70] Y.Chen B.Krishnamurthy S.Sen and Y.Zhang, *Sketch-based change detection : Methods, evaluation, and applications*, Imconf (2003).

[71] C.Cheng T.Kung and K.Tan, *Use of spectral analysis in defense against dos attaks*, Proceedings of IEEE Globecom (2002).

[72] N. D. Vvedenskaya, R. L. Dobrushin, and F. I. Karpelevich, *A queueing system with a choice of the shorter of two queues—an asymptotic approach*, Problemy Peredachi Informatsii **32** (1996), no. 1, 20–34.

[73] N. D. Vvedenskaya and Y. M. Sukhov, *Dobrushin's mean-field approximation for a queue with dynamic routing*, Tech. Report 3328, INRIA, dec 1997.

[74] K.Y Whang, B.T. V.Z., and H.M. Taylor, *A linear-time probabilistic counting algorithm for database applications*, ACM Transactions on Database Systems **15** (1990), 208–229.

[75] T. Zseby, M. Molina, N. Duffield, S. Niccolini, and F. Raspall, *Sampling and filtering techniques for IP packet selection*, January 2006.

# Résumé

Cette thèse s'inscrit dans le domaine de l'analyse et la modélisation du trafic Internet à l'échelle des flots. Les informations sur les flots (surtout les grands flots) sont très utiles dans différents domaines comme l'ingénierie du trafic, la supervision du réseau et la sécurité. L'extraction en ligne des statistiques sur les flots est une tâche difficile à cause du très haut débit du trafic actuel. Nous nous sommes intéressés dans cette thèse à l'étude de deux classes d'algorithmes traitant en ligne le trafic Internet.

Dans la première partie, nous avons conçu un nouvel algorithme basé sur les filtres de Bloom pour l'identification en ligne des grands flots. Le point fort de cet algorithme est l'adaptation automatique aux variations du trafic. Une application intéressante est la détection en ligne des attaques par déni de service. Nous avons donc développé une version de l'algorithme qui intègre les spécificités des attaques. L'expérimentation en ligne montre que cette nouvelle méthode est capable d'identifier quasiment toutes les sources de trafic anormal avec un délai très court. Nous avons aussi étudié la performance de l'algorithme d'identification en ligne des grands flots. En considérant un modèle simplifié, nous avons pu approcher l'erreur générée par cet algorithme sur l'estimation du nombre de grands flots. Cette étude a permis en particulier d'évaluer l'impact des différents paramètres de l'algorithme sur sa performance.

Les algorithmes présentés dans la première partie s'appliquent sur la totalité du trafic, ce qui n'est pas toujours possible car dans certains cas, on ne dispose que du trafic échantillonné. La deuxième partie de la thèse est consacrée à l'étude de l'échantillonnage et des algorithmes d'inférence des caractéristiques du trafic d'origine. D'abord, en utilisant un résultat d'approximations poissonniennes, nous avons montré que les deux méthodes d'échantillonnage : déterministe et probabiliste donnent des résultats équivalents du point de vue composition du trafic échantillonné en flots. Ensuite, nous avons conçu un algorithme permettant d'estimer, par un calcul asymptotique, à partir du trafic échantillonné, le nombre de flots dans le trafic réel et la distribution de leur taille sur un intervalle de temps court. Ceci permet de faire l'hypothèse à priori que cette distribution suit une loi de Pareto. Cette hypothèse a été validée sur des traces de trafic de différentes natures.

**Mots-clés:** Filtres de Bloom, grands flots, attaques par déni de service, échantillonnage, loi de Pareto.

# Abstract

This thesis is a contribution to the field of Internet traffic analysis at the flow level. For traffic engineering purposes like supervision and security for example, it is important to be able to characterize flows, especially the large ones. Due to the very high bit rate and the huge number of flows in IP traffic, it is very difficult to extract on-line statistics on flows. In this thesis we focused on two kinds of on-line algorithms for Internet traffic analysis.
In the first part, we developed a new algorithm based on Bloom filters for large flows identification. The advantage of this algorithm is it can adapt to traffic variations. An interesting application to this algorithm is the on-line detection of denial of service attacks. For this purpose, we proposed an adapted algorithm taking into account attacks specificities. On-line experiments show that this new method is able to identify almost all sources of anomalous traffic in a very short delay. In addition, we analyzed the performances of the algorithm for on-line identification of large flows. We analytically estimated the error generated by the algorithm on the number of elephants.
The algorithms presented in the first part are performed on the exhaustive traffic which is not usually possible because in some cases we have only access to the sampled traffic. The second part of the thesis is dedicated to sampling analysis and the study of algorithms inferring the original traffic characteristics. First, using a result on Poisson approximations, we proved that the two sampling methods : deterministic and probabilistic give equivalent results in terms of sampled traffic composition at the flow level. Then we developed a new method inferring, from the sampled traffic, via asymptotic procedures, flows number and size distribution in a small time window. This enables us to suppose à priori that this distribution is a Pareto. This hypothesis was validated against different traffic traces.



**Keywords:** Bloom filters, large flows, denial of service attacks, sampling, Pareto.