**Université de Nice - Sophia Antipolis**
École Doctorale STIC

# THÈSE

A présenter pour obtenir le titre de :

*Docteur en Sciences de l'Université de Nice - Sophia Antipolis*

Spécialité : INFORMATIQUE

par

Mohammad MALLI

Équipe d'accueil : Planète – INRIA Sophia Antipolis

## INFERRING INTERNET TOPOLOGY FROM APPLICATION POINT OF VIEW

Thèse dirigée par Chadi BARAKAT et Walid DABBOUS

A soutenir publiquement à l'INRIA le 22 Septembre 2006 devant le jury composé de :

| | | | |
|---|---|---|---|
| Président : | Michel | RIVEILL | UNSA/UFR EPU, France |
| Co-directeurs : | Chadi | BARAKAT | INRIA, France |
| | Walid | DABBOUS | INRIA, France |
| Rapporteurs : | Torsten | BRAUN | University of Bern, Switzerland |
| | Nazim | AGOULMINE | University of INT-Evry, France |
| Examinateurs : | Katia | OBRACZKA | University of California Santa Cruz, USA |
| | Laurent | MATTY | Lancaster University, UK |

# THÈSE

## LA TOPOLOGY DE L'INTERNET VUE PAR LES APPLICATIONS

## INFERRING INTERNET TOPOLOGY FROM APPLICATION POINT OF VIEW

MOHAMMAD MALLI

September 2006

# INFERRING INTERNET TOPOLOGY FROM APPLICATION POINT OF VIEW

Mohammad Malli
Thesis Supervisors: Chadi Barakat and Walid Dabbous
Planète Project, INRIA Sophia Antipolis, France

## ABSTRACT

In Peer-to-Peer and overlay networks, the quality of service perceived by end-users can be optimized at the application level by identifying the best peer to contact or to take as neighbor. This requires to define a proximity function that evaluates how much two peers are close to each other from application point of view.

Different functions are introduced in the literature to characterize the proximity of peers, but most of them are based on simple metrics such as the delay, the number of hops and the geographical location. We believe that these metrics are not enough to characterize the proximity given the heterogeneity of the Internet in terms of path characteristics and access link speed, and the diversity of application requirements. Some applications (e.g., transfer of large files and interactive audio service) are sensitive to other network parameters such as the bandwidth and the loss rate.

Thus, the proximity should be defined at the application level taking into consideration the network metrics that decide on the application performance. To this end, we introduce in this dissertation the notion of *CHESS*, an application-aware space for enhanced scalable services in overlay networks. In this new space, the proximity is characterized according to a utility function that models the quality perceived by peers at the application level. A peer is closer than another one to some third peer if it provides a better utility function, even if the path connecting it to the third peer is longer.

We begin by studying whether the delay proximity is a good approximation of the proximity in CHESS. We try to answer this question with extensive measurements carried out over the Planetlab platform. To this end, we consider a large set of peers and we measure path characteristics among them. Then, we consider two typical applications: a file transfer running over the TCP protocol, and an interactive audio service. For each application, we propose a metric that models the application quality by considering the critical network parameters (e.g., delay, bandwidth, loss rate) affecting the application performance. Using this metric, we evaluate the enhancement of the performance perceived by peers when they choose their neighbors based on the proximity in CHESS instead of the delay-based one.

Our main observation is the following. Delay, bandwidth and loss metrics are slightly correlated, which means that, in our setting, one cannot rely on one of these metrics in defining proximity when the application is more sensitive to the others. For example, if one uses the delay to decide on the closest peer to contact for a file transfer, the application performance deteriorates compared to the optimal scenario where neighbors are identified based on the predicted file transfer latency. Furthermore, if one contacts the delay closest peer for an interactive

audio service, the speech quality is not as high as that obtained when the peer to contact is the one providing the best predicted speech rating.

Then, we propose a model for a scalable estimation of the bandwidth among peers which is required to deploy CHESS. Our model estimates the bandwidth among peers using the bandwidth of the indirect paths that join them via a set of well defined proxies or relays that we call landmark nodes. Basically, the direct and the indirect path share the same tightest link with some probability that depends on the location of the correspondent landmark with respect to the direct path or to one of the path end points. This probability is higher if the landmark is closer to one of the path end points. It can be also higher if the delay of the indirect path is nearer to that of the direct one. Thus, the bandwidth of each indirect path contributes to the estimation of the bandwidth of the direct one according to some defined probability.

Again, using Planetlab measurements, we evaluate the solution and analyze the impact of the location, number, and distribution of the landmarks on the accuracy of the estimation. We obtain satisfactory results when the delays of some indirect paths are close to that of the direct path. Better results are obtained when some landmarks are located near one of the extremities of the direct path to characterize. Moreover, the results show that our estimation model is able to infer accurately the bandwidth among a worldwide set of peers when using 40 to 50 landmarks.

Finally, we compare the proximity obtained in the CHESS space with that obtained in the delay space from application point of view. A typical file transfer application is considered to evaluate the quality of service perceived by peers when they choose their neighbors based on these two distinguished proximity notions. We observe that the proximity in CHESS, which is determined easily and scalably using our model for bandwidth estimation, provides a much better quality compared to that obtained when using the delay alone.

# LA TOPOLOGIE DE L'INTERNET VUE PAR LES APPLICATIONS

Mohammad Malli
Directeur de thèse: Chadi Barakat et Walid Dabbous
Projet Planète, INRIA Sophia Antipolis, France

## RÉSUMÉ EN FRANÇAIS

L'utilisation répandue des réseaux Pair à Pair et *Overlay* justifie la nécessité d'optimiser la performance perçue par les utilisateurs au niveau applicatif. Ceci revient à définir une fonction de proximité qui évalue combien deux pairs sont proches l'un de l'autre. La caractérisation de la proximité aide à identifier le meilleur pair à contacter ou à prendre comme voisin.

Différentes fonctions ont été proposées dans la littérature pour caractériser la proximité entre les pairs, mais la plupart d'entre eux [8, 9, 21, 22, 23, 24, 25, 26, 27, 28, 29] sont basés sur une métrique simple telle que le délai, le nombre des sauts, et l'endroit géographique. Nous pensons que ces métriques ne sont pas optimales pour caractériser la proximité à cause de l'hétérogénéité de l'Internet en termes de caractéristiques des chemins, la vitesse des liens, et la diversité des conditions d'application. Quelques applications (par exemple, transfert de grands fichiers et service audio interactif) sont sensibles à d'autres paramètres de réseau comme la bande passante disponible et le taux de perte.

Par conséquent, la proximité doit être définie au niveau applicatif en prenant en compte les paramètres du réseau qui ont un impact sur la performance de l'application. À cet effet, nous présentons dans cette thèse la notion de *CHESS* (une abréviation de *an application-aware spaCe for enHancEd Scalable Services in overlay networks*), qui est un espace applicatif construit en fonction des besoins d'application. Dans ce nouvel espace, la proximité se caractérise selon une fonction d'utilité qui modélise la qualité de service perçue par les pairs au niveau applicatif. Un pair est plus proche qu'un autre d'un certain troisième pair s'il fournit une meilleure fonction d'utilité, même si le chemin qui le relie au troisième pair est plus long.

Nous commençons la dissertation en étudiant si la proximité du délai est une bonne approximation de la proximité dans CHESS [4]. Nous essayons de répondre à cette question par des mesures étendues effectuées au-dessus du réseau expérimental mondial *Planetlab*. À cet effet, nous considérons un grand ensemble de pairs et nous mesurons les caractéristiques des chemins qui les joignent. Nous considérons par la suite deux applications typiques: un transfert de fichier fonctionnant au-dessus du protocole TCP, et un service audio interactif. Pour chaque application, nous proposons une métrique qui modélise sa qualité en considérant les paramètres critiques du réseau (par exemple, le délai, la largeur de la bande passante disponible, le taux de perte) qui ont un impact sur la performance de l'application [5]. En outre, nous évaluons le perfectionnement de la performance perçue par les pairs quand ils choisissent leurs voisins en se basant sur la proximité dans CHESS, qui est déterminé en utilisant les fonctions de qualité proposées au lieu de celle basée sur le délai.

Notre observation principale est la suivante: le délai, la largeur de la bande passante disponible et le taux de perte sont légèrement corrélés, ce qui signifie qu'on ne peut pas baser

sur un de ces métriques pour définir la proximité quand l'application est plus sensible aux autres. Par exemple, si nous utilisons le délai pour décider du pair le plus proche et ceci dans le but de le contacter pour un transfert de fichier, la performance de l'application dégrade par rapport au scénario optimal où les voisins sont identifiés en se basant sur la prédiction du temps de transfert des fichiers. En outre, si nous contactons le pair le plus proche en termes de délai pour un service audio interactif, la qualité de la parole n'est pas aussi bonne que celle obtenue quand le pair contacte celui avec qui on prévoit le meilleur facteur de qualité de la parole.

La contrainte principale pour déterminer la proximité dans CHESS est l'inférence des paramètres du réseau qui sont incluent dans la fonction de qualité d'une manière qui passe à l'échelle. En d'autres termes, l'estimation des paramètres de réseau sur le chemin joignant deux pairs quelconques dans un grand système devrait être réalisée avec la moindre quantité de mesure et une coopération limitée entre les pairs.

Nous commençons par montrer que la méthode de factorisation de matrice [30, 31], qui est proposé initialement pour estimer le délai, est appropriée pour estimer aussi le taux de perte. Cependant, ce n'est pas le cas pour la bande passante disponible qui n'est pas du tout une métrique additive. La bande passante disponible dépend du goulot d'étranglement qui peut apparaître sur n'importe quel lien de chemin. Alors, un modèle d'estimation de la bande passante disponible doit identifier l'itinéraire reliant chaque couple de pairs pour pouvoir mesurer le lien de goulot d'étranglement. Le défi est qu'un tel modèle doit passer à l'échelle et être facile à déployer.

A partir de ce résultat, nous proposons un modèle pour estimer la bande passante disponible entre les pairs d'une façon qui passe à l'échelle [3]. Notre modèle estime la bande passante disponible sur les chemins directs (c-à-d, chemins IP) qui joignent les pairs en utilisant la bande passante disponible sur les chemins indirects. Ces chemins les relient par l'intermédiaire d'un ensemble de relais bien définis que nous appelons landmarks. En principe, les chemins direct et indirect se partagent le même goulot avec une certaine probabilité qui dépend de l'endroit du landmark correspondant par rapport au chemin direct ou à une des extrémités du chemin. Cette probabilité est plus haute si le landmark est plus proche d'une des extrémités de chemin. Elle peut être également plus haute si le délai du chemin indirect est plus proche de celui du chemin direct. C'est pourquoi nous supposons que la bande passante disponible sur chaque chemin indirect contribue à l'estimation de la bande passante du chemin direct suivant une certaine probabilité que nous définissons.

En utilisant encore des mesures sur Planetlab, nous évaluons la solution et analysons l'impact de l'endroit et du nombre des landmarks sur l'exactitude de l'estimation [1]. Nous découvrons que notre modèle d'estimation peut inférer exactement la bande passante disponible sur les chemins, ceux qui joignent un ensemble de pairs mondialement distribués en utilisant un nombre entre 40 et 50 landmarks.

Finalement, nous comparons la proximité caractérisée dans l'espace CHESS à celle obtenue dans l'espace du délai du point de vue application [2]. Une application typique de transfert de fichier est considérée pour évaluer la qualité du service perçue par les pairs quand ils choisissent leurs voisins en se basant sur ces deux notions distinguées de proximité. Nous déterminons la proximité dans CHESS entre un ensemble de noeuds de Planetlab mondialement distribués en utilisant notre métrique [5] et ceci dans le but de prévoir le temps de transfert. Nous observons que la caractérisation de cette proximité, en utilisant notre modèle d'estimation de la bande passante, mène à une qualité bien meilleure que celle obtenue en utilisant seulement le délai.

# CONTENTS

**Abstract** **iii**

**Résumé en Français** **vi**

**List of Figures** **xiv**

**List of Tables** **xv**

**1 Introduction** **1**
1.1 Thesis domain: Internet topology inference . . . . . 1
1.2 Thesis contribution: application-centric approach . . . . . 3
1.3 Thesis Organization . . . . . 5

**2 An overview of Internet topology inference** **7**
2.1 Summary . . . . . 7
2.2 Introduction and motivations . . . . . 7
2.3 Active measurement tools . . . . . 9
2.4 Active approaches . . . . . 13
2.4.1 Coordinate-based approaches . . . . . 13
2.4.2 Traceroute-based approaches . . . . . 15
2.4.3 Other active approaches . . . . . 21
2.5 Conclusions and Perspectives . . . . . 26

**3 Motivating the proximity in CHESS** **29**
3.1 Summary . . . . . 29
3.2 Introduction . . . . . 29
3.3 Data setup and notations . . . . . 30
3.4 Motivating CHESS . . . . . 32

3.4.1 Delay vs. bottleneck capacity . . . 32
3.4.2 Delay vs. available bandwidth . . . 33
3.4.3 Bottleneck vs. available bandwidth . . . 35
3.4.4 Delay vs. loss rate . . . 37
3.4.5 Load vs. loss rate . . . 37
3.5 Introducing CHESS . . . 40
3.5.1 Goal . . . 40
3.5.2 Potential applications . . . 40
3.5.3 Challenges . . . 46
3.6 Conclusions . . . 47

**4 Application quality enhancement due to the proximity in CHESS 51**
4.1 Summary . . . 51
4.2 Introduction . . . 51
4.3 File transfer over TCP . . . 52
4.3.1 Transfer Time Prediction Function . . . 53
4.3.2 Evaluating the efficiency of our latency metric . . . 59
4.3.3 Impact of Proximity definition on the transfer latency . . . 62
4.4 Interactive audio service . . . 65
4.5 Conclusions . . . 69

**5 A Scalable Characterization of the Proximity in CHESS 71**
5.1 Summary . . . 71
5.2 Introduction and Motivation . . . 72
5.3 Distance Matrix Factorization . . . 73
5.4 Scalable end-to-end bandwidth inference . . . 79
5.4.1 Model Overview . . . 79
5.4.2 Impact of landmarks' locations . . . 83
5.4.3 Bandwidth estimation function . . . 88
5.4.4 Impact of the number of landmarks . . . 89
5.5 Enhanced proximity perceived by the application . . . 93
5.6 Conclusions . . . 95

**6 Conclusions and future work 97**

**A Présentation des Travaux de Thèse en Français** **99**
A.1 Introduction . . . . . 99
A.1.1 Domaine de la thèse: Inférence de la topologie de l'Internet . . . 100
A.2 Contribution de la thèse: proximité applicative . . . . . 101
A.2.1 L'organisation de la thèse . . . . . 105
A.3 Chapitre 2: Une vue d'ensemble des approches inférant la topologie . 106
A.4 Chapitre 3: Motivation de la proximité dans CHESS . . . . . 109
A.5 Chapitre 4: L'amélioration de la performance dû à la proximité dans CHESS . . . . . 110
A.6 Chapitre 5: Déploiement à grande échelle de la proximité dans CHESS . 112
A.7 Chapitre 6: Conclusions et travaux futurs . . . . . 115

**Bibliography** **118**

# LIST OF FIGURES


2.1 Paradigm of Internet Topology Inference . . . . . 10
2.2 The simple Vivaldi algorithm, with a constant timestep $\delta$. . . . . 15
2.3 A network embedding example . . . . . 15
2.4 An example on router inflation after traceroute aggregation . . . . . 17
2.5 Sample visualization from Skitter data (taken from [60]) . . . . . 18
2.6 Architecture of Rocketfuel. The Database (DB) becomes the inter-process communication substrate. . . . . 19
2.7 Path reductions . . . . . 20
2.8 Alias resolution by IP identifiers. A solid arrow represents messages from an IP to an alias, the dotted arrow to another alias. . . . . 21
2.9 Example on routing in 2-d space . . . . . 23
2.10 An example describing the closest node discovery achieved by Meridian (taken from [8]) . . . . . 25

3.1 Delay versus bottleneck bandwidth . . . . . 34
3.2 Delay versus available bandwidth . . . . . 36
3.3 Bottleneck bandwidth versus available bandwidth . . . . . 38
3.4 Delay versus loss rate . . . . . 39
3.5 Load versus loss rate . . . . . 41
3.6 An example scheme for best server selection . . . . . 43

4.1 Transfer time prediction accuracy . . . . . 60
4.2 Transfer time enhancement when ranking peers is based on the proximity in CHESS instead of the delay proximity . . . . . 64
4.3 Speech quality degradation when ranking peers is based on the proximity in CHESS instead of the delay proximity . . . . . 68

5.1 Matrix reconstruction error after the reduction to the different dimensions over the 3 Planetlab datasets . . . . . 77
5.2 Direct versus Indirect paths . . . . . 82
5.3 Bandwidth estimations based on indirect paths' delays . . . . . 85
5.4 Bandwidth estimations based on the delay distance between landmarks and peers . . . . . 87
5.5 Mean estimation error obtained for different number of landmarks chosen in a random, max-distance, and N-means ways . . . . . 90
5.6 Standard deviation of the estimation error obtained for different number of landmarks chosen in a random, max-distance, and N-means ways . . 91
5.7 Transfer time enhancement when using the proximity in CHESS instead of the delay proximity . . . . . 94

# List of Tables


2.1 Mostly used tools with their characteristics measured . . . . . 11
2.2 Tools classified according to their measurements: Per-Path or Per-Hop . . 11

4.1 Transfer time prediction when $A$ limits the download rate . . . . . 61
4.2 Transfer time prediction when $W_{max}$ limits the download rate . . . . . 62
4.3 Transfer time prediction when P limits the download rate . . . . . 62
4.4 Rfactor intervals, quality ratings, and the associated MOS . . . . . 66

5.1 Estimation error by SVD over dataset 1 . . . . . 80
5.2 Estimation error by SVD over dataset 2 . . . . . 80
5.3 Estimation error by SVD over dataset 3 . . . . . 80

# 1

# INTRODUCTION

In this thesis, we propose an active approach that better characterizes the proximity among overlay network nodes. It consists in determining the proximity among network nodes at the application-level based on application-specific requirements in order to enhance the quality of service perceived by end users. We begin the dissertation by presenting, in Chapter 2, an overview of the active approaches that have been proposed in the literature for network topology inference. Then, we present the motivation, description, and experimentation of our proposal [1, 2, 3, 4, 5] in Chapters 3, 4, and 5.

## 1.1 Thesis domain: Internet topology inference

The worldwide increasing number of Internet users and networks, with the absence of any centralized administration, makes the topology of the Internet more and more complex. A huge number of routers are installed in the backbone and edges of the Internet in order to provide a high degree of connectivity between the different nodes (e.g., end hosts, servers) spread over the globe. Besides, digital contents become shared among the Internet users and stored in replicated servers distributed across the Internet in order to improve the quality of service offered to the users and to provide a high availability of data. The replication of data over different nodes makes the choice of its location a challenging problem that requires a knowledge of the topology to make it optimal.

There are two main approaches for inferring the network topology: the active ap-

proach and the passive approach. The active approach is based on sending messages (e.g., ICMP echo messages) among network nodes (e.g., end hosts, proxies, etc.) in order to infer the network topology (e.g., [6, 7, 8, 9]). Basically, active probing can be used to infer the topology of all the Internet, of a particular ISP or of a particular AS. Furthermore, it can be used for characterizing the network proximity (e.g., in terms of delay) among overlay nodes as well as the performance on the paths joining them. The approach that we propose in this thesis belongs to the class of active approach. In the next chapter, we review the body of the literature that fit within this class of approaches.

The passive approach [10, 11, 12, 13, 14, 15, 16, 17] consists in inferring the network topology without injecting new packets (probing packets) into the network. It consists in inferring the topology by observing the data traffic circulating across the network, by looking at the routing tables (e.g., BGP [18] routing tables), or by observing the control messages exchanged between routers. This class of approaches is not explored in this thesis.

We define briefly the key network entities that are frequently mentioned in the following discussions:

- Autonomous System (AS), which is a collection of routers under a single administrative authority, using a common Interior Gateway Protocol for routing packets.

- Internet Service Provider (ISP), is the network manager and administrator who provides the Internet access to a part or all the end users of a certain AS.

- Content Distribution Network (CDN), where client requests are forwarded by request redirectors, and where the contents are stored in mirror servers geographically distributed over the Internet. Many companies, like Akamai [19], provide CDNs to content providers.

- Peer-to-Peer network (e.g., BitTorrent [20]), where peers behave as clients and servers.

- An overlay network is a virtual network of nodes connected by logical links on top of one or more existing networks. There are two main kinds of overlay networks which are peer-to-peer network and CDN network.

## 1.2 Thesis contribution: application-centric approach

The emerging widespread use of Peer-to-Peer (P2P) and overlay networks argues the need to optimize the performance perceived by users at the application level. This amounts to defining a proximity function that evaluates how much two peers are close to each other. The characterization of the proximity helps in identifying the best peer to contact or to take as neighbor.

Different functions are introduced in the literature to characterize the proximity of peers, but most of them [8, 9, 21, 22, 23, 24, 25, 26, 27, 28, 29] are based on simple metrics such as the delay, the number of hops, and the geographical location. We believe that these metrics are not enough to characterize the proximity given the heterogeneity of the Internet in terms of path characteristics and access link speed, and the diversity of application requirements. Some applications (e.g., transfer of large files and interactive audio service) are sensitive to other network parameters as the bandwidth and the loss rate.

Therefore, the proximity should be defined at the application level taking into consideration the network metrics that decide on the application performance. To this end, we introduce in this thesis the notion of *CHESS*, an application-aware space for enhanced scalable services in overlay networks. In this new space, the proximity is characterized according to a utility function that models the quality perceived by peers at the application level. A peer is closer than another one to some third peer if it provides a better utility function, even if the path connecting it to the third peer is longer.

We begin by studying whether the delay proximity is a good approximation of the proximity in CHESS [4]. We try to answer this question with extensive measurements carried out over the Planetlab platform. To this end, we consider a large set of peers and we measure path characteristics among them. Then, we consider two typical applications: a file transfer running over the TCP protocol, and an interactive audio service. For each application, we propose a metric that models the application quality by considering the critical network parameters (e.g., delay, bandwidth, loss rate) impacting the application performance [5]. Furthermore, we evaluate the enhancement of the performance perceived by peers when they choose their neighbors based on the proximity in CHESS, which is determined using the proposed quality functions instead of the delay-based one.

Our main observation is the following. Delay, bandwidth and loss metrics are slightly correlated, which means that, in our setting, one cannot rely on one of these

metrics to define the proximity when the application is more sensitive to the others. For example, if one uses the delay to decide on the closest peer to contact for a file transfer, the application performance deteriorates compared to the optimal scenario where neighbors are identified based on the predicted file transfer latency. Furthermore, if one contacts the delay closest peer for an interactive audio service, the speech quality is not as high as that obtained when the peer to contact is the one providing the best predicted speech rating factor.

The main constraint for determining the proximity in CHESS is how to infer the network parameters that impact the utility function in an easy and scalable way. In other terms, the estimation of the network parameters between any two peers in a large system should be achieved with a small measurement overhead and a limited cooperation among peers.

We start by showing that the matrix factorization approach [30, 31], initially proposed for estimating the end-to-end delay, is appropriate for estimating the end-to-end loss rate as well. However, this is not the case for the bandwidth which is far from being an additive metric. The bandwidth depends on the bottleneck link that may appear anywhere along an end-to-end path. Any model for bandwidth estimation has to identify the route connecting each couple of peers to be able to gauge the bottleneck link. The challenge is that such model must be scalable and easy to deploy.

Given this result, we propose a model for estimating the bandwidth among peers in an easy and scalable way [3]. Our model estimates the bandwidth among peers using the bandwidth of the indirect paths that join them via a set of well defined proxies or relays that we call landmark nodes. Basically, the direct path (i.e., IP path) and the indirect path share the same tight link with some probability that depends on the location of the correspondent landmark with respect to the direct path or to one of the path end points. This probability is higher if the landmark is closer to one of the path end points. It can be also higher if the delay of the indirect path is closer to that of the direct one. Thus, it can be assumed that the bandwidth of each indirect path contributes to the estimation of the bandwidth of the direct one according to some probability that we define.

Again, using Planetlab measurements, we evaluate the solution and analyze the impact of the location and number of landmarks on the accuracy of the estimation [1]. We find out that our estimation model is able to infer accurately the bandwidth among a worldwide set of peers using 40 to 50 landmarks.

Finally, we compare the proximity obtained in the CHESS space to that obtained in

the delay space from application point of view [2]. A typical file transfer application is considered to evaluate the quality of service perceived by peers when they choose their neighbors based on these two distinguished proximity notions. We determine the proximity in CHESS among a worldwide distributed set of Planetlab nodes using our transfer time prediction metric [5]. We observe that the characterization of this proximity, using our bandwidth estimation model, leads to a much better quality than that obtained when using the delay alone.

## 1.3 Thesis Organization

The dissertation is structured as the following. In the next chapter, we present an overview of the active approaches proposed in the literature for inferring the topology of the Internet. Particularly, we focus on those relying on Internet coordinate systems, and those using traceroute probes.

In Chapter 3, we first motivate the need for the CHESS space by studying the correlation among the different path characteristics. Then, we explain how such proximity notion can be particularly useful for server selection and overlay construction.

In Chapter 4, we provide a comparison between the delay proximity and the one characterized in CHESS. This is done by evaluating the impact of these two proximity notions on the application performance. For this purpose, we consider two typical applications: a file transfer running over the TCP protocol and an interactive audio service. For each application, we first develop utility function predicting the performance quality. Then, we evaluate the enhancement of the performance perceived by peers when they choose their neighbors based on the proximity in CHESS instead of the delay-based one.

In Chapter 5, we first describe the matrix factorization approach and study its capacity to estimate the delay, the loss rate and the available bandwidth. Then, we propose a model for estimating the bandwidth among peers in an easy and scalable way. In the last part of this chapter, we evaluate the performance gain achieved when considering the bandwidth estimations for determining the proximity in the CHESS space. Finally, the dissertation is concluded in Chapter 6 with some perspectives on the future research in this area.

# 2

# AN OVERVIEW OF INTERNET TOPOLOGY INFERENCE

## 2.1 Summary

Inferring the topology of the Internet is not a trivial task due to the immense scale of the Internet, its continuous evolution, and its distributed administration. At the same time, topology information is of particular importance for network operators and users. It guarantees an efficient operation of the network, and a better tuning of its protocols and applications. We present, in this chapter, an overview of the body of the literature that deals with Internet topology inference. We, particularly focus on the contributions classified as active approaches.

## 2.2 Introduction and motivations

Inferring the network topology is a huge research domain that becomes of higher interest with the increasing evolution of the Internet. It consists in providing specific network topology information as locating network nodes, characterizing their connectivity, and determining the performance on the network nodes and their joining paths. This can be realized at the following three main network levels: (i) at the router level,

for a particular ISP or for all the Internet, (ii) at the Autonomous System (AS) level, and (iii) at the application level.

Knowing the topology of the Internet is interesting from different standpoints as described in the following:

- An Internet user (resp. peer) can profit from inferring the network topology for different purposes. Here are some examples:
  - Identifying the best server (resp. best peer) to download a content in point-to-point or the best set of servers (resp. best peers) to download the content in parallel.
  - Optimizing the construction of structured peer-to-peer network to minimize the lookup time for a content.
  - Optimizing the construction of overlay multicast trees for content distribution (e.g., video distribution) so as to enhance the quality of service perceived by end users.
  - Optimizing the performance achieved by end-to-end protocols (e.g., data collection protocols) [32].
  - Identifying the best Internet access when there are many ones (multi-homed machine).
- A digital content service provider can use the inferred topology information to enhance the quality of its applications. Among the applications, one can find:
  - Better distribution of its servers across the Internet, given the network distribution of clients who request the service.
  - Better distribution of web proxies and better policies to push documents and update the content of caches.
  - Adjust the structuring of overlay network infrastructure (e.g., overlay content distributors) to enhance the performance of content distribution (e.g., video distribution).
  - Better redirection of clients to the best server handling the requested information.

- An ISP can use the topology information to optimize its network. We mention the following main purposes:
  - Analyzing routing protocols and network robustness and resilience. As an example, we mention the profit from an efficient network-layer restoration: to switch sending packets from one interface to another based on certain criteria such as a link panic or a QoS degradation.
  - Better connections with the other ISPs, and better setting of BGP routing tables.
  - Verifying if the QoS promised by a neighboring ISP is satisfied.
  - Discovering the prefixes that are responsible of a nefarious behavior such as a pattern attack, anomalies, etc.; along with knowing the locations and the ISPs of these prefixes.
  - For traffic engineering purposes inside an ISP's network. As an example, we mention the profit from locating unpleasant points and avoiding them by re-routing traffic through another path inside the network.
- A research scientist can profit from inferring the network topology for studying the evolution of the Internet and for obtaining more realistic scenarios in their research studies. This can help then to solve the actual problems such as those concerning the network protocols behavior.

In this chapter, we present a briefing on the active approaches that have been proposed in the literature for network topology inference. The outline of the chapter is as follows. Next, we present an overview of the active measurement tools that are useful for network topology inference. Then, we describe the different proposed works that fit within the active approach for inferring the network topology. Particularly, we focus on those relying on Internet coordinate systems, and those using traceroute probes. Finally, we conclude the chapter in Section 5.6.

## 2.3 Active measurement tools

As described in Figure 2.1, inferring the network topology requires the invention of an inference architecture. This architecture depends on a network measurement

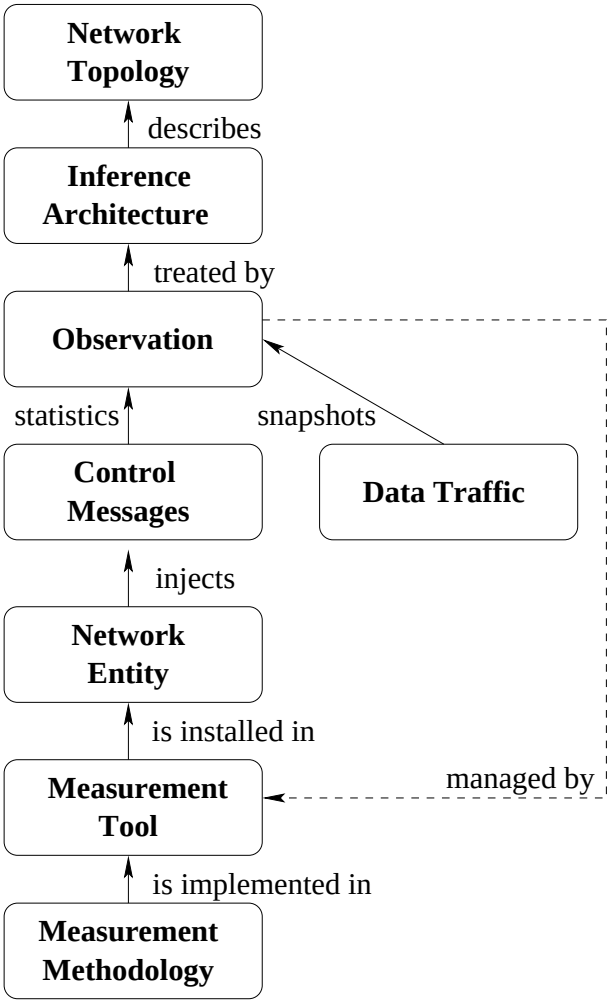



**Figure** 2.1: Paradigm of Internet Topology Inference

tool along with many other engines as the one that is necessary for manipulating the obtained traces. Thus, measuring the performance among network entities is one of the main engines which are necessary for inferring the network topology. Network measurement can be realized by using either an active tool or a passive tool.

A passive tool [33, 35, 36, 37, 38] analyzes network traffic by collecting packets traversing a network link, or a switch/router device. The measurement can be used for showing network traffic in real-time or for analyzing the traffic off-line after saving the measured data. Observing and analyzing the network traffic are important for inferring the network topology and optimizing mainly the communication protocols. There are two main types of passive tools:

| BW capacity | Available BW | Achievable BW | Round Trip Time | Loss | Hoplist |
|---|---|---|---|---|---|
| bing | cprobe | Iperf | bing | bing | traceroute |
| bprobe | netest | netest | clink | netest | |
| clink | pathload | Netperf | netest | pathchar | |
| Nettimer | pipechar | ttcp | pathchar | pchar | |
| pathchar | TReno | | pchar | pipechar | |
| pathrate | Abing | Abing | pipechar | ping | |
| pchar | IGI | | ping | Abing | |
| sprobe | pathchirp | | | | |

**Table** 2.1: Mostly used tools with their characteristics measured

| Per-Path | Per-Hop |
|---|---|
| bing, bprobe, cprobe, ttcp | clink |
| Iperf, netest, Netperf, TReno | pathchar |
| Nettimer, pathload, pathrate, Abing | pchar |
| ping, sprobe, traceroute | pipechar |

**Table** 2.2: Tools classified according to their measurements: Per-Path or Per-Hop

- Tools that are used for monitoring packets traversing network interfaces such as Tcpdump [33], IPMON [35], and multiQ [36].
- Tools that are used for providing router/switch traffic statistics such as SNMP [37] and Netflow [38].

On the other hand, an active measurement tool is a software module which aims to describe the characteristics of a network entity by injecting stimulus packets (probing packets) into the network and then measuring the response. Most of the proposed inference approaches rely on active measurement tools. We present in this section a brief overview of the main active tools that have been proposed in the literature.

The characteristics measured on the path between two nodes can be the bandwidth capacity, the available bandwidth, the achievable bandwidth, the round trip time delay, the loss rate, and the hop list. Table 2.1 classifies the mostly used tools according to the measured characteristics. These characteristics are measured per path by some tools and per hop (refers to the IP level) by other tools as described in Table 2.2.

Measurement methodologies vary from one active tool to another depending on

which technique they use: sending ICMP echo, sending UDP packets with port unreachable, varying packets' TTL, varying packets' size, sending packet pairs, sending packet trains, using path flooding, using SLOPS (Self-Loading Periodic Streams), and using TCP simulation.

The basic tool in active measurement for topology inference is Traceroute. It can provide the unidirectional IP path from its location to a destination and the round trip time of each hop on the path. A unidirectional IP path is defined as the IP devices traversed by IP packets traveling from a source to a destination at a single point in time. Traceroute consists in setting the IP TTL field from 1 to $n$ until the ultimate destination is reached. This behavior leads to sending a packet to each router along a path without actually knowing the path. Upon receiving a packet with an expired TTL (equal to 0), the hop generates an ICMP Time Exceeded response back to the source, thus identifying the hop and its round trip delay. Each UDP packet is sent to a highly numbered port (probably-unused), so when the destination receives the packet it responds with ICMP Port Unreachable.

Many active tools have been proposed in the literature for measuring the end-to-end available bandwidth [39, 40, 41, 42, 43, 44, 45]. The earliest work (proposed by Keshav in [46]) was based on the packet pair dispersion. Then, Carter and Crovella propose the cprobe tool [48] for estimating the available bandwidth based on packet train dispersion. Instead of using pairs of packets, cprobe sends short trains of ICMP packets and computes the available bandwidth as the probe size divided by the interval between the arrival of the last ICMP ECHO and the first ICMP ECHO in a train. A similar approach is used by pipechar [49].

Besides, many tools [39, 41, 43, 51] have been proposed for measuring the end-to-end available bandwidth based on the probe rate model. It consists in sending a periodic packet stream (i.e., a number of equal sized packets) from the sender to the receiver at a certain rate and then monitoring the delay variation of the probing packets. Basically, if the stream rate is lower than the available bandwidth along the path, then the arrival rate of the probed stream at the receiver will match the sending rate at the source. In contrast, if the probing stream is sent at a rate higher than the available bandwidth, then it will be queued in the network and delayed before reaching the receiver. Thus, one can measure the available bandwidth by searching for the turning point at which the delay increases considerably and setting the available bandwidth to the probe sending rate that has lastly been used.

## 2.4 Active approaches

An active approach consists in providing specific network topology information by relying on a pre-defined inference architecture. In this section, we describe briefly the major active approaches proposed in the literature. We begin by describing the mostly known active approaches for locating network nodes; particularly, we focus on the network embedding approaches which assign to each node a position in a cartesian space. Then, we describe how to infer the network topology using the answers of traceroute probes.

### 2.4.1 Coordinate-based approaches

Inferring network distances and characteristics (e.g., delay, bandwidth, loss rate) among a large number of peers without achieving a mesh-based measurements is a hard problem. Many approaches, as [8, 9, 21, 23, 24, 25, 26, 27], have been proposed for estimating the end-to-end delay among peers from a set of partially observed measurements. Most of these solutions are based on the network embedding. It consists in assigning to each peer a position (i.e., a vector of coordinates) in a cartesian space where the delay between any two peers is estimated by their Euclidean distance. Peers' coordinates are usually deduced from delay measurements to a number N of reference nodes which are called landmarks $L\{L_1, ..., L_N\}$.

More formally, suppose that the network contains $n$ peers $p = \{p_1, p_2, ..., p_n\}$. The delay on the paths joining peers is represented by an $n \times n$ matrix D, where $D_{ij}$ is the measured delay from $p_i$ to $p_j$. A network embedding is a mapping $C : p - \mathbb{R}^d$ in the following way:

$$D_{ij} \approx \| \overrightarrow{C}_i - \overrightarrow{C}_j \|, \forall i, j = 1, ..., n \tag{2.1}$$

where $\overrightarrow{C}_i = (C_{i1}, C_{i2}, ..., C_{id})$ is the coordinate vector of $p_i$ giving its position in a d-dimensional Euclidean space. Thus, the delay between any two peers $p_i$ and $p_j$ can be estimated by the Euclidean distance between their coordinates:

$$\widehat{D}_{ij} = \| \overrightarrow{C}_i - \overrightarrow{C}_j \| = \left( \sum_{k=1}^{d} (C_{ik} - C_{jk})^2 \right)^{\frac{1}{2}} \tag{2.2}$$

The challenge of network embedding is to assign a coordinate vector $C_i$ to each peer $p_i$ from a partially observed delay matrix D. Global Network Positioning System

(GNP) [25] was the first work in this field. In GNP, peers measure the delay to a set of well-known landmark nodes. Then, the Simplex Downhill method [52] is used to deduce peers' coordinates by minimizing an objective function representing the error between the estimated and the measured delays. The same algorithm has been applied in PIC [53], after investigating issues related to security. ICS [24] and Virtual Landmarks [23] are based on Lipschitz embedding and Principle Component Analysis (PCA). They embed the peers in a low dimensional Euclidean space obtained by compressing the full delay matrix using the PCA method.

In [9], Vivaldi simulates the overlay network by a network of physical springs. It proposes to determine peers' coordinates in a distributed way without the need to dispose of a fixed set of landmarks. Each peer contacts a random set of peers and adjusts its coordinates permanently until minimizing the potential energy of a spring system. Figure 2.2 shows the pseudocode for the Vivaldi algorithm. This code is called whenever a new RTT measurement is available. The node that did the measurement calculates the error in its current prediction to the target node (line 1). Then, it determines the direction along which it should move towards or away from the target node (lines 2 and 3). Finally, the node moves a fraction of the distance to the target node in line 4 using a constant timestep $\delta$. To obtain fast convergence and avoid oscillation, Vivaldi proposes the following adaptive timestep $\delta$:

$$\delta = \mathrm{constant} \cdot \frac{\mathrm{localerror}}{\mathrm{localerror} + \mathrm{remoteerror}} \tag{2.3}$$

By using such $\delta$ expression, an accurate node targeting an inaccurate node will not move much, an inaccurate node will move more, and two nodes of similar accuracy will split the difference.

Coordinate-based approaches suffer particularly from the fact that they provide Euclidean distances which are symmetric (i.e., $\widehat{D}_{ij} = \widehat{D}_{ji} \ \forall\, i,\ j$) and satisfy the triangle inequality (i.e., $\widehat{D}_{ij} + \widehat{D}_{jk} \geq \widehat{D}_{ik} \ \forall\, i,\ j,\ k$). This may not be consistent with the real network topology [54, 55, 56, 57]. Moreover, the authors in [31] indicate one more limitation that some peers probably do not have a direct path joining them. In this case, the estimated delay of these paths is inaccurate. This is illustrated in the example of Figure 2.3 where we present a simple network topology containing four peers connected only to their neighbors by unit delays. The estimated delays in the two-dimensional embedding are $\widehat{D}_{14} = \widehat{D}_{23} = \sqrt{2}$ while the real delays in this topology are $D_{14} = D_{23} = 2$. Similar cases arise in tree-based topologies. In such cases, it may that

```
// Node i has measured node j to be rtt ms away,
// and node j says it has coordinates x_j.
simple_vivaldi(rtt, x_j)
  // Compute error of this sample. (1)
  e = rtt − ||x_i − x_j||
  // Find the direction of the force the error is causing. (2)
  dir = u(x_i − x_j)
  // The force vector is proportional to the error (3)
  f = dir × e
  // Move a a small step in the direction of the force. (4)
  x_i = x_i + δ × dir
```

**Figure** 2.2: The simple Vivaldi algorithm, with a constant timestep $\delta$.

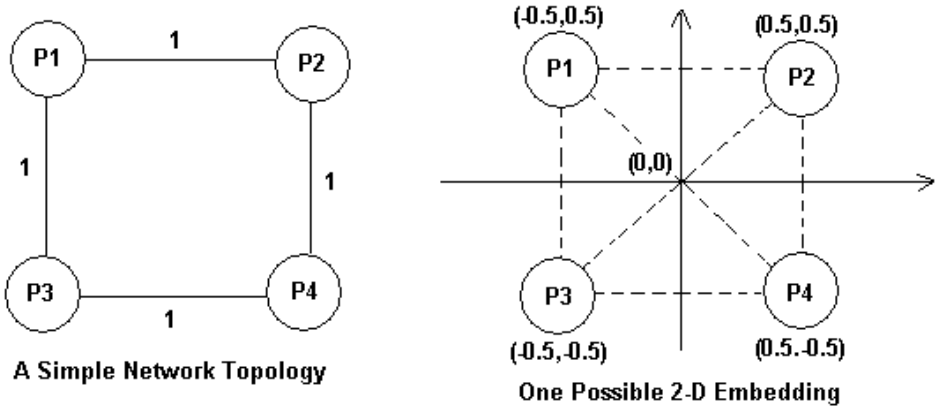



**Figure** 2.3: A network embedding example

there is no Euclidean embedding that can model exactly the real network delays.

In [31, 30], the authors propose to infer the network delay in a way that is able to model the sub-optimal routing and asymmetric routing policies. They do that by using the distance matrix factorization. In Chapter 5, we describe this powerful model and we test its capacity to estimate the delay, the loss rate and the available bandwidth.

### 2.4.2 Traceroute-based approaches

Many approaches are designed and implemented to infer the network topology using the basic idea of the traceroute tool. The major challenges of inferring the network topology using traceroute probes are:

- The need for a large number of nodes geographically distributed to be the destinations of traceroute probes.
- The need to know the location of these nodes in order to maximize the amount of new information we could obtain.
- The need to identify the network interfaces that belong to the same IP device (alias resolution).
- A fraction of destinations may have IP addresses assigned by Dynamic Host Configuration Protocol (DHCP). The association between such an address and an actual host is temporary and random which makes the topology measurements to a DHCP address of little value.
- The difficulty to map IP paths behind firewalls or Network Address Translators (NATs).
- A router inflation may be also occurred due to the fact that some routers do not send ICMP messages back to the source. Such case appears as an empty line with the symbol star ($*$) in the traceroute output. Two stars belong to the same router if the upstream router, the downstream router, and the destinations are the same. Otherwise, it is difficult to recognize the stars belonging to the same routers. Thus, it may happen that few unknown routers in the actual topology be represented after traceroute aggregation by multiple unknown routers in the inferred one. Such router inflation problem can be seen in the example presented in Figure 2.4.

Next, we describe two interesting traceroute-based approaches, Skitter [34] and Rocktefuel [58], which have been proposed to infer the network topology.

**Skitter**

Skitter [59] is a widely used traceroute-based tool deployed by CAIDA [60] for actively probing the Internet in order to analyze topology and performance. It uses BGP tables and a database of Web servers to find destination prefixes. These prefixes are probed from about 20 skitter monitors geographically distributed. The main achievements of Skitter are the following:

- It records each hop from a source to many destinations.

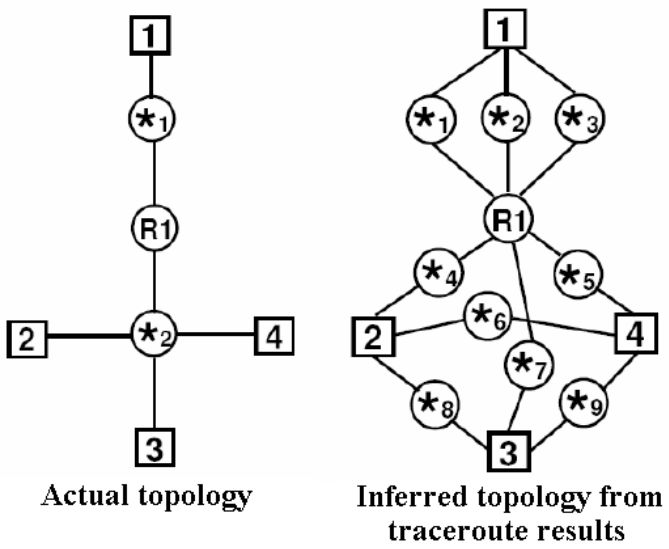



**Figure** 2.4: An example on router inflation after traceroute aggregation

- It collects round trip time (RTT) along with path (hop) data.
- It can provide indications of low-frequency persistent routing changes by determining the correlations between RTT and time of day.
- It can be used to visualize the directed graph from a source to a large part of the Internet. Figure 2.5 shows a sample visualization obtained from Skitter data.

For alias resolution, the skitter output packets holds the destination address in their last 4 bytes field. Basically, a router may transmit an echo reply via an interface that is different than the interface to which the echo request was sent, and may use the IP address of an outgoing interface as the source address in the echo reply. In this case, one can recognize that the source address and the address indicated in this last field belong to the same IP device. But, this technique is not efficient when for example the destination uses the same probed interface for transmitting the echo reply.

The best topology coverage by skitter monitors is far from complete. The largest lists of monitored destinations contain only about 800 thousands addresses; this is far from having one monitored destination in each /24 network (i.e., 256 IP addresses). Practically, there are over 16 million potential /24 segments in the IPv4 address space, and only about 4 million of them are currently routable.

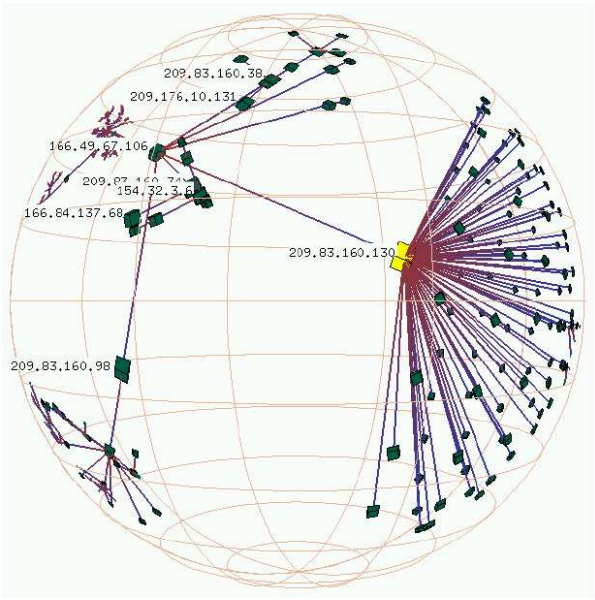



**Figure** 2.5: Sample visualization from Skitter data (taken from [60])

### Rocketfuel

Another solution, called Rocketfuel [58], infers the topology using traceroute probes, BGP routing tables, and DNS information. Rocketfuel aims to infer an ISP map using as few measurements as possible. The architecture of Rocketfuel (described in Figure 2.6) aims mainly to find out the useful prefixes to be probed for the discovery of an ISP topology, then to collect the aliases which belong to the same router. Next, we describe the main functionalities of this architecture.

A BGP table maps a destination IP address prefix to a set of ASes that are traversed to reach that destination. Thus, the use of BGP tables permits to pick up the prefixes whose traceroutes transit the ISP being mapped. Based on BGP tables, Rocketfuel defines three classes of traceroutes that transit the ISP network:

- Traceroutes to dependent prefixes which belong to the ISP or one of its customers. All traceroute probes to these prefixes from any vantage point should transit the ISP.

- Traceroutes from insiders which are traceroute servers located in a dependent prefix. Traceroutes from insiders to any prefix should transit the ISP.

- Traceroutes that are likely to transit the ISP based on the AS-path indicated in the

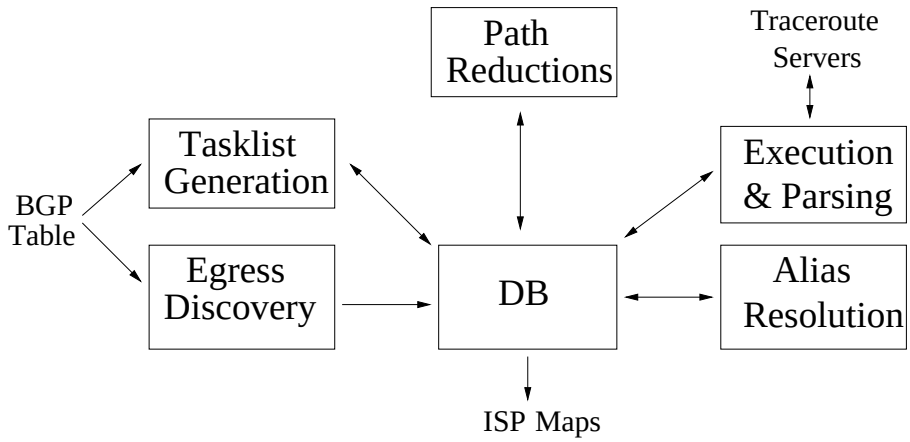



**Figure** 2.6: Architecture of Rocketfuel. The Database (DB) becomes the inter-process communication substrate.

BGP table.

Then, Rocketfuel filters these prefixes by suppressing those which are likely to follow redundant paths through the ISP network. Three techniques are used to reduce the redundant measurements:

- Ingress Reduction. When traceroutes from two different vantage points to the same destination enter the ISP at the same point, the path through the ISP is likely to be the same as shown in Figure 2.7a. In this case, only one probing is needed either from T1 or T2.

- Egress Reduction. Similarly, traces from the same ingress to any prefix behind the same egress router should traverse the same path as shown in Figure 2.7b. Thus, only one prefix is needed for being probed either P1 or P2.

- Next-hop AS Reduction. The path going through an ISP usually depends only on the next-hop AS, not on the specific destination prefix (see Figure 2.7c). Thus, only one trace from ingress router to next-hop AS is needed so as by probing either P1 or P2.

Using the described techniques, Rocketfuel finds that it can reduce the number of traces required to map an ISP by three orders of magnitude compared to the all-to-all approach.

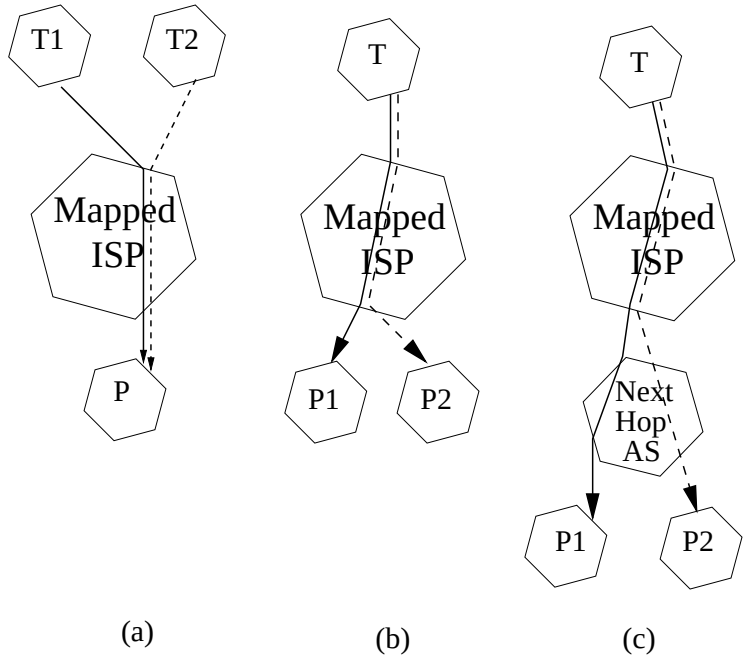



**Figure** 2.7: Path reductions

Also, Rocketfuel considers the alias resolution problem for assembling the IP interfaces listed in a traceroute into their corresponding routers. Rocketfuel's alias resolution tool, called Ally, depends on the Mercator technique [61]. The Mercator technique consists in sending traceroute probes to the potentially aliased IP addresses with a highly numbered UDP port, and a TTL equal to 255. If aliases belong to the same router, the router will respond by "UDP port unreachable" messages with the address of an outgoing interface as the source address. This technique is efficient in the sense that it requires only one message to each IP address but it misses many aliases. This can be caused by routers that have more than one outgoing interface and have answered from different interfaces for some reason (e.g., load balancing). Therefore, Ally sends the probe packets like in Mercator but with consecutive IP identifiers[1] as shown in Figure 2.8. The two "UDP port unreachable" responses include the identifiers x and y respectively. Then, Ally sends a third message to the address that responded first. Hence, if $x < y < z$, and $z - x$ is small, Ally considers that the IP identifiers most probably come from a single counter which means that the addresses are likely aliases.

[1]An IP identifier is designed to uniquely identify the portions of a packet for reassembly after fragmentation.

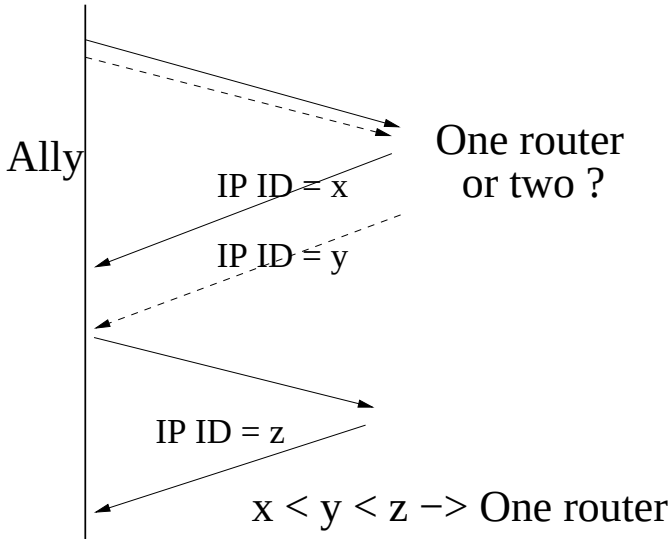



**Figure** 2.8: Alias resolution by IP identifiers. A solid arrow represents messages from an IP to an alias, the dotted arrow to another alias.

Rocketfuel relies on the DNS information to check up if an obtained router address belongs to the ISP being mapped. This information serves to identify the router role and location. Basically, each ISP has a naming convention for its routers as for example sl-bb11-nyc-3-0.sprintlink.net, which is a Sprint backbone (bb11) router in New York City (nyc).

### 2.4.3 Other active approaches

Many other approaches have been proposed in the literature to locate network nodes and estimate the distances on their joining paths based on partially observed measurements. In this section, we underline some well-known approaches that have been proposed in this field.

#### Binning method

One interesting approach for clustering network nodes is the binning method [26]. It consists in clustering the nodes into bins where a bin is identified by a specific increasing order of closeness to landmark nodes. Thus, each ordering of landmarks represents a bin (e.g., for $N = 4$, $L_2L_4L_1L_3$ represents a bin). A node measures its RTT to each

landmark and ranks the landmarks in an increasing order of RTT. Then, each node belongs to the bin which has the same ordering of landmarks. It is expected that the nodes of the same bin are relatively closer to each other than to those in the other bins. Moreover, it is expected that the more the order of two bins is different the farther are their nodes.

The representation of a bin can be refined to use not only the ordering of landmarks but also the absolute value of the RTT measurements. These values can be indicated by dividing the range of possible delay values into a number of levels. For example, the range of possible delay values can be divided into the three following levels: (i) level 0 for delays in the range [0,100]ms, (ii) level 1 for delays in the range [100,200]ms, and (iii) level 2 for delays longer than 200ms.

Thus, a bin can be identified by this notation: $L_iL_jL_k : a_ia_ja_k$ where $L_iL_jL_k$ is the relative ordering of landmarks and $a_ia_ja_k$ are the delay levels of each ordering respectively.

One potential application that can profit from such clustering method is the topologically aware construction of overlay networks. Based on the binning strategy, [26] constructs the overlay network CAN [63] (Content-Addressable Network) to be congruent with the underlying IP topology. CAN is an application-level network forming a virtual d-dimensional cartesian coordinate space. This space is partitioned among all the overlay nodes in a manner that each node is responsible of a space portion. Routing in CAN consists in sending packets along the direct line path through the cartesian space from the source to the destination coordinates as shown in the example drawn in Figure 2.9. Since the space is constructed using topology information, the time to route packets over the overlay nodes is expected to be minimized.

We briefly explain how the CAN overlay network can be constructed using the binning scheme. This scheme consists in dividing the space into the number of possible orderings of landmarks. Thus, there are N! equal sized portions when the number of landmarks is equal to N. Each portion belongs to a single ordering. The partitioning of the space is done in a cyclical ordering of the dimensions (e.g., xyzxyzx...). First, the space is divided into N portions along the first dimension. Then, along the second dimension, each portion is divided into $(N-1)$ portions each of which is divided to $(N-2)$ portions and so on. A CAN node determines its bin based on its RTT measurements to the landmarks. Then, the node joins the CAN in a random point of the correspondent portion of the coordinate space which corresponds to its landmark ordering. When a node joins a portion which is already assigned to another node, the

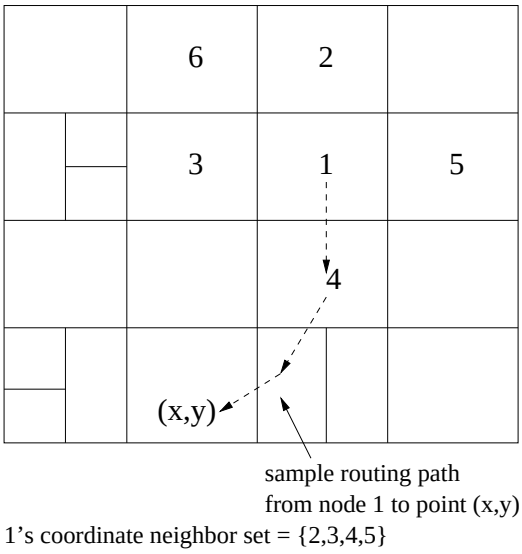



**Figure** 2.9: Example on routing in 2-d space

portion is divided equally between them. The problem is that this behavior may achieve an unequal dividing of the space among the overlay nodes. Even though the average number of network hops on the path between two nodes decreases when using such space partitioning, the load balancing between the nodes may be affected.

### Hotz method

Hotz [62] estimates the distance between two nodes based on the delay vectors (i.e., delay measurement to a set of reference nodes) of these nodes. It consists in bounding the distance $D(p_i, p_j)$ (or the delay) between two nodes $p_i$ and $p_j$ by two values which are the difference distance $|D(p_i, L) - D(L, p_j)|$ and the summation distance $(D(p_i, L) + D(L, p_j))$ as shown in the following equation:

$$|D(p_i, L) - D(L, p_j)| < D(p_i, p_j) < (D(p_i, L) + D(L, p_j)) \tag{2.4}$$

where $D(p_i, L)$ and $D(L, p_j)$ are the distances between a landmark $L$ with nodes $p_i$ and $p_j$ respectively. Thus, if there are $N$ landmark nodes then the delay between two nodes $p_i$ and $p_j$ is lower bounded by $max|D(p_i, L_k) - D(L_k, p_j)|$ and upper bounded by $min|D(p_i, L_k)+D(L_k, p_j)|$ where $k$ vary from 1 to $N$ as shown in the following equation:

$$max_{k=1...N}|D(p_i,L_k)-D(L_k,p_j)| < D(p_i,p_j) < min_{k=1...N}(D(p_i,L_k)+D(L_k,p_j)). \quad (2.5)$$

Thus, the distance between the nodes $p_i$ and $p_j$ can be estimated as the average value of the two bounds. Clearly, such approach assumes that the network distances satisfy the triangle inequality which may not be consistent with the real network topology [55, 57].

### IDmaps scheme

IDmaps [27](Internet Distance Map Service) is an architecture proposed for estimating the delay among network nodes. It consists in disposing of a few hundreds or thousands of *tracer* nodes which form a set of infrastructure nodes. The tracers measure the delay among each others. They also measure the delay to every IP address prefix and determine the closest tracer to each prefix. The delay between two hosts $h_1$ and $h_2$ is estimated as the delay from the prefix of $h_1$ to its closest tracer, plus the delay from the prefix of $h_2$ to its closest tracer, plus the delay between the two tracers.

### Meridian approach

Meridian [8] consists of an overlay network structured around multi-resolution rings. Meridian determines the delay closest node to a given target using a distributed algorithm. It consists in performing a multi-hop search where each hop exponentially reduces the distance to the target. This is inspired from peer lookup algorithms, as Chord [64], Pastry [65], and Tapestry [66], which are proposed to reduce the content lookup time in structured peer-to-peer networks. In such peer-to-peer structures the query is transmitted in each hop exponentially closer (in terms of numerical distance) to the destination. In Meridian, the delay metric is used instead of numerical distance to perform the routing.

Figure 2.10 shows how Meridian can discover the closest node to a given target. When a Meridian node receives a request to find the closest node to a target, it determines its delay to the target. Once this delay is determined, it probes its ring members within a well-defined interval to determine their distances to the target. Then, the request is forwarded to the discovered closest node, and the process continues until no closer node is detected.

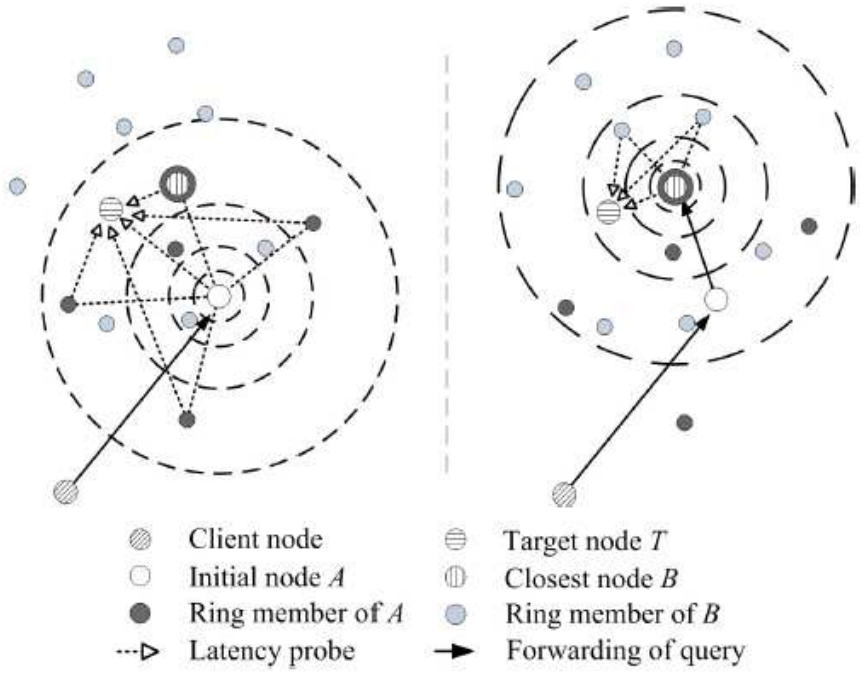



**Figure** 2.10: An example describing the closest node discovery achieved by Meridian (taken from [8])

### Geolocation approaches

Many works have been proposed for inferring the geographical location of network nodes. We mention the following two interesting ones:

- Constraint-Based Geolocation (CBG) approach [28] infers the geographical location of network nodes using multilateration. It consists in estimating the geographical position of a node using its geographical distances to a set of landmarks. Geographical distances to the landmarks are deduced from the correspondent delay distances (obtained by direct probing between the landmarks and the target host) by relying on the assumption that digital information travels along fiber optic cables at almost exactly 2/3 the speed of light in a vacuum. Basically, given the geographical locations of the landmarks and their geographical distances to a given target host, an estimation of the location of the target host is achieved using multilateration, as done with the Global Positioning System (GPS) [67].

- IP2Geo system [29] estimates the physical location of a remote server using information from the content of DNS names, whois queries, pings from fixed locations,

and BGP information.

## 2.5 Conclusions and Perspectives

The continuous evolution of the Internet and its distributed administration make the inference of its topology more and more complex. Inferring the topology of the Internet becomes a wide research domain which is of particular interest for network operators and users. It is necessary for controlling the network, tuning its protocols, and enhancing the performance of its applications. We presented in this chapter an overview on the existing schemes proposed for inferring actively the network topology. We are interested particularly by the schemes that locate network nodes using a partial set of measurements (e.g., coordinate-based approaches, Rocketfuel).

We have realized that most of the existing schemes use simple metrics (e.g., delay, geographical distance) for characterizing the proximity among network nodes. We believe that these metrics are not enough to characterize the proximity for many applications (e.g., transfer of large files). The proximity must be defined at the application level taking into consideration the network metrics that decide on the application performance. To this end, we propose in this thesis an approach for characterizing the proximity according to utility functions that model the quality perceived by end users at the application level.

The deployment of such proximity model requires the estimation of the network parameters in an easy and scalable way. Existing coordinate-based approaches are frequently used for estimating scalably the delay among network nodes. Such approaches suffer particularly from the fact that they provide distances that are symmetric and satisfy the triangle inequality which may not be consistent with the real network topology. These limitations can be avoided by using the matrix factorization approach which permits to infer the network delay in a way that is able to model the sub-optimal routing and asymmetric routing policies. We study in this thesis this powerful model and test its capacity to estimate the delay, the loss rate and the available bandwidth.

Our experiments show that while the matrix factorization approach provides accurate estimations for the delay and loss rate parameters, it is not the case for the available bandwidth. This is because the latter parameter is not an additive metric and it rather depends on the bottleneck link that may appear anywhere along an end-to-end path. This means that network embedding approaches are not expected to be

the appropriate tools for estimating the bandwidth parameter. Therefore, we propose in the last part of this thesis a scalable model for estimating the end-to-end available bandwidth.

# 3

# MOTIVATING THE PROXIMITY IN CHESS

## 3.1 Summary

We introduce in this chapter *CHESS*, an application-aware space for enhanced scalable services in overlay networks. In this new space, the proximity of peers is determined according to a utility function that considers the critical performance parameters (e.g., on both network paths and peers sides) impacting the application performance; not simply the delay as done in existing approaches. In this chapter, we motivate the need for CHESS by showing that path characteristics are not highly correlated, and so the proximity in one space, say for example the delay, does not automatically lead to a proximity in another space as the bandwidth one. The challenge of determining the proximity in the CHESS space is to estimate the network parameters, that impact the utility function, in an easy and scalable way.

## 3.2 Introduction

In Peer-to-Peer and overlay networks, the quality of service perceived by end-users can be optimized at the application level by identifying the best peer to contact or to take as neighbor. This requires to define a proximity function that evaluates how much two peers are close to each other from application point of view.

Different functions are introduced in the literature to characterize the proximity of peers, but most of them [9, 23, 25, 28] are based on simple metrics such as the delay, the number of hops or the geographical location. We believe that these metrics are not enough to characterize the proximity given the heterogeneity of the Internet in terms of path characteristics and access link speed, and the diversity of application requirements. Some applications (e.g., transfer of large files) are sensitive to other network parameters such as the bandwidth.

Therefore, the proximity should be defined at the application level taking into consideration the network metrics that decide on the application performance. To this end, we introduce the notion of CHESS space. In this new space, the proximity is characterized according to a utility function that models the quality perceived by peers at the application level. A peer is closer than another one to some third peer if it provides a better utility function, even if the path connecting it to the third peer is longer.

Based on extensive measurements over the Planetlab overlay network [69], we motivate the need for our new proximity notion by studying how much a proximity-based ranking of peers using the delay deviates from that using other network parameters (e.g., available bandwidth, loss rate). Our main observation is the following. Delay, bandwidth and loss metrics are slightly correlated, which means that, in our setting, one cannot rely on one of these metrics in defining proximity when the application is more sensitive to the others. Particularly, the best peer to contact is not always the closest one. Therefore, the knowledge of the different network parameters between peers helps in improving the performance of applications by allowing the definition of better proximity models.

The chapter is structured as follows. Next, we present our measurement setup. In Section 3.4, we study the correlation of the different measured network characteristics. Then, we introduce in Section 3.5 some potential applications that can profit from characterizing the proximity in CHESS. Finally, the chapter is concluded in Section 5.6.

## 3.3 Data setup and notations

Our experiments consist of real measurements run over the Planetlab platform [69]. We do not claim that this platform is representative of all networks, but we believe that it is the best evaluation testbed available nowadays that satisfies the large scale requirement of our study. Moreover, this platform has proved its capacity to be appropriate for

measurements [68].

We measure the end-to-end network parameters of the paths connecting Planetlab nodes using the *Abing* tool [70]. This tool is based on the packet pair dispersion technique [71]. It consists in sending a total number of 20 probe packet-pairs between the two sides of the measured path. Abing has the advantage of short measurement time on the order of the second, a rich set of results (e.g, bandwidth in both directions), and a good functioning over Planetlab. The measurement accuracy provided by Abing on Planetlab is quite reasonable compared to other measurement tools [72]. All these features make the Abing tool appropriate for our study.

Our experiments on Planetlab consist in the following measurement sets. We take 127 Planetlab nodes spread over the Internet and covering America, Europe, and Asia. Forward and reverse paths between each pair of Planetlab nodes are considered, which leads to 16002 measurements. These measurements are repeated three times during the last week of February 2005. For each unidirectional path between two Planetlab nodes, we measure the round-trip time *RTT*, the available bandwidth *A*, the bottleneck bandwidth (or capacity) *BC*, and the packet loss rate *P*. *P* is estimated as the ratio of the number of lost packets and sent packets. *BC* is the speed of the slowest link along the path. We recall that available bandwidth means the remaining bandwidth left on a path between any two network nodes. It is determined by the link having the smallest residual bandwidth[1].

In our discussion, along the thesis, we consider a *server* as being either a server among a set of replicated servers in a CDN or a peer in a peer-to-peer network. The *best server* is the server which is able to provide the requested service to the client (peer) with a better QoS than all other servers. The *central server* is the resolver (or the redirector) in the overlay network. It receives requests from clients (peers) and provides them back the address(es) of the best server(s) (best peer(s)). Also, we mean by *client* a standard client in the client/server paradigm, or a peer that requests a content in a peer-to-peer network. Clearly, the best server varies from one client to another based on the position of the client and the state of networks and servers. Besides, we almost refer to the Planetlab nodes by the term peers when describing our experimentation results. Moreover, for any of the network metrics, say X, we denote by $X(p_i, p_j)$ the value of the metric associated to the path starting from peer $p_i$ and ending at peer $p_j$.

[1] In the rest of the thesis, we refer to this link by the term *tight* link.

## 3.4 Motivating CHESS

Different definitions were studied in the literature for characterizing the proximity among peers, and hence for selecting the appropriate peer to contact. These definitions can be classified into two main approaches static and dynamic. The difference between these approaches lies in the metric they consider. Static approaches [28, 29] use metrics that change rarely over time as the number of hops, the domain name and the geographical location. Dynamic approaches [9, 29, 23] are based on the measurement of variable network metrics. They mainly focus on the delay and consider it as a measure of closeness of peers; the appropriate peer to contact is often taken as the closest one in the delay space. The focus on the delay is for its low measurement cost (i.e., measurement time , amount of probing bytes). However, its use hides the implicit assumption that the path with the closest peer (in terms of delay) has the minimum (or relatively small) loss rate and the maximum (or relatively large) bandwidth.

While we believe that the delay can be an appropriate measure of proximity for some applications (e.g., non greedy delay sensitive applications or those seeking for geographical proximity), it is not clear if it is the right measure to consider for other applications whose quality is a function of diverse network parameters. Bandwidth greedy applications and multimedia ones are typical candidates for a more enhanced definition of proximity. To clarify this point, we use our measurements results and study the correlation among path characteristics. We want to check whether (i) the characteristics are correlated with each other, and (ii) how much a proximity-based ranking of peers using the delay deviates from that using other path characteristics.

As we will see in this section, there is a clear low correlation among path characteristics which motivates the need for an enhanced model for proximity. The closest peer in terms of delay is far from being optimal in the bandwidth space or loss rate space, and vice versa.

### 3.4.1 Delay vs. bottleneck capacity

Take a peer $p$ and let $p_d$ be the closest peer to $p$ in terms of delay and $p_b$ the best peer from $p$'s point of view in terms of bottleneck capacity. First, we want to study how much the bottleneck capacity on the path connecting $p$ to $p_d$, $BC(p, p_d)$, deviates from the largest one measured on the path between $p$ and $p_b$, $BC(p, p_b)$. Figure 3.1(a) draws the complementary cumulative distribution function (CCDF) of

the ratio $BC(p, p_d)/BC(p, p_b)$. The curve is calculated over all peers and for each data set. For a value $x$ on the x-axis, the corresponding value on the y-axis gives the percentage of peers having on their path to the nearest peer a bottleneck bandwidth larger than $x$ times the maximum bottleneck bandwidth.

The figure shows that only around 5% of peers have the maximum bottleneck capacity on their path to the nearest peer. Moreover, a large percentage of peers, around 80%, have less than 20% the maximum bottleneck capacity on this path. This indicates that selecting the best peer in terms of delay leads in most cases to a bottleneck capacity far from the optimal. Applications having a high bandwidth requirement could suffer from this choice.

Now, we generalize our results to the other peers than the closest one. We plot in Figure 3.1(b) the bottleneck capacity versus the round trip time that have been measured three times over the 16002 paths. This means that the figure contains 48006 measurements for this couple of parameters. The Figure shows that *BC* does not decrease uniformly when *RTT* increases. Furthermore, the correlation coefficient [73] between these two variables is small and equal to $-0.068$[2].

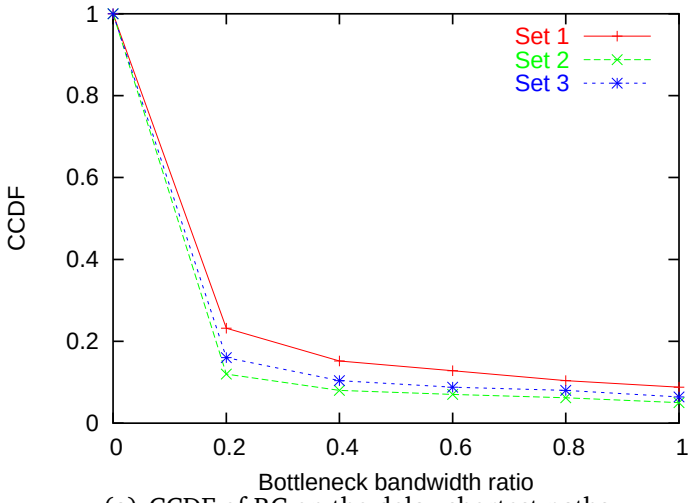
Figure 3.1(c) plots the average bottleneck capacity for all peers of rank $r$ in the delay space, $r$ varying from 1 to 126. In other words, for each peer among the 127 peers, we take its neighbor of rank $r$ in the delay space, we measure its bottleneck capacity, then we average this bandwidth over the 127 peers. This is done for the three data sets. Again, the figure shows a slow decrease of the bottleneck capacity with the delay-based peer rank. The delay closest peers are far from yielding the best bottleneck capacity.

### 3.4.2 Delay vs. available bandwidth

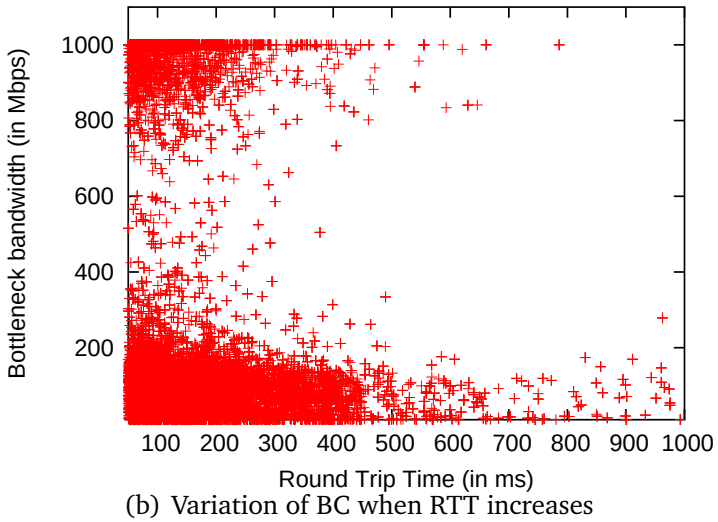
We repeat the same analysis but this time for the delay and the available bandwidth. For a peer $p$, we denote by $p_a$ the best peer from $p$'s point of view in terms of available bandwidth. Figure 3.2(a) shows the CCDF of the ratio $A(p, p_d)/A(p, p_a)$. It illustrates how far is the available bandwidth on the delay shortest path from the optimal available bandwidth. The figure is plotted over the 127 peers and for each data set.

We can see that a small percentage of peers, between 5% and 15%, have the maximum available bandwidth on their path with the nearest peer. A large proportion of peers, around 80%, have less than 50% of the maximum available bandwidth on this

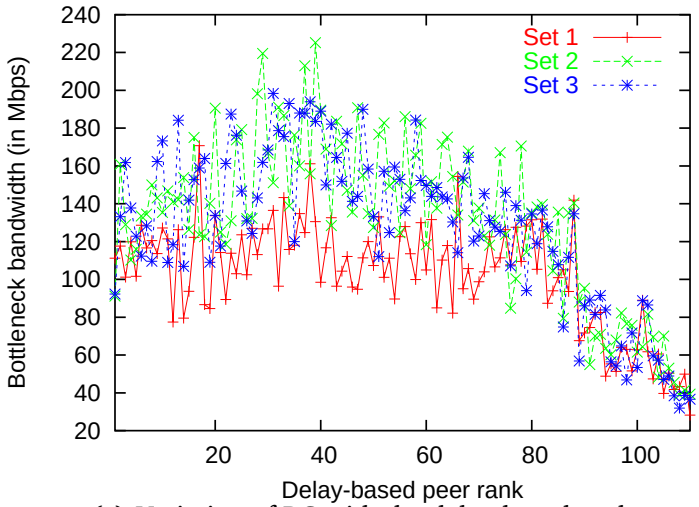
[2]One would have expected a coefficient closer to -1 than to 0.




(a) CCDF of BC on the delay shortest paths




(b) Variation of BC when RTT increases




(c) Variation of BC with the delay-based rank

**Figure** 3.1: Delay versus bottleneck bandwidth

path. Even though these numbers are better than in the bottleneck bandwidth case, the delay is still far from being the proximity metric to use to detect the peer with the maximum available bandwidth.

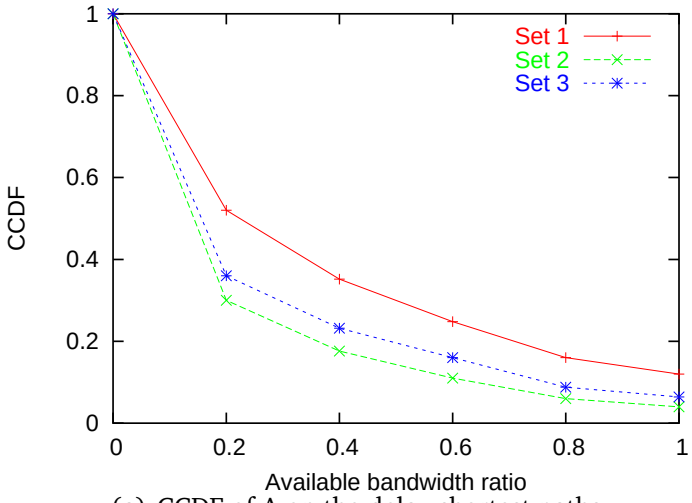
Figure 3.2(b) plots the available bandwidth versus the round trip time for the total 48006 measurements achieved over the 16002 paths. There is no strong correlation between $A$ and $RTT$. In our setting, the two variables are lightly negatively correlated with a coefficient equal to $-0.01$. Similar result can be observed in Figure 3.2(c) where we plot the available bandwidth for peers of rank $r$ in the delay space, $r$ varying from 1 to 126, averaged over the 127 peers. The figure is plotted for the three data sets. We note that looking at farther and farther peers in the delay space does not lead to an important decrease in the available bandwidth, and so there is a high chance of having the optimal peer from bandwidth point of view located far away (in the delay space) from the peer requesting the service.

### 3.4.3 Bottleneck vs. available bandwidth

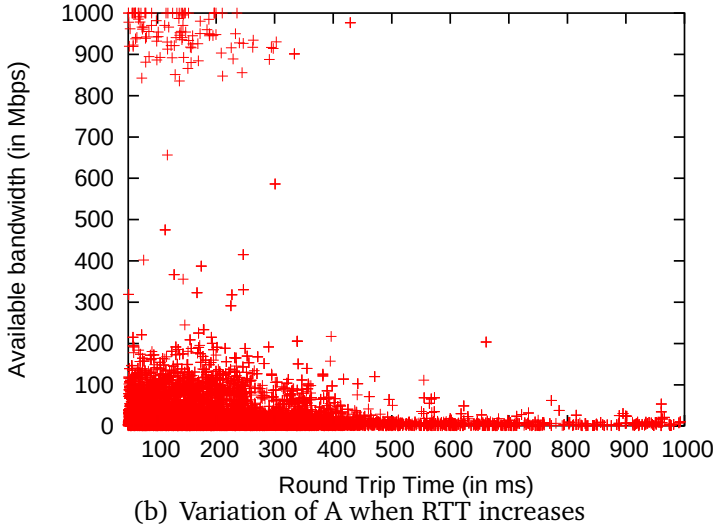
The bottleneck capacity provides an indication on the maximum performance one can achieve. The available bandwidth indicates how much the network is loaded. It is linked to the bottleneck capacity, but since Internet paths are differently loaded, there should be no reason to think that these two characteristics can replace each other when defining proximity for applications sensitive to the bandwidth. This is what we analyze in this section.

For a peer $p$, we plot in Figure 3.3(a) the percentage of peers having a ratio $A(p, p_b)/A(p, p_a)$ larger than $x$, $x$ between 0 and 1. This is calculated and drawn for each data sets. In other words, we check the difference between the available bandwidth on the path having the maximum bottleneck bandwidth and the maximum available bandwidth. The figure shows that around 10% of peers have the maximum $A$ on the path having the best $BC$. Moreover, a large percentage of peers, around 80%, have less than 60% the maximum $A$ on the path having the best $BC$. Clearly, selecting the peer with the maximum bottleneck capacity is not equivalent to selecting the one with the maximum available bandwidth, and the error is not negligible.

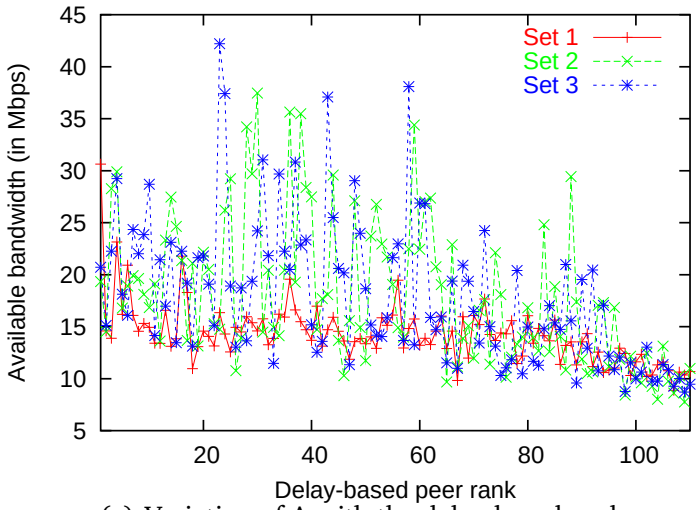
Then, we study how these two characteristics behave over all peers. We plot in Figure 3.3(b) the available bandwidth versus the bottleneck bandwidth for the total 48006 measurements. A positive correlation can be seen, which when computed, yields a coefficient equal to 0.36. Figure 3.3(c) plots the available bandwidth averaged over




(a) CCDF of A on the delay shortest paths




(b) Variation of A when RTT increases




(c) Variation of A with the delay-based rank

**Figure** 3.2: Delay versus available bandwidth

all peers having the rank $r$ in the decreasing-order of the bottleneck bandwidth space. The figure shows the results obtained in our three data sets. Clearly, the farther a peer in the bottleneck space, the smaller the available bandwidth. But, in spite of this correlation, we suggest not to replace these two metrics in the proximity definition when the application requires one of them. Both need to be considered simultaneously for the proximity definition to be efficient.

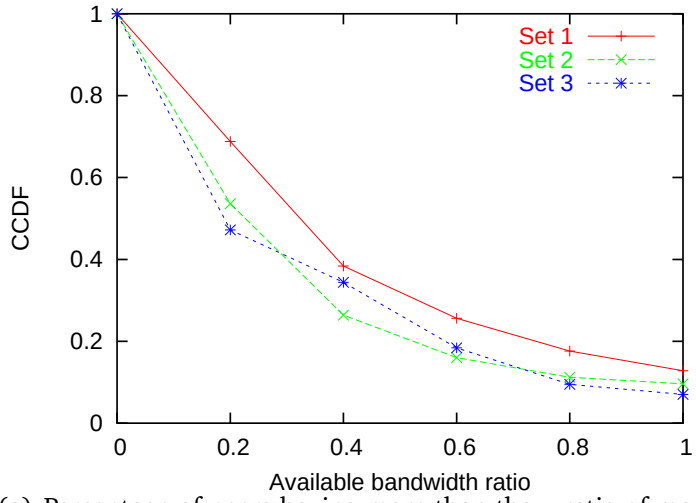
### 3.4.4 Delay vs. loss rate

Applications are sensitive to the loss rate. We want to check in this section how well a definition of proximity based on delay satisfies the loss rate. We find that all peers have, in our setting, a null loss rate ($P = 0$) on their path with at least one other peer. To check whether the nearest peer results in the minimum loss rate (i.e., zero), we plot in Figure 3.4(a) the CDF of the loss rate on the path connecting a peer $p$ to its nearest peer $p_d$. The distribution is computed over the 127 peers and for each data set. We can see that around 89% of peers have the minimum loss rate ($P = 0$) on their path to the nearest peer. Our measurements show that the delay and loss rate are positively correlated with a coefficient equal to 0.38. Another observation we made is that as long as we move away from a peer in the delay space, the loss rate jumps to values on the order of several percents, then it increases slowly. This is well illustrated in Figure 3.4(b) where we plot the packet loss rate on the path connecting a peer to its neighbor of rank $r$ in the delay space, $r$ changing from 1 to 126. The figure is averaged over the 127 peers and plotted for the three data sets.

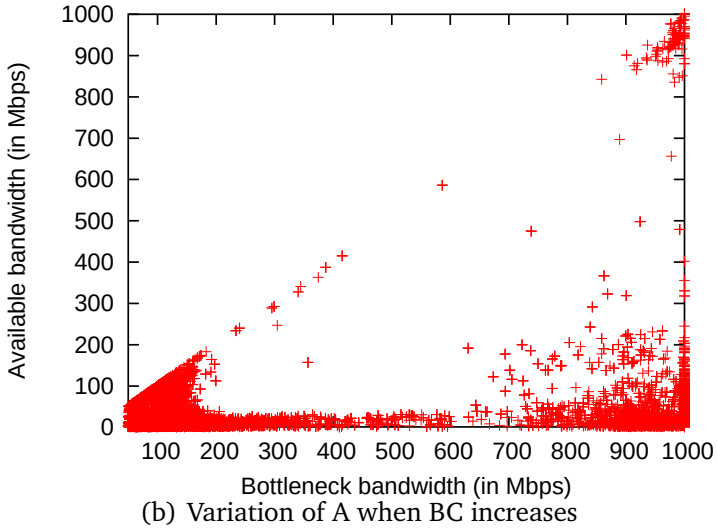
In summary, the closest peer seems to give the minimum loss rate in most cases. The reason could be the fact that both are located in a non-congested neighborhood. Now, when it comes to selecting more than one peer for a certain service sensitive to the loss rate, taking the delay as a metric of proximity stops being efficient, and the loss rate has to be considered as well.

### 3.4.5 Load vs. loss rate

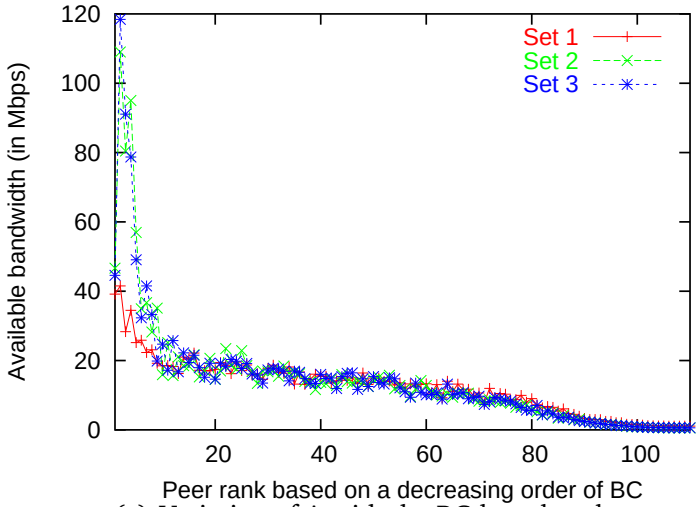
Finally, we check the correlation between the network load ($\rho = 1 - A/BC$) and the loss rate. Surprisingly, we find these metrics to be lowly correlated, with a coefficient




(a) Percentage of peers having more than the x ratio of maxA on their maxBC paths

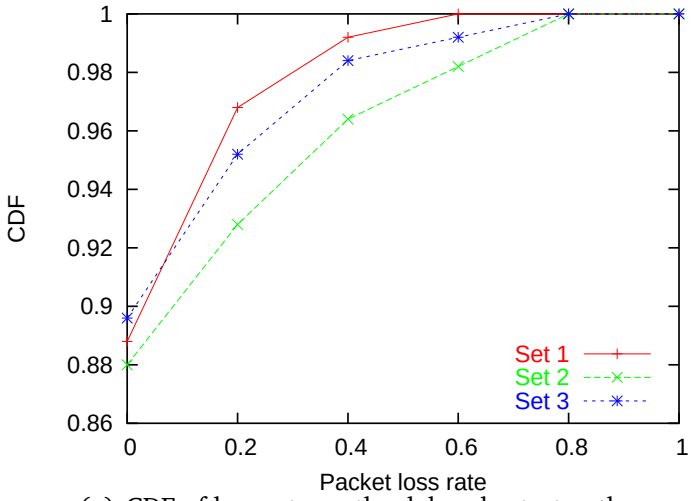



(b) Variation of A when BC increases

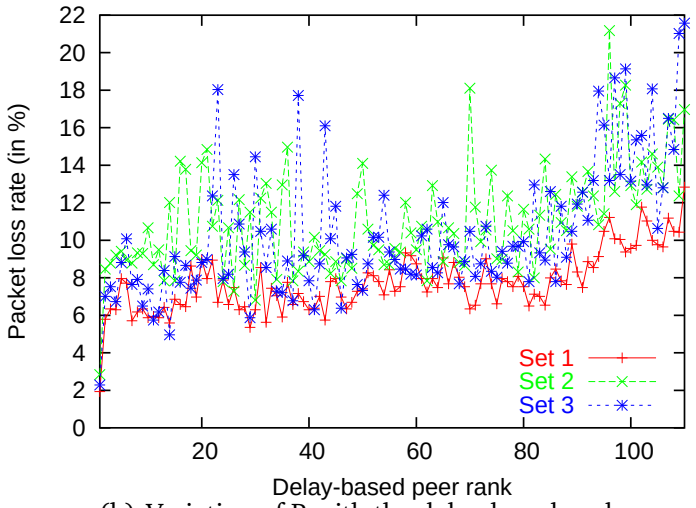



(c) Variation of A with the BC-based rank

**Figure** 3.3: Bottleneck bandwidth versus available bandwidth




(a) CDF of loss rate on the delay shortest paths




(b) Variation of P with the delay-based rank

**Figure** 3.4: Delay versus loss rate

of correlation equal to $0.0277$ in our setting[3]. As in the delay case, we want to check whether the loss rate is satisfied if one takes the load as a proximity metric. Let $p_\rho$ be the peer with the minimum load. We plot in Figure 3.5(a) the distribution of $P(p, p_\rho)$ computed over the 127 peers and for each data set. The figure shows that around 66% of peers have the minimum loss rate ($P = 0$) on their lowest loaded path. Also, the major percentage of peers , around 80%, have a loss rate smaller than $0.1$. We complete the analysis by plotting in Figure 3.5(b) the packet loss rate as a function of the peer rank in the load space. This is averaged over the 127 peers and plotted for each data set.

The observation we can make from these figures is that load and loss rate are not highly correlated, and so they need to be both considered simultaneously for an efficient proximity definition (if the application requires both of them).

## 3.5 Introducing CHESS

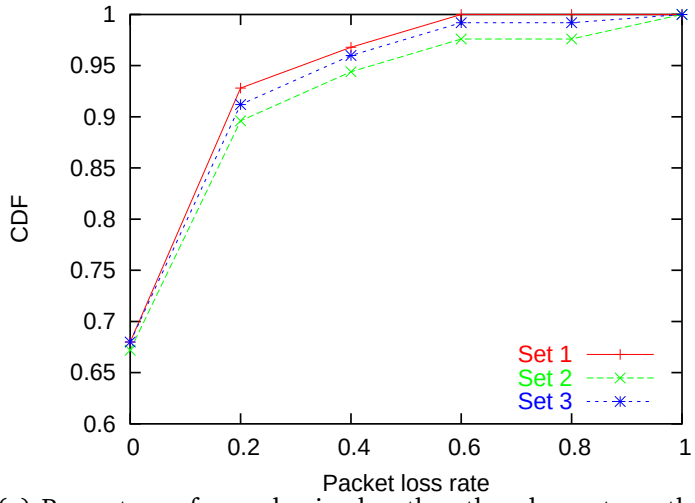
### 3.5.1 Goal

The weak correlation among path characteristics pointed out by our measurements motivates us to introduce the proximity in CHESS. It consists in determining the proximity at the application-level by estimating some utility function that models the application quality such as the transfer time for file transfer applications. Peers are ranked with respect to each other using the values given by the utility function for the paths joining them. A peer is closer than another one to some reference peer if it provides a better utility function, even if the path leading to it is longer.

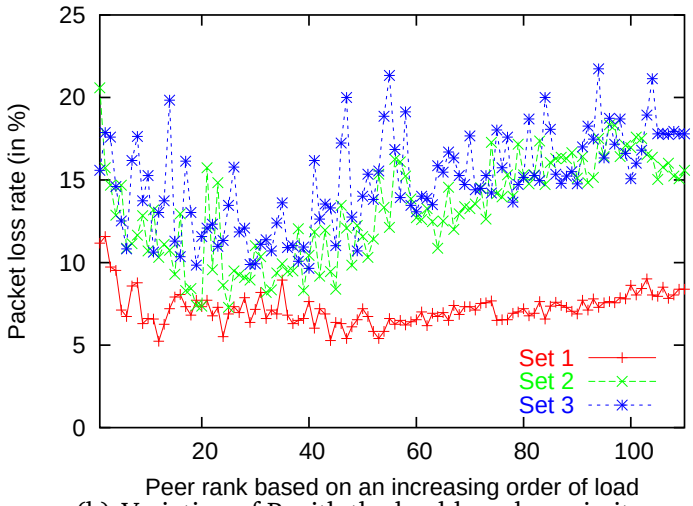
### 3.5.2 Potential applications

Such proximity characterization can be useful for enhancing the quality of a plenty of applications. Particularly, it can be used for server selection and overlay construction. Next, we first describe an example scheme that aims to enhance the quality of service perceived by the client (resp. file transfer time, speech quality) by providing to him the address(es) of the best server(s) after ranking the replicated servers from the best one to the worst one based on some utility function (resp. transfer time prediction,

[3] One would have expected these two metrics to be strongly correlated with a coefficient of correlation closer to 1 than to 0.




(a) Percentage of peers having less than the x loss rate on their minimum loaded paths




(b) Variation of P with the load-based proximity

**Figure** 3.5: Load versus loss rate

speech rating factor). Then, we describe briefly how our new notion of proximity can be efficient for constructing overlay networks.

**Server selection**

Service replication is a scalable solution for the distribution of digital content over the Internet. The need for this replication is caused by the increasing number of Internet users and by the desire to improve the QoS. Also, it is important for achieving a high availability of the service. Many overlay networks (e.g., Content Distributed Networks (CDN), and peer-to-peer networks) are proposed and installed to realize this replication.

Many policies have been studied in the literature for best server selection. The mostly used approaches can be classified to the following three categories:

- Using the DNS (Domain Name System) to get the IP address of the best server. This widely used technique is simple: the DNS servers distribute the IP addresses of multiple servers associated to a unique name with a round robin algorithm. It is clear that this solution is not designed to improve the QoS since it does not consider any static or dynamic performance limitations. It only ensures basic load balancing.

- Offering the client a list of servers and let him choose manually the best server to contact. The client choice in this case is based on his own criteria, for example the geographical proximity.

- Choosing the closest server in terms of delay. Inferring the delay closeness between client and servers can be done using one of the scalable approaches presented in the previous chapter. For example, this can be done using the coordinate vectors of client and servers. The coordinate of each node can be determined using one of the network embedding approaches (see the previous chapter for further details). Also, the closeness can be determined by identifying the bin of the client and each server. This can be done by measuring their RTT (Round-Trip Time) to a set of landmark points as described in the previous chapter. By knowing the bins of the client and servers, the DNS server can classify the servers (from the best one to the worst one) based on the distance between their bins and the client's one.

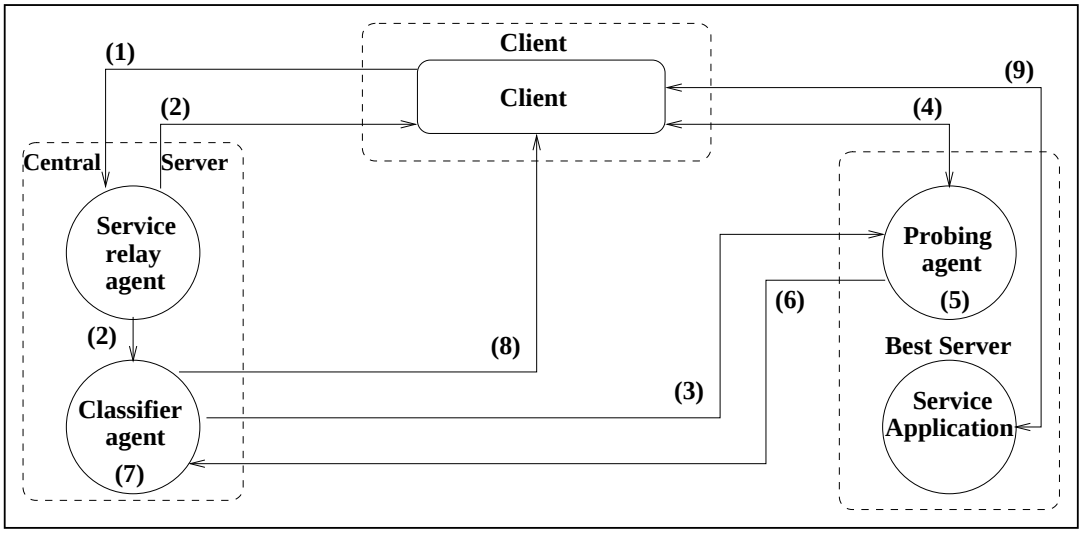



**Figure** 3.6: An example scheme for best server selection

Thus, most of the existing solutions for best server selection are based on simple metrics such as the delay, and the geographical location which are uncorrelated with other network characteristics (e.g., available bandwidth, loss rate); as we have obtained in Section 3.4. Hence, these metrics are not enough to characterize the proximity given the heterogeneity of the Internet in terms of path characteristics and access link speed, and the diversity of application requirements.

We realize that the proximity must be characterized at the application level taking into consideration the network metrics that decide on the application performance. We propose to do that using a utility function that models the quality perceived by peers at the application level. In this framework, a peer is closer than another one to some third peer if it provides a better utility function, whatever the position of each peer in the geographical and delay spaces.

For selecting the best server based on the proximity in CHESS, we present the example scheme described in Figure 3.6. The figure shows the following set of agents implemented in the central and replicated servers:

- Each replicated server holds
  - *Load-estimator* agent which measures continuously the load on the server,
  - *Probing agent* which probes the client and predicts the application quality that can be achieved between the client and the servers, and

  - *Service application* which provides the client with the requested service if the server has been selected by the central server as being the best one.

- The central server holds

  - *Service relay agent* which is a well-known application-level gateway, and
  - *Classifier agent* which ranks the servers, based on the predicted quality values, to provide back to the client the IP address(es) of the best one(s).

Next, we describe how this architecture can be efficient for selecting the best server for two typical applications which are file transfer over TCP and interactive audio service.

*1) File transfer over TCP:* Take the case where the service consists of clients downloading files from a set of replicated servers using the TCP protocol and where the QoS provided to clients is maximized if the transfer time is minimized. In this case, choosing the best server amounts to downloading the file from the server that is able to provide the minimum transfer time. This improves the QoS provided to clients and avoids network and server congestion by distributing the load over servers and network paths that are less loaded than others.

Before establishing the connection between the client and the best server for content download, one possible way to characterize the proximity in CHESS can be realized in the following way (as described in Figure 3.6):

1. The client sends its request to the service relay agent in the central server.
2. The service relay agent calls the classifier agent and gives it the client's IP address and client receiving buffer space (obtained from the window size advertised by the client in the TCP header).
3. The classifier agent sends the client's IP address and client receiving buffer space to the probing agents located in a certain set of replicated servers. For example, to serve a French client, the classifier agent sends the client information to the probing agents located in the replicated French servers.
4. Each probing agent in a replicated server probes the client in order to obtain the performance on the path between them, and reads the current server performance parameters determined by the load estimator agent.

5. Each probing agent computes the predicted transfer time metric defined in Section 4.3.1.

6. Each probing agent sends the obtained value of the predicted transfer time to the classifier agent in the central server

7. The classifier agent compares the predicted transfer time values received from the different probing agents and sorts these values in an increasing order.

8. The classifier agent sends to the client the IP address(es) of the best server(s) (which provide the smallest transfer time value).

9. A TCP connection is established between the client and the best server. In case a set of IP addresses is provided as corresponding to the best servers, the client can open a TCP connection with each of these servers and download the content in parallel.

In the next chapter, we develop a metric predicting the content transfer time between client and server. Then, we evaluate the enhancement perceived by clients when they choose the best server(s) based on the predicted transfer time instead of the delay-based one.

*2) Interactive audio service:* Replicating servers can be also done in the scope of interactive audio (i.e., VoIP) service. In such case, a set of replicated servers are distributed in the network to provide clients with the same audio communication. To serve a client, a central unit is in charge of identifying the server that can provide the best speech quality. As in the case of file transfer, this can be done by considering the load on the servers and the performance parameters on the network paths joining clients to servers. In the next chapter, we describe how one can predict the speech quality subjectively using the E-model defined by ITU-T G107 [74]. Then, we evaluate the enhancement of the performance perceived by the clients when they choose the best server based on the predicted speech quality instead of the delay-based one.

*3) Discussion:* The described scheme proposes direct probing between clients and servers to estimate the network parameters on their joining paths. In this case, the implementation of the probing agent on the server side aims to make the service transparent to clients. Another reason is that the probing agent can easily estimate the load on the server without sending to the network any control message. However, when characterizing the proximity in a large scale network, the direct probing between

clients and servers must be avoided. In this case, the probing agent located in the replicated server side must be substituted by another mechanism which is able to estimate scalably the network parameters on the paths between client and servers.

#### Overlay construction

The characterization of the proximity in CHESS can be useful for enhancing the quality for plenty of other applications as well. Take the case of constructing an overlay multicast tree for video streaming. In such case, peers can be distributed along the tree not simply based on their geographical location or their delay closeness, but also by considering other application specific requirements. Particularly, peer's fan-in (i.e., maximum bitrate that can be received by peer) and fan-out (i.e., maximum bitrate that can be sent by peer) bandwidth limitations can be taken into account when assigning to the peer a position in the tree.

The delay, the available bandwidth, and the packet loss rate parameters on the network paths joining peers can be also considered along with peers' bandwidth limitations for constructing the overlay tree. The difficulty behind the realization of such overlay construction appears in large scale networks where network parameters must be estimated in an easy and scalable way.

### 3.5.3 Challenges

The characterization of the proximity in CHESS is a challenging task. It requires the identification of the appropriate utility function for each application in a first stage. Then, the measurement of the different network parameters that impact the utility function. This is difficult to achieve in large scale networks where the number of peers can be huge. In such case, the cost of the direct probing among peers may outweigh the profit of the characterized proximity. Hence, we underline the two following challenging tasks that have been investigated in this thesis for characterizing the proximity in CHESS:

- The determination of utility function predicting the application quality perceived by clients (e.g., transfer time prediction). This requires to identify the critical parameters impacting the application performance. Take the case of file transfer applications over the TCP protocol. In such particular case, one may consider

when modeling the transfer time prediction function the following critical network parameters:

- The characteristics of the paths between client and server: the available bandwidth, the round-trip time, and the packet loss rate.
- The performance limitations of the server: the maximum congestion window value that can be buffered for transmission, and the idle time lost due to the buffering of the requests in the server (time between the arrival of a request and the establishment of the corresponding TCP connection). This idle time depends on the server load.
- The performance limitation of the client, which is represented by its receiving buffer space that can be communicated by the advertised window field in the TCP header.

We note that the indicated limitations are considered, in Chapter 4, when developing the metric predicting the content transfer time from a server to a client.

- The estimation of the network parameters, impacting the utility function, in an easy and scalable way. In other terms, this should be achieved with a small measurement overhead and a limited cooperation among nodes. Particularly, the determination of the network parameters, on the paths joining a large number of peers, must be achieved by avoiding the direct probing among them. This issue is explored in Chapter 5.

## 3.6 Conclusions

We introduce in this chapter the concept of application-aware space that we call CHESS. The characterization of the proximity in this space relies on the determination of the path characteristics and application requirements. The need for this notion of proximity is deduced from the low correlation observed among path characteristics. Particularly, a proximity in the delay space does not automatically lead to a proximity in another space as the bandwidth one. The challenge of determining the proximity in the CHESS space is to estimate the network parameters, that impact the utility function, in an easy and scalable way. In the next chapters, we focus on the deployment of this

new definition of proximity and on the evaluation of its gain with different application types.

# 4

# APPLICATION QUALITY ENHANCEMENT DUE TO THE PROXIMITY IN CHESS

## 4.1 Summary

In this chapter, we wonder whether the delay proximity is a good approximation of the proximity in CHESS. We try to answer this question by evaluating the impact of these two proximity notions on the application performance. To this end, we consider two typical applications: a file transfer running over the TCP protocol, and an interactive audio service. For each application, we first develop utility function predicting the performance quality. Then, we evaluate the enhancement of the performance perceived by peers when they choose their neighbors based on the proximity in CHESS instead of the delay-based one. With Planetlab measurements, we observe that the delay proximity is not always a good predictor of quality and that other network parameters have to be considered as well based on the application requirements.

## 4.2 Introduction

The characterization of the proximity in CHESS requires the prediction of the quality perceived by peers. This amounts to develop for each application the correspondent

utility function that models the quality perceived by peers. This can be done by considering the critical performance limitations that decide on the application quality.

In this chapter, we consider two typical applications: a file transfer running over the TCP protocol, and an interactive audio service. For file transfer application, we develop a metric that corresponds to a prediction of the transfer time of a content from server to client. The originality of this metric is in the fact that it considers the characteristics of the paths (i.e., delay, available bandwidth, loss rate) between servers and client together with the server's load and client's maximum congestion window. Our measurement results show that our metric is able to predict with a high accuracy the transfer time of contents transferred on paths having different performance status. For the interactive audio service, we use the E-model for predicting the objective speech quality. The efficiency of the E-model is proven in the literature.

For each application, we determine the proximity in CHESS among a large set of Planetlab nodes using the corresponding utility function. Then, we evaluate the enhancement of the performance perceived by peers when they choose their neighbors based on the proximity in CHESS instead of the delay-based one.

Our main observation is that if one uses the delay to decide on the closest peer to contact for a file transfer, the application performance deteriorates compared to the case where neighbors are identified based on the predicted transfer time function. Furthermore, if one contacts the delay closest peer for an interactive audio service, the speech quality is not as high as that obtained when the peer to contact is the one providing the best predicted speech rating. The same result extends to the other neighbors beyond the closest one.

The chapter is organized as the following. Next, we explore the case of the file transfer application. In Section 4.3.1, we develop our metric for transfer time prediction. Then, we study the impact of proximity definition on the transfer latency in Section 4.3.3. The case of interactive audio service is elaborated in Section 4.4. Finally, the chapter is concluded in Section 5.6.

## 4.3 File transfer over TCP

Firstly, we take the case of file transfer over the TCP protocol. This case can be encountered in the emerging file sharing P2P applications or in the replicated web server context. Applications using TCP are known to form the majority of Internet

traffic [60]. For such applications, the optimal peer to select is the one allowing the transfer of the file within the shortest time. We call *latency* the transfer time.

The optimal ranking of peers from the standpoint of a certain peer is the one providing an increasing vector of transfer latency. This ranking defines the proximity among peers in the CHESS space. Any other ranking results in a different vector and yields a latency increase. Next, we first describe our transfer time prediction function (PTT). Then, we evaluate the enhancement of the TCP latency when the proximity in the CHESS space is used instead of that in the delay space to perform the ranking of peers from the best to the worst.

### 4.3.1 Transfer Time Prediction Function

To predict the content transfer time from a server to a client, we consider that:

- The server uses TCP New Reno for content download.
- The server has an initial congestion window $W_1$ equal to 1 packet and its congestion window during the $i-th$ round trip time $W_i$ is limited by the value $W_{max}$ which is imposed by server or client buffer limitations.
- The client sends an acknowledgment (ACK) for every $b$ data segments received from the server. The value of $b$ is usually equal to 2 due to the Delayed ACK functionality in TCP.
- The channel drops packets independently of each other with a constant probability $P$. Thus, the average number of packets successfully transmitted between the beginning of the transfer and the first packet loss in this case is $1/P$.

The parameters and functions used in our latency prediction function are expressed in the following units:

- $W_i$, $W_{max}$, $d$, and $E[W_{ss}]$ are in packets of size $m$ bytes.
- $E[T_s]$, $E[L_{ss}]$, $E[L_{ca}]$, RTT, and PTT are in seconds.
- $m$, $E[S_{ss}]$, and $S$ are expressed in bytes.
- $A$, $R_{max}$, and $R_{ca}$ are in bytes per second.

To estimate the transfer time of a content of size S, we propose the following function denoted by PTT (Predicted Transfer Time):

$$PTT = E[T_s] + E[L_{ss}] + E[L_{ca}]. \tag{4.1}$$

$E[T_s]$ is the mean request waiting time, i.e., the average time that a request spends in the socket's arrival queue of a server before it is handled by a thread. $E[L_{ss}]$ is the transfer time spent during the slow start phase at the beginning of the download, and $E[L_{ca}]$ is the transfer time spent during the congestion avoidance phase.

We model the socket's arrival queue of a server and its associated threads (of number c) as an M/M/c queue, where $\lambda$ is the mean request arrival rate, and $\mu$ is the mean service rate. Using known results from queuing theory [75], we can write:

$$E[T_s] = \frac{(\frac{\lambda}{\mu})^{\sqrt{2\cdot(c+1)}} - 1}{c \cdot (\mu - \lambda)}. \tag{4.2}$$

Basically, $\lambda$ is calculated in the kernel of the server platform by marking permanently in a certain file the time when a SYN packet arrives to the socket's arrival buffer. $\mu$ is calculated by marking permanently in a certain file the time when a thread begins to serve a request (queued in the socket's arrival buffer) and the time when it finishes serving this request (the TCP connection state is created in the server and the ACK is sent back to the client). Then, $\lambda$ and $\mu$ are updated periodically to be used when necessary.

We use $\gamma$ as the rate of exponential growth of TCP congestion window $W_i$ during the slow start phase:

$$W_{i+1} = W_i + \frac{W_i}{b} = \left(1 + \frac{1}{b}\right) \cdot W_i = \gamma \cdot W_i. \tag{4.3}$$

The end of the slow start phase can be caused by the occurrence of a packet loss along the path in one of three cases (if the content size allows):

- The bandwidth is saturated on the path server - client (client sending rate reaches A). This case can only happen if the product available bandwidth-delay is less than the maximum window value ($A.RTT < W_{max} \cdot m$).
- The congestion window value of the slow start phase reaches the buffering capacity $W_{max}$ (client sending rate reaches $W_{max} \cdot m/RTT$), then after a certain time, a packet is lost on the path server - client due to the random loss process of rate P

we are assuming (in opposite to previous case, the available bandwidth $A$ is not reached here).

- Before reaching the buffering capacity or the available bandwidth on the path server - client, a packet is lost on the path after an average number of packets equal to $1/P$ has been successfully transmitted to the client. In this case, we refer to the client window size reached at the end of the slow start phase by the term $W_P$ which can be expressed as the following:

$$\sum_{i=0}^{\log_\gamma W_P} \gamma^i = \frac{1}{P}, \tag{4.4}$$

then,

$$W_P = \frac{1}{\gamma} \cdot \left( \frac{\gamma - 1}{P} + 1 \right). \tag{4.5}$$

The maximum sending rate that can be reached at the end of the slow start phase can be expressed by taking the minimum over the last three mentioned cases:

$$R_{max} = min \left( A, W_{max} \cdot \frac{m}{RTT}, \frac{1}{\gamma} \cdot (\frac{\gamma - 1}{P} + 1) \cdot \frac{m}{RTT} \right). \tag{4.6}$$

Small size contents can be completely transferred during the slow start phase. When the content size is large, it starts being transmitted in the slow start phase, then continues its transmission in the congestion avoidance phase. In the next two sections we investigate these two cases.

### Transfer completed in the slow start phase

In this case, we consider that $r_S + 1$ round-trips are required to complete the download. The download ends before TCP transmission rate reaches $R_{max}$. The latency components have the following expressions:

$$\begin{aligned} E[L_{ss}] = (r_S + 1) \cdot RTT, \text{ and } E[L_{ca}] = 0 \\ \text{if } \gamma^{r_S} \cdot \frac{m}{RTT} \leq R_{max}, \end{aligned} \tag{4.7}$$

where,

$$\sum_{i=0}^{r_S} \gamma^i = \frac{S}{m}. \tag{4.8}$$

It follows that,

$$r_S = \log_\gamma \left( \frac{S \cdot (\gamma - 1)}{m} + 1 \right) - 1, \tag{4.9}$$

and,

$$\gamma^{r_S} \cdot \frac{m}{RTT} = \frac{S \cdot (\gamma - 1) + m}{\gamma \cdot RTT}. \tag{4.10}$$

Hence, when the transfer is completed in the slow start phase before reaching $R_{max}$, the transfer time is:

$$E[L_{ss}] = \log_\gamma \left( \frac{S \cdot (\gamma - 1)}{m} + 1 \right) \cdot RTT \text{ and } E[L_{ca}] = 0$$
$$\text{if } \frac{S \cdot (\gamma - 1) + m}{\gamma \cdot RTT} \leq R_{max}. \tag{4.11}$$

**Transfer completed in the congestion avoidance phase**

The content size is longer than being achieved in the slow start phase, so it continues its transmission in the congestion avoidance phase (or in the steady state) where the transmission is completed after that the sender rate has reached $R_{max}$:

$$\frac{S \cdot (\gamma - 1) + m}{\gamma \cdot RTT} > R_{max}. \tag{4.12}$$

We evaluate the window expected to be reached at the end of the slow start phase by the following expression:

$$E[W_{ss}] = R_{max} \cdot \frac{RTT}{m} = \gamma^n =$$
$$\begin{cases} A \cdot \frac{RTT}{m} & \text{if } R_{max} = A \\ W_{max} & \text{if } R_{max} = W_{max} \cdot \frac{m}{RTT} \\ \frac{1}{\gamma} \cdot (\frac{\gamma - 1}{P} + 1) & \text{if } R_{max} = \frac{1}{\gamma} \cdot (\frac{\gamma - 1}{P} + 1) \cdot \frac{m}{RTT} \end{cases} \tag{4.13}$$

Thus, we express the number of rounds $n$ which is required to reach the window size $E[W_{ss}]$ (i.e., to reach $R_{max}$) since the beginning of the transfer as:

$$n = \begin{cases} \log_\gamma(A \cdot \frac{RTT}{m}) & \text{if } R_{max} = A \\ \log_\gamma W_{max} & \text{if } R_{max} = W_{max} \cdot \frac{m}{RTT} \\ \log_\gamma(\frac{\gamma-1}{p} + 1) - 1 & \text{if } R_{max} = \frac{1}{\gamma} \cdot (\frac{\gamma-1}{p} + 1) \cdot \frac{m}{RTT} \end{cases} \tag{4.14}$$

We note $r_n+1$ the number of slow start rounds required to transfer $E[S_{ss}]$ data bytes in the case where the transmission is completed after that the sending rate reaches $R_{max}$ (see Equation (4.12)). Hence, $r_n$ can be expressed as:

$$r_n = \begin{cases} n & \text{if } R_{max} = A \\ n + \frac{d}{W_{max}} & \text{if } R_{max} = W_{max} \cdot \frac{m}{RTT} \\ n & \text{if } R_{max} = \frac{1}{\gamma} \cdot (\frac{\gamma-1}{p} + 1) \cdot \frac{m}{RTT} \end{cases} \tag{4.15}$$

where $d$ is the average number of packets which is transmitted successfully in the slow start phase between the time when the transmitting buffer is saturated and the time when the transfer is completed or a packet loss is occurred. We evaluate $d$ in these two cases as the following:

$$d = \begin{cases} \frac{S}{m} - \sum_{i=0}^{\log_\gamma W_{max}} \gamma^i & \text{if } \frac{S}{m} \leq \frac{1}{p} \\ \frac{1}{p} - \sum_{i=0}^{\log_\gamma W_{max}} \gamma^i & \text{if } \frac{S}{m} > \frac{1}{p} \end{cases} \tag{4.16}$$

This implies the following expressions of $d$,

$$d = \begin{cases} \frac{S}{m} - \frac{\gamma \cdot W_{max} - 1}{\gamma - 1} & \text{if } \frac{S}{m} \leq \frac{1}{p} \\ \frac{1}{p} - \frac{\gamma \cdot W_{max} - 1}{\gamma - 1} & \text{if } \frac{S}{m} > \frac{1}{p} \end{cases} \tag{4.17}$$

The time required to transfer $E[S_{ss}]$ data bytes can be expressed as:

$$E[L_{ss}] = RTT \cdot (r_n + 1), \tag{4.18}$$

Thus, Equations (4.14), (4.15), (4.17), and (4.18) give:

$$E[L_{ss}] = RTT \cdot \begin{cases} \log_\gamma(A \cdot \frac{RTT}{m}) + 1 \\ \quad : \text{if } R_{max} = A \\ \\ \log_\gamma W_{max} + \frac{\frac{S}{m} + \frac{1}{\gamma-1}}{W_{max}} - \frac{1}{\gamma-1} \\ \quad : \text{if } R_{max} = W_{max} \cdot \frac{m}{RTT} \text{ and } \frac{S}{m} \leq \frac{1}{P} \\ \\ \log_\gamma W_{max} + \frac{\frac{1}{P} + \frac{1}{\gamma-1}}{W_{max}} - \frac{1}{\gamma-1} \\ \quad : \text{if } R_{max} = W_{max} \cdot \frac{m}{RTT} \text{ and } \frac{S}{m} > \frac{1}{P} \\ \\ \log_\gamma(\frac{\gamma-1}{P} + 1) \\ \quad : \text{if } R_{max} = \frac{1}{\gamma} \cdot (\frac{\gamma-1}{P} + 1) \cdot \frac{m}{RTT} \end{cases} \tag{4.19}$$

The maximum number of bytes that can be sent, when the transmission is completed after the sending rate reaches $R_{max}$ (the condition in Equation (4.12) is satisfied), gets the following expression:

$$E[S_{ss}] = m \cdot \begin{cases} \sum_{i=0}^{n} \gamma^i & \text{if } R_{max} = A \\ \\ \sum_{i=0}^{n} \gamma^i + d & \text{if } R_{max} = W_{max} \cdot \frac{m}{RTT} \\ \\ \sum_{i=0}^{n} \gamma^i & \text{if } R_{max} = \frac{1}{\gamma} \cdot (\frac{\gamma-1}{P} + 1) \cdot \frac{m}{RTT} \end{cases} \tag{4.20}$$

Hence, we can evaluate the transfer time required to complete the transfer in the congestion avoidance phase or in the steady state phase (if the content size allows), as:

$$E[L_{ca}] = \frac{S - E[S_{ss}]}{R_{ca}}, \tag{4.21}$$

where $R_{ca}$ is the TCP average throughput in the congestion avoidance phase (or in steady state). From [80], we use the following expression:

$$R_{ca} = \min\left(A, W_{max} \cdot \frac{m}{RTT}, R_p\right) \tag{4.22}$$

where,

$$R_p = \frac{m}{RTT \cdot \sqrt{\frac{2 \cdot b \cdot P}{3}} + 4 \cdot RTT \cdot \min(1, 3 \cdot \sqrt{\frac{3 \cdot b \cdot P}{8}}) \cdot P \cdot (1 + 32 \cdot P^2)} \tag{4.23}$$

Then,

$$E[L_{ca}] = \frac{1}{R_{ca}} \cdot \begin{cases} S - \frac{1}{\gamma-1} \cdot (\gamma \cdot A \cdot RTT - m) \\ \quad : \text{if } R_{max} = A \\ \\ 0 \\ \quad : \text{if } R_{max} = W_{max} \cdot \frac{m}{RTT} \text{ and } \frac{S}{m} \leq \frac{1}{P} \\ \\ S - \frac{m}{P} \\ \quad : \text{if } R_{max} = W_{max} \cdot \frac{m}{RTT} \text{ and } \frac{S}{m} > \frac{1}{P} \\ \\ S - \frac{m}{P} \\ \quad : \text{if } R_{max} = \frac{1}{\gamma} \cdot (\frac{\gamma-1}{P} + 1) \cdot \frac{m}{RTT} \end{cases} \tag{4.24}$$

Finally, we conclude the following expressions for predicting the content transfer time:

$$PTT = \begin{cases} \text{Equation (4.2)} + \text{Equation (4.11)} \\ \quad : \text{if } \frac{S \cdot (\gamma-1)+m}{\gamma \cdot RTT} \leq R_{max} \\ \\ \text{Equation (4.2)} + \text{Equation (4.19)} + \text{Equation (4.24)} \\ \quad : \text{if } \frac{S \cdot (\gamma-1)+m}{\gamma \cdot RTT} > R_{max} \end{cases} \tag{4.25}$$

Computing PTT is of low complexity. The limitations at the server and client sides (i.e., $\lambda$, $\mu$, $c$, and $W_{max}$) can be determined easily as described before. The major difficulty is the knowledge of the performance parameters on the end-to-end network path where the content transfer take place. This must be achieved by avoiding direct probing between client and servers. Otherwise, measurements cost may outweigh the profit of the characterized proximity. This issue is explored in the next chapter.

### 4.3.2 Evaluating the efficiency of our latency metric

To evaluate the accuracy of our metric, we compare the predicted transfer time (PTT) to the measured transfer time (ReTT) for 105 files, of various sizes ranging between 100KB and 100MB. These files are gathered in Sophia Antipolis from 20 ftp

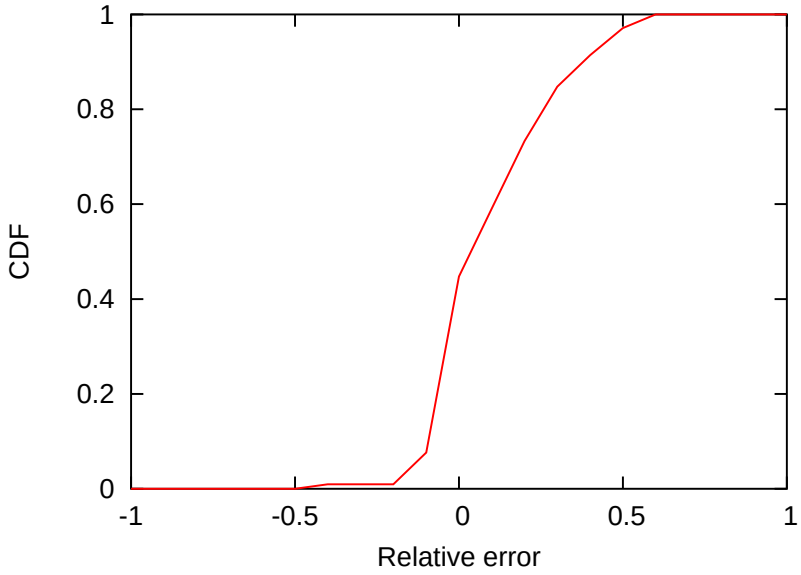



**Figure** 4.1: Transfer time prediction accuracy

servers worldwide distributed [76]. We evaluate the prediction accuracy using the following expression:

$$PredictionAccuracy = \frac{PTT - ReTT}{ReTT}. \tag{4.26}$$

We do not consider, when predicting the transfer time with PTT, the mean request waiting time in the server (Equation (4.2)) due to the inability to read remotely from the WWW servers the parameters required to compute such information ($\lambda$, $\mu$ and c). For non highly loaded servers, the request waiting time can be neglected.

Figure 4.1 draws the CDF of the prediction accuracy for the total 105 transferred files. The figure shows that approximatively (i) 52% of the predictions are accurate within 10%, (ii) 73% of the predictions are accurate within 20%, and (iii) around 98% of the predictions are accurate within 50%. Thus, our metric is able to predict with a good accuracy the transfer time of contents (of different sizes) transferred on paths having different performance status. Next, we study the impact of the different network parameters on the transfer time prediction of large file transfers.

To show the weakness of the prediction based only on the geographical proximity, number of hops or delay, we present the following scenario: a client (in Sophia Antipolis) downloads a file of size 75MB from two servers, $S_{berlin}$ (in Berlin) and $S_{paris}$ (in Paris), during a congested period. We choose to download such large file size during a congested period in order to observe the effect of the bandwidth limitation. As

| | $S_{berlin}$ | $S_{paris}$ |
|---|---|---|
| hops | 13 | 10 |
| RTT | 41 ms | 23 ms |
| A | 10 Mbps | 8 Mbps |
| P | 0 | 0 |
| PTT | 60.30 s | 75.12 s |
| ReTT | 63 s | 77 s |
| S/ReTT | 9.52 Mbps | 7.79 Mbps |

**Table** 4.1: Transfer time prediction when $A$ limits the download rate

shown in Table 4.1, $S_{paris}$ is closer to the client than $S_{berlin}$ in terms of geographical proximity, number of hops and RTT. While the best server selection based on these criterions must be $S_{paris}$, the selection based on our PTT function is $S_{berlin}$, which is the correct choice verified by the real transfer time (ReTT). We observe (in both downloads) that the download rate obtained after dividing the file size by the transfer time, is limited by $A$. Thus, the good performance of our prediction in this scenario is caused by the fact that our PTT function considers the limitation of the available bandwidth (see Equation (4.22)).

Considering the available bandwidth alone is not sufficient; $W_{max}$ should also be taken into account in the prediction function. Table 4.2 proves this claim, where the transfer is achieved in the congestion avoidance phase for the different large document sizes collected from $S_{berlin}$ (in Berlin) to the client (in Sophia Antipolis). During these connections, the measured parameters have the following values: $A$ is equal to $24.34\mathrm{Mbps}$, $P$ is equal to zero, and $W_{max}$ (imposed by the client maximum receiving window) is equal to $65535\mathrm{bytes}$. We observe that the download rate obtained, after dividing each file size by its transfer time, is limited by $W_{max} \cdot m/\mathrm{RTT}$ (equal to $12.2\mathrm{Mbps}$) even though there is more available bandwidth on the path and a negligible packet loss ratio. This is caused by the receiving window limitation which is taken into account in our metric (see Equation (4.22)).

After showing the critical impact of the available bandwidth (in Table 4.1) and the maximum receiving window size (in Table 4.2) limitations on the transfer time prediction, we present in Table 4.3 a trace where the server's maximum sending rate is reduced due to a non-negligible packet loss rate. This trace is the result of 2 files transfer from 2 FTP servers (one located in Hong-Kong and another in Poland) to our

| | PTT (in s) | ReTT (in s) | S/ReTT (in Mbps) |
|---|---|---|---|
| 5.4 MB | 3.85 | 4 | 10.8 |
| 14.63 MB | 9.90 | 11 | 10.64 |
| 25.26 MB | 16.88 | 18 | 11.23 |
| 66 MB | 43.60 | 45 | 11.73 |
| 95.81 MB | 63.16 | 64 | 11.98 |
| 102.24 MB | 67.63 | 68 | 12.03 |

**Table** 4.2: Transfer time prediction when $W_{max}$ limits the download rate

| $S_{hkong}$ | | | | | |
|---|---|---|---|---|---|
| S | 9.75MB | RTT | 381.84ms | P | 0.03 |
| A | 5.77Mbps | $W_{max}$ | 45 | $R_p$ | 115.93kbps |
| PTT | 688s | ReTT | 701s | S/ReTT | 113.87kbps |
| $S_{poland}$ | | | | | |
| S | 10.64MB | RTT | 96 ms | P | 0.04 |
| A | 6.6Mbps | $W_{max}$ | 45 | $R_p$ | 370.16kbps |
| PTT | 235.334s | ReTT | 239s | S/ReTT | 364.79kbps |

**Table** 4.3: Transfer time prediction when P limits the download rate

end host in Sophia Antipolis. During these connections, the value of the packet loss rate is significant. We observe in Table 4.3 that the download rate obtained, after dividing each file size by its transfer time, is limited by $R_p$ (see Equation (4.23)) which is less than the available bandwidth on the path and the limited rate imposed by $W_{max}$. Thus, the sending rate limitation is caused by the packet loss rate, which is considered in our metric (see Equation (4.22)).

### 4.3.3 Impact of Proximity definition on the transfer latency

Within the CHESS space, peers are ranked from the standpoint of a certain peer in a decreasing order of the utility function. Close peers are those providing the best application quality independently of their network locality. In this section, we consider the case of file transfer over the TCP protocol. For such applications, the optimal peer to select is the one allowing the transfer of the file within the shortest latency.

The latency of a TCP transfer depends on the file size. Short transfers are known to

be dominated by the slow start phase which is mainly a function of the round-trip time. Long transfers are dominated by the congestion avoidance phase where the available bandwidth and the loss rate figure in addition to the round-trip time. This difference in the sensitivity to network parameters makes interesting the problem of peer ranking for applications using TCP.

The enhancement of TCP latency between the proximity in CHESS and the delay-based one is computed as follows. Take a peer $p$ and denote the peer having the rank $r$ in the delay space by $p_d(r)$, i.e., the peer having the r-th smallest RTT on its path to $p$. Denote by $p_c(r)$ the peer having a rank $r$ in the CHESS space. This peer has, on its path with $p$, the r-th smallest PTT.

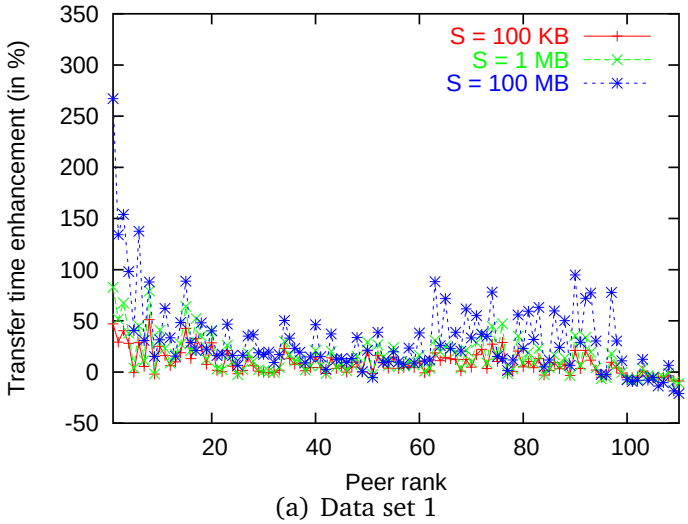
Let $\mathsf{PTT}(x, y)$ denote the transfer latency between peer $x$ and peer $y$. We define the transfer time enhancement (TTE) at rank $r$ as the relative error of $\mathsf{PTT}(p, p_d(r))$ calculated with respect to $\mathsf{PTT}(p, p_c(r))$. More formally $\mathsf{TTE}(r)$ has the following expression:

$$\mathsf{TTE}(r) = \frac{\mathsf{PTT}(p, p_d(r)) - \mathsf{PTT}(p, p_c(r))}{\mathsf{PTT}(p, p_c(r))}. \tag{4.27}$$

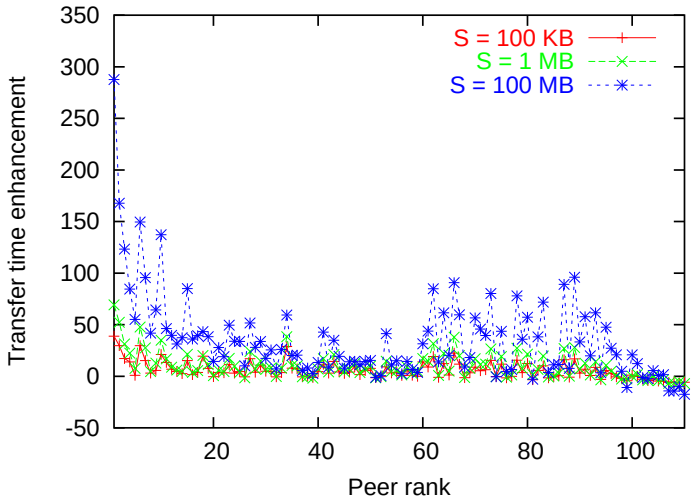
Then, we average the values of TTE at rank $r$ over all peers $p$ (of number 127). With this enhancement function, we are able to evaluate how much on average ranking peers based on the proximity in CHESS provides lower transfer time comparing to the delay case.

We plot in Figure 4.2 the average transfer time enhancement as a function of the rank $r$ for different file sizes (i.e., 100KB, 1MB, 100MB). The enhancement is computed for our three datasets and drawn separately in Figure 4.2. The figures show that the enhancement is much higher for the case of large file transfer. It is between 250% and 300% when the transfer is achieved from the closest peer in the CHESS space. When the rank $r$ increases, the enhancement decreases and it becomes negative at some high ranks. This can be interpreted by the fact that the peers having high ranks in the CHESS space (resp. delay space) are those having low ranks in the delay space (resp. CHESS space) which provide higher (resp. lower) transfer latency. This is completely coherent with our reasoning, at the beginning of the paper, that the paths with farer peers may have better performance (e.g., larger available bandwidth) than those with close peers. Hence, considering the delay alone for proximity characterization is far from being optimal for large file transfer applications.

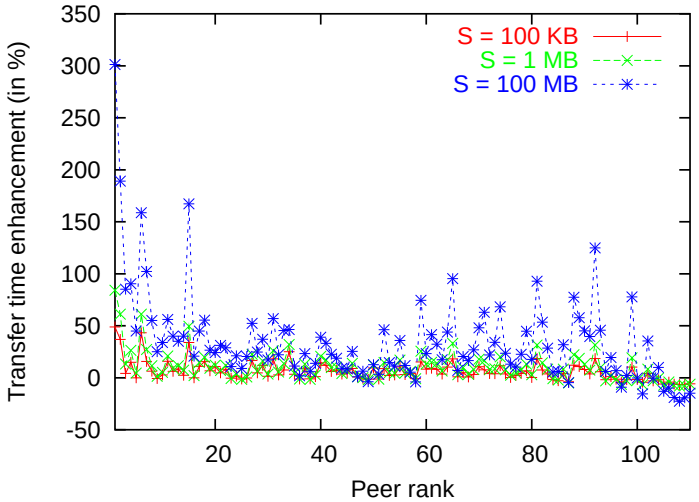
For small file transfer, the enhancement, shown in the figure, is much smaller (i.e., does not exceed 50%) and more uniform than the case of large file transfer. This dis-




(a) Data set 1




(b) Data set 2




(c) Data set 3

**Figure** 4.2: Transfer time enhancement when ranking peers is based on the proximity in CHESS instead of the delay proximity

crepancy between small and large files is due to the different sensitivities the slow start and congestion avoidance phases have to network parameters. Indeed, short transfers are sensitive to the delay, and since the enhancement is computed with respect to the delay ranking, it is small and we are close to the application-level case. Long transfers are more sensitive to the bandwidth (i.e., bandwidth greedy) and since the bandwidth is uncorrelated with the delay (as obtained in the previous chapter), the enhancement is large. Thus, for large file transfer applications, it is much more profitable to characterize the proximity in CHESS, which is based on the utility function computed at the application-level, instead of using the delay proximity.

## 4.4 Interactive audio service

We consider now the case of an interactive audio service, where a set of replicated servers are distributed in the network to provide clients with the same audio communication. To serve a client, a central unit is in charge of identifying the server that can provide the best speech quality. As in the case of file transfer, we do not consider the load at end points and subsequently, we ignore the issue of load balancing. This allows to focus on the impact of network path parameters on proximity characterization.

The speech quality suffers mainly from packet loss and delay. The optimal ranking of peers with respect to a certain peer is the one providing a decreased order of the speech quality. Ranking peers in this way defines the proximity in CHESS. Any other ranking yields a lower speech quality. Our aim is to evaluate how delay-based ranking of peers deviates from the optimal one. This can tell us whether the delay-based proximity is a good predictor of the one determined in CHESS from the standpoint of interactive audio applications.

Speech quality can be characterized subjectively using the Mean Opinion Score (MOS) test. It can be also determined with the E-model, defined by ITU-T G107 [74], which predicts the subjective quality using objective measures (e.g. end-to-end delay, and loss rate). The E-model expresses the audio quality as a rating factor `Rfactor` that accounts for the different transmission parameters having an impact on the conversation. The `Rfactor` calculated by the E-Model ranges from 100 (the best case) to 0 (the worst case). The mapping from the `Rfactor` to the subjective quality and to the MOS score is illustrated in Table 4.4.

Many papers (e.g., [77, 78]) apply the E-model for evaluating the impairments of

| Rfactor interval | Quality of voice rating | MOS |
|---|---|---|
| $90 < Rfactor < 100$ | Best | $4.34 - 4.5$ |
| $80 < Rfactor < 90$ | High | $4.03 - 4.34$ |
| $70 < Rfactor < 80$ | Medium | $3.60 - 4.03$ |
| $60 < Rfactor < 70$ | Low | $3.10 - 3.60$ |
| $50 < Rfactor < 60$ | Poor | $2.58 - 3.10$ |

**Table** 4.4: Rfactor intervals, quality ratings, and the associated MOS

IP telephony applications and provide expressions for the rating factor $Rfactor$. In this chapter, we use the analytical model obtained in [77] and we consider the case of the well known G.711 codec [79]. This model is a reduction of the ITU-T's E-Model.

Let $Rfactor(p_i, p_j)$ denote the speech quality rating between peer $p_i$ and peer $p_j$. According to the E-model proposed in [77], it can be written in the following reduced form:

$$Rfactor(p_i, p_j) = Rfactor_0(p_i, p_j) - I_d(p_i, p_j) - I_e(p_i, p_j), \tag{4.28}$$

where $Rfactor_0(p_i, p_j)$ is the intrinsic quality of the used codec, $I_d(p_i, p_j)$ is the impairment caused by the end-to-end delay, and $I_e(p_i, p_j)$ is the impairment caused by the end-to-end packet loss.

The end-to-end packet loss process is mainly composed of a network component and another one introduced by the de-jitter buffer (i.e., the buffer used at the end points to compensate jitter). The end-to-end delay is composed of the codec delay component, the delay introduced by the de-jitter buffer , and the network delay component. The codec delay component of G.711 codec is equal to $10 \cdot N$ where N is the number of $10ms$ voice frames packed into a single IP packet. We take N equal 2 and subsequently set the codec delay to $20ms$ (i.e. encoding and packetization delay). We assume that the de-jitter buffer introduces a $50ms$ delay and a 2% packet loss. These typical values are considered due to the unavailability of a standard reference model for de-jitter buffer implementations. For the network delay component, we take the half of the measured RTT by assuming that the path is symmetric due to the difficulty to measure accurately the one-way delay.

As for the file transfer application, we experiment the benefits of determining the proximity in the CHESS space for identifying the best server to contact. Now, it is in the scope of an interactive audio service. In this case, $p_c(r)$ is the peer having a rank $r$

in the CHESS space. This peer has, on its path with $p$, the r-th largest $Rfactor$. Recall that the optimal ranking of peers corresponds to a decreased order of the $Rfactor$. The delay-based ranking of peers is obtained by the increased order of the RTT. Hence, we evaluate the enhancement of the speech quality between the proximity in CHESS and the delay-based one with the following expression:

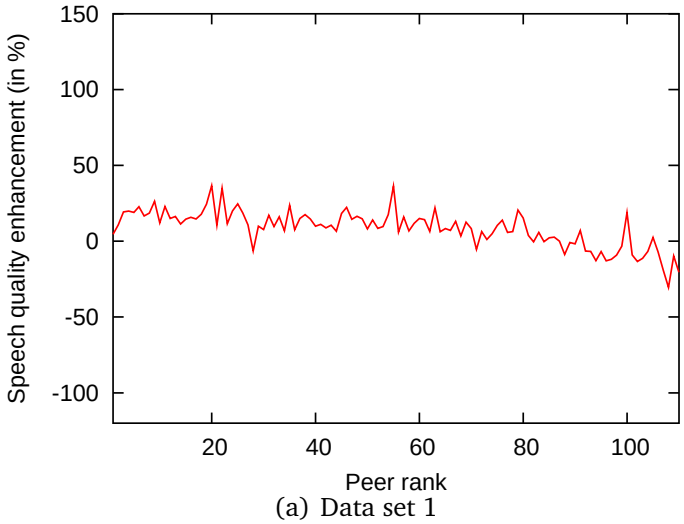
$$SQE(r) = \frac{Rfactor(p, p_c(r)) - Rfactor(p, p_d(r))}{Rfactor(p, p_d(r))}. \tag{4.29}$$

Then, we average the absolute values of $enhancement$ at rank $r$ over all peers $p$ (of number 127). With this enhancement function, we are able to evaluate how much on average ranking peers based on the proximity in CHESS provides better speech quality comparing to the delay case.

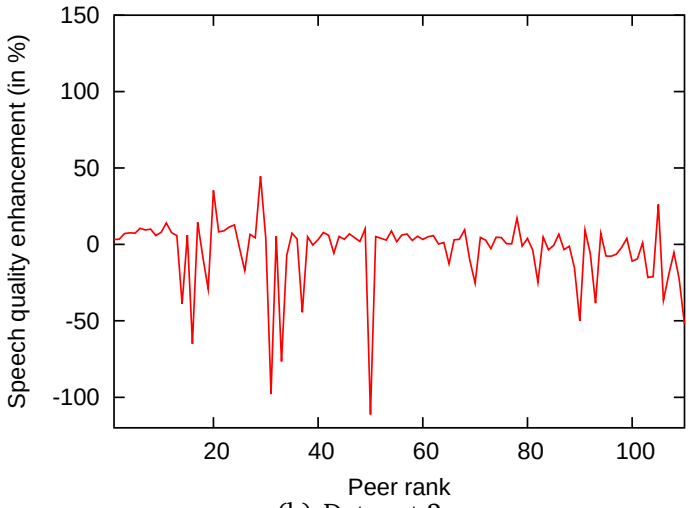
Therefore, we plot in Figure 4.3 the average speech quality enhancement as a function of the rank $r$. As before, the enhancement is computed for our three datasets and drawn separately in Figure 4.3. We observe in these figures that the enhancement is around 5% when the transfer is achieved from the closest peer in the CHESS space. This means that choosing the closest peer in the CHESS space lightly outperforms the case where peers are chosen according to the delay proximity. In other terms, the closest peer in the CHESS space is very probable to be identical to that of the delay space in our settings. This is due to the fact that the majority of peers (i.e., around 90%) have a null loss rate on their path with the delay nearest peer (as obtained in the previous chapter). Indeed, the G711 codec requires a bitrate of $64kbps$ which almost achievable on the paths joining our peers. Thus, choosing the delay closest server for interactive audio service is close to the optimal decision in our settings. Now, when it comes that the delay closest server is busy, the call has to be redirected to another farer and less loaded server. In this case, the figure shows that selecting the server according to the proximity in CHESS provides a considerable speech quality enhancement.

This can be interpreted as the following. The path with the nearest server has a minimum loss rate in our settings. It seems that it is located in a non-congested neighborhood. But, as long as we move away from a peer in the delay space, the loss rate jumps rapidly to values on the order of several percents (see the previous chapter for more details). Thus, when it is necessary to select another server for some reason (e.g., load balancing), taking the delay as a metric of proximity stops being efficient, and the loss rate has to be considered as well.

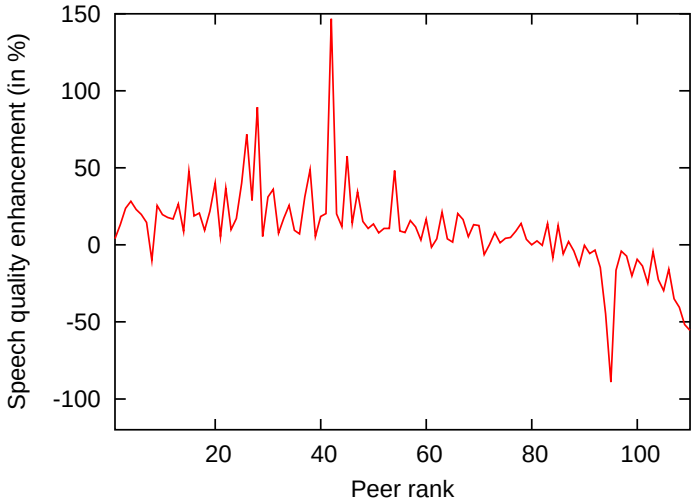
Besides, the figures show that at some high ranks the enhancement is negative. This




(a) Data set 1




(b) Data set 2




(c) Data set 3

**Figure** 4.3: Speech quality degradation when ranking peers is based on the proximity in CHESS instead of the delay proximity

can be interpreted by the fact that the peers having these ranks in the CHESS space (resp. delay space) can be those having low ranks in the delay space (resp. CHESS space) which provide lower (resp. higher) speech rating factors. We conclude that the delay-based proximity is a poor predictor for the interactive audio quality. This stems from the non consideration of the loss rate.

## 4.5 Conclusions

We describe in this chapter how application quality can be improved when proximity is defined using utility functions that model application performance. For file transfer application, this is done using metric that estimates the transfer time by considering the critical parameters on the network paths together with limitations on the client and server sides. Also, we use the E-model for predicting the speech quality for interactive audio applications. For each application, we show with Planetlab experiments how determining the proximity in the CHESS space instead of the delay-based one, permits to enhance the application quality perceived by peers. In the next chapter, we describe how network parameters can be inferred in an easy and scalable way.

# 5

# A Scalable Characterization of the Proximity in CHESS

## 5.1 Summary

The critical task, for determining the proximity in CHESS, is to estimate the network parameters, that impact the utility function, in an easy and scalable way. In this chapter, we show that the delay and loss parameters can be estimated scalably using the matrix factorization approach. Besides, we propose a scalable model that estimates the bandwidth[1] among peers using the bandwidth of the indirect paths that join them via a set of landmarks[2]. Our idea is that an indirect path shares the same tight link with the direct path with a probability that depends on the location of the corresponding landmark with respect to the direct path or any of the two peers subject to bandwidth inference. The results show that a set of 40 to 50 landmarks could be enough for estimating the bandwidth among a worldwide distributed set of peers. Finally, we obtain that the characterization of this proximity, using our bandwidth estimation model, leads to a much better quality than that obtained when using the delay alone.

[1] In this chapter, we almost use the term bandwidth to refer to the end-to-end available bandwidth.

[2] The notion of landmark in our context of bandwidth estimation is different from that used for delay estimation as in [25, 23, 21, 26]. Delay landmarks can be seen as reference points for inferring peer's network position. Bandwidth landmarks can be seen as intermediate nodes connecting peers with indirect paths that are suitable to the direct path bandwidth inference.

## 5.2 Introduction and Motivation

Characterizing the proximity in CHESS requires the estimation of the network parameters including utility function in an easy and scalable way. In other terms, the estimation of the network parameters, between any two peers in a large system, should be achieved with a small measurement overhead and a limited cooperation among peers.

While there were recently several scalable models for estimating the delay [8, 9, 21, 23, 24, 25, 26, 27], there is to the best of our knowledge one only, called *BRoute*[7], for estimating the bandwidth. BRoute assumes that Internet bottlenecks are most probably located on path edges (i.e., the first and the last 4 links of a complete IP level path) and are shared by many different paths. The tool proceeds first by measuring the available bandwidth over the edge links. Then, it estimates the end-to-end available bandwidth of a path as the minimum bandwidth of the link edges it identifies. This is done by determining the peers' AS-level source and sink trees and by identifying the shortest AS path. We believe that these tasks are challenging [81, 82, 14] and add a lot of complexity to the scheme.

In this work, we start by showing that the matrix factorization approach [30, 31], initially proposed for estimating the end-to-end delay, is appropriate for estimating the end-to-end loss rate as well. This is not the case for the bandwidth which is far from being an additive metric. The bandwidth depends on the bottleneck link that may appear anywhere along an end-to-end path. Any model for bandwidth estimation has to identify the route connecting each couple of peers to be able to gauge the bottleneck link. The challenge is that such model must be scalable and easy to deploy.

Therefore, we propose a model for a scalable estimation of the bandwidth among peers that does not require AS-level source and sink trees and does not assume that Internet bottlenecks are on paths' edges. Our model estimates the bandwidth among peers using the bandwidth of the indirect paths that join them via a set of well defined proxies or relays that we call landmark nodes. Basically, the direct and the indirect path share the same tight link with some probability that depends on the location of the correspondent landmark with respect to the direct path or to one of the path end points. This probability is higher if the landmark is closer to one of the path end points. It can be also higher if the delay of the indirect path is nearer to that of the direct one. Thus, the bandwidth of each indirect path contributes to the estimation of the bandwidth of the direct one according to its assigned probability. In this way, one do not care if the bottleneck is on the edge links of the path joining the two peers or if it

is in the middle. Moreover, a peer does not need to determine its AS-level source/sink trees to deduce the edge links that join it to the other peers as done in BRoute. Instead of that, peers have to identify the indirect paths that better represent the direct ones. This task is much easier to realize since we propose to obtain such information using the delay of the paths connecting peers to the landmarks.

Again, using Planetlab measurements, we evaluate the solution and analyze the impact of the location, and number of landmarks on the accuracy of the estimation. We obtain that our estimation model is able to infer accurately the bandwidth among a worldwide set of peers using 40 to 50 landmarks.

Finally, we determine the proximity among peers in both the delay and CHESS spaces in order to evaluate their impact on the application performance. A typical file transfer application is considered to evaluate the quality of service perceived by peers when they choose their neighbors based on these two distinguished proximity notions. We determine the proximity in CHESS among a worldwide distributed set of Planetlab nodes using our transfer time prediction function. We observe that the characterization of this proximity, using our bandwidth estimation model, leads to a much better quality than that obtained when using the delay alone.

The outline of the chapter is as follows. Next, we describe the matrix factorization approach and then we use it for estimating scalably the delay and loss rate parameters. We introduce, in Section 5.4, a scalable model for estimating the bandwidth. In Section 5.5, we evaluate the performance gain achieved when considering the bandwidth estimations for determining the proximity in the CHESS space. The paper is concluded in Section 5.6.

## 5.3 Distance Matrix Factorization

In [30, 31], the authors propose to infer the network delay in a way that is able to model the sub-optimal and asymmetric routing that may exist in practice. This model is based on the distance matrix factorization. Next, we describe the model and then we study its capacity to estimate the delay, the loss rate and the available bandwidth.

Basically, nearby peers are expected to have similar delay distances to all the other peers. Then, an $n \times n$ delay matrix may contain dependent rows[3]. From linear algebra, such matrix can be expressed as the product of two smaller matrices:

[3]Dependent rows means equal rows or can be expressed as a linear combination of other rows.

$$D \approx XY^T, \tag{5.1}$$

where X and Y are $n \times d$ matrices with $d \ll n$. In contrast with the coordinate-based approaches that assign to each peer a coordinate vector, a delay matrix factorization associates to each peer $p_i$ two vectors $\overrightarrow{X_i}$ and $\overrightarrow{Y_i}$ of d dimensions. $\overrightarrow{X_i}$ is called the outgoing vector and $\overrightarrow{Y_i}$ the incoming vector for peer $p_i$. Hence, the estimated delay from peer $p_i$ to peer $p_j$ is simply the scalar product of the outgoing vector of $p_i$ and the incoming vector of $p_j$.

The factorization of D ($n \times n$) into two matrices of smaller dimension can be done by its singular value decomposition (SVD). The singular value decomposition of D can be expressed as:

$$D = USV^T, \tag{5.2}$$

where U and V are $n \times n$ orthogonal matrices, and S is an $n \times n$ diagonal matrix. Calculating SVD consists of finding the eigen values and eigen vectors of the matrices $DD^T$ and $D^TD$. The matrix S contains the singular values of D ranked in a decreasing order[4]. The columns of the matrices U and V are respectively the eigen vectors of $DD^T$ and $D^TD$.

When the number of dependent rows in D increases, the number of principle components decreases and subsequently the number of singular values that are significant in magnitude decreases as well. This is due to the property of the singular values that measure the significance of the contribution from each principle component. After eliminating the null and negligible values, the remaining singular values get a number d which is less or equal to n.

Thus, an $n \times n$ delay matrix D can be decomposed into two smaller matrices X ($n \times d$) and Y ($n \times d$) which are computed as the following:

$$X_{ij} = U_{ij}\sqrt{Sjj}, \tag{5.3}$$

$$Y_{ij} = V_{ij}\sqrt{Sjj}, \tag{5.4}$$

where $i = 1...n$ and $j = 1...d$. This leads to the matrix $\hat{D} = XY^T$ which is a low-rank approximation to the real delay matrix D. This approximation depends obviously on

[4]The singular values are calculated as square roots of the nonzero eigen values of $DD^T$.

the choice of $d$ which can be determined by minimizing the following squared error function:

$$\sum_{i}^{n}\sum_{j}^{n}(D_{ij} - \overrightarrow{X_i} \cdot \overrightarrow{Y_j})^2, \tag{5.5}$$

where $\overrightarrow{X}_i \in \mathbb{R}^d$ and $\overrightarrow{Y}_j \in \mathbb{R}^d$.

**Distance reconstruction**

We evaluate the error occurred from such approximation for different values of the reduced dimension $d$. This is done for the delay, loss rate, and bandwidth matrices. Note that in the literature only the delay was studied. We want to check whether the matrices for the other metrics can be compressed as well. This is important for the design of the scalable CHESS space. We construct the three matrices using our three measurement sets carried out over the 127 Planetlab nodes. The diagonal of each constructed matrix contains null values. These null values make difficult the matrix dimensionality reduction which depends on the linear dependency among rows. Therefore, we interchange the diagonal values by values that simulate a peer clustering process. For the delay and loss rate matrices, we fill the diagonal values (resp. $D_{ii}$, and $P_{ii}$) by the minimum values of the correspondent rows (resp. $min_{j=\{1..127\}}D_{ij}$, and $min_{j=\{1..127\}}P_{ij}$). For the available bandwidth matrices, we fill the diagonal values ($A_{ii}$) by the maximum value of the correspondent rows ($max_{j=\{1..127\}}A_{ij}$).

We plot in Figure 5.1, the mean reconstruction error of these matrices as a function of the reduced dimension $d$. For different values of the reduced dimension $d$ on the x-axis, the y-axis draws the mean relative error (MRE) of the approximated matrix values (estimated values $\hat{D}_{ij}, i, j = \{1...n\}$) with respect to the real values (i.e., measured values $D_{ij}, i, j = \{1...n\}$). More formally, MRE is computed as the following:

$$MRE = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{|\hat{D}_{ij} - D_{ij}|}{D_{ij}} \tag{5.6}$$

Figure 5.1 shows how the reconstruction error (i.e., mean relative error MRE on the y-axis) decreases when the measured matrix is less and less compressed (i.e., the value of $d$ on the x-axis increases). For the delay and loss rate matrices, the figure shows that the reconstruction error is small even for a large dimensionality reduction (as for $d = 10$, MRE between 0.4 and 0.6). This is not the case for the available

bandwidth matrix where the reconstruction error is more than twice that obtained for reconstructing the delay and loss matrices. This can be explained by the fact that the delay and loss parameters are additive and subsequently it is expected to find a linear dependency among their vectors. This is not the case for the bandwidth parameter.

**Distance prediction**

We apply the matrix factorization technique for estimating the network parameters among peers that have not been used for constructing the network matrices. This is already done for the delay, in [30, 31], where the authors estimate the delay on the path between two peers $p_i$ and $p_j$ as the dot product between the delay outgoing vector of $p_i$ and the delay incoming vector of $p_j$. The two vectors are calculated separately by probing a set of landmarks. In that paper, the authors show that their matrix factorization model provides more accurate delay estimations comparing to the other network embedding approaches. We wonder whether this model is able to estimate accurately the loss rate and the bandwidth as well. Therefore, we take 40 nodes, out of the 127 Planetlab nodes, as landmarks[5] and the remaining 87 nodes as peers.

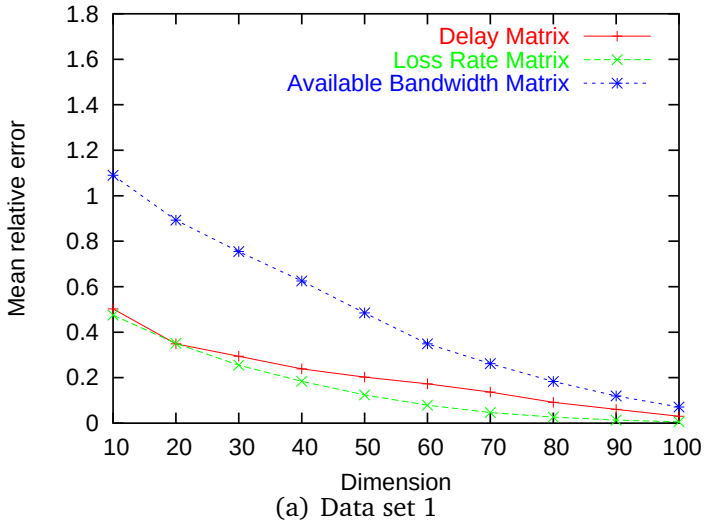
We begin by constructing the delay, loss, and bandwidth matrices using the measurements achieved among the 40 landmarks over our three datasets. The landmarks are selected in the following three ways:

- **Random**
  We randomly choose landmarks without considering their pairwise distances.

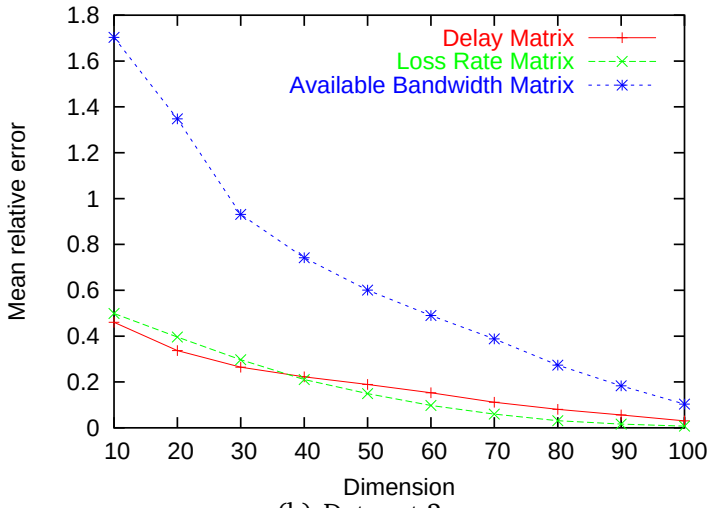
- **Maximum-distance**
  The maximum-distance algorithm consists in determining the landmark set whose summed pairwise distance is greedily maximized. This is realized as follows: (1) at the beginning, we randomly choose one peer as the first landmark, (2) from the rest of peers, we determine the next landmark as the one having the maximum average delay to the existing landmarks, (3) we repeat step (2) until reaching the required number of landmarks.

- **N-means**
  N-means is a well-known algorithm for grouping peers into N clusters and then

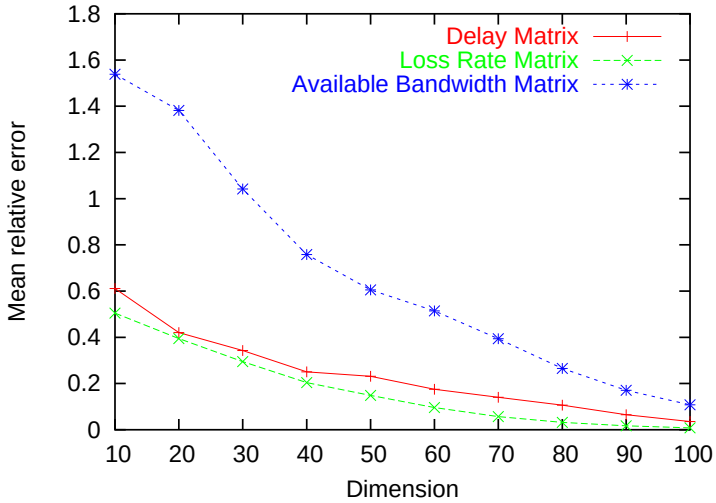
[5] We note that previous studies show that 20 landmarks can be enough for estimating the delay among a wide distributed set of peers.




(a) Data set 1




(b) Data set 2




(c) Data set 3

**Figure** 5.1: Matrix reconstruction error after the reduction to the different dimensions over the 3 Planetlab datasets

choosing the closest peers to the clusters' centroids as landmarks. We identify the N-means landmark set with the following procedure: (1) initially, we randomly choose N peers as landmarks, (2) we assign each of the rest of peers to the cluster of the closest landmark, (3) we re-determine the N landmarks by identifying in each cluster the peer that minimizes the summed delay with the other peers in the same cluster, (4) we repeat step (2) and (3) until reaching an unchangeable set of landmarks.

Then, we determine the delay, loss, and bandwidth outgoing and incoming vectors for each peer from the measurement it conducted to the landmarks. First, landmarks compute their vectors by measuring the paths among each other (we take $d = n = 40$). Next, peers infer their vectors in a distributed way. We explain how this to be done for the delay. The loss and bandwidth vectors are calculated similarly.

For a given landmark set, each peer $p$ has to measure the delay to and from each landmark. The delay from $p$ to landmark $L_i$ is denoted by $D_i^{out}$ and that from the landmark $L_i$ to $p$ by $D_i^{in}$. The outgoing ($\overrightarrow{X}_{new}$) and incoming ($\overrightarrow{Y}_{new}$) vectors of $p$ should satisfy the following equations:

$$D_i^{out} = \overrightarrow{X}_{new} \cdot \overrightarrow{Y}_i \tag{5.7}$$

$$D_i^{in} = \overrightarrow{X}_i \cdot \overrightarrow{Y}_{new} \tag{5.8}$$

The solution of these equations can be obtained by minimizing the square error summed over all landmarks:

$$\overrightarrow{X}_{new} = arg_{\overrightarrow{U} \in \mathbb{R}^d} Min \sum_{i=1}^{n} (D_i^{out} - \overrightarrow{U} \cdot \overrightarrow{Y}_i)^2 \tag{5.9}$$

$$\overrightarrow{Y}_{new} = arg_{\overrightarrow{U} \in \mathbb{R}^d} Min \sum_{i=1}^{n} (D_i^{in} - \overrightarrow{X}_i \cdot \overrightarrow{U})^2 \tag{5.10}$$

The solution can be also expressed in the following form of matrix operations:

$$\overrightarrow{X}_{new} = (D^{out} Y)(Y^T Y)^{-1} \tag{5.11}$$

$$\overrightarrow{Y}_{new} = (D^{in} X)(X^T X)^{-1} \tag{5.12}$$

After determining the delay (resp. loss and bandwidth) outgoing and incoming vectors, we estimate the delay (resp. loss and bandwidth) between each pair of the 87 peers by the scalar dot product of their vectors. We evaluate the mean and the standard deviation of the absolute relative error (denoted respectively by MRE and stdRE) of the estimated delay (resp. estimated loss and bandwidth) compared to the measured values. Tables 5.3, 5.1, and 5.2 show MRE and stdRE of the delay, loss and bandwidth estimation error obtained when using the different landmark sets and over our three datasets. The table shows that the estimation accuracy is quasi-similar when landmark sets are chosen using N-means, and max-distance algorithms. In these cases, MRE and stdRE for the delay estimation varies between 0.35 and 0.5. With the random landmark set, the estimations are less accurate. We note that these results are coherent with those obtained in [31]. Besides, we observe in the tables a good estimation accuracy for the loss rate parameter when using the N-means and max-distance landmark sets; MRE and stdRE varies respectively in the intervals $[0.25, 0.5]$ and $[0.3, 0.7]$. For the bandwidth, the table shows that the estimations are not as accurate as the delay and loss ones. A relatively large estimation errors are obtained for the different landmark and measurement sets; MRE and stdRE vary respectively in the intervals $[1.5, 2.5]$ and $[1.5, 5]$.

Thus, while the matrix factorization approach provides accurate estimations for the delay and loss rate parameters, it is not the case for the bandwidth. This is because the bandwidth is not an additive metric and it rather depends on the bottleneck link that may appear anywhere along an end-to-end path. This means that network embedding approaches are not expected to be the appropriate tools for estimating this parameter. One may conclude that a bandwidth estimation model must identify the route connecting each couple of peers to detect its bottleneck link. The challenge is that such model must be scalable and easy to deploy.

## 5.4 Scalable end-to-end bandwidth inference

### 5.4.1 Model Overview

Our model consists in inferring the bandwidth between any pair of peers based only on their bandwidth vectors. A peer obtains its bandwidth vector by measuring the direct and reverse bandwidth on its path to the landmarks. Hence, the scheme is scalable since its overhead is linear with the number of peers in the system. Also, it is

| MRE | Random | Max-distance | N-means |
|---|---|---|---|
| Delay | 1.2 | 0.52 | 0.5 |
| loss | 0.9 | 0.355 | 0.24 |
| ABw | 1.4 | 1.85 | 2.8 |
| stdRE | Random | Max-distance | N-means |
| Delay | 0.66 | 0.4 | 0.43 |
| loss | 0.8 | 0.728 | 0.43 |
| ABw | 2.7 | 1.6 | 2.7 |

**Table** 5.1: Estimation error by SVD over dataset 1

| MRE | Random | Max-distance | N-means |
|---|---|---|---|
| Delay | 1.19 | 0.49 | 0.47 |
| loss | 0.65 | 0.28 | 0.25 |
| ABw | 2.5 | 2.3 | 2 |
| stdRE | Random | Max-distance | N-means |
| Delay | 0.71 | 0.4 | 0.41 |
| loss | 0.43 | 0.32 | 0.34 |
| ABw | 4.8 | 5 | 4.7 |

**Table** 5.2: Estimation error by SVD over dataset 2

| MRE | Random | Max-distance | N-means |
|---|---|---|---|
| Delay | 1.3 | 0.49 | 0.5 |
| loss | 0.54 | 0.275 | 0.244 |
| ABw | 1.9 | 1.6 | 1.8 |
| stdRE | Random | Max-distance | N-means |
| Delay | 0.70 | 0.37 | 0.42 |
| loss | 0.4 | 0.43 | 0.42 |
| ABw | 3.6 | 4.2 | 3.9 |

**Table** 5.3: Estimation error by SVD over dataset 3

easy to implement since (i) peers do not need to know and probe each other; any node can estimate the bandwidth between any two peers based on their bandwidth vectors, and (ii) no need for the routing information used in [7] (described in Section 5.2).

For each couple of peers, we denote by:

- *Direct path* the network path that joins them directly using IP routing, and by
- *Indirect path* the path that joins the two peers by passing by a landmark node. We note that N indirect paths (i.e., N being the number of landmarks) are assigned to each direct path.

We estimate the end-to-end bandwidth of a path joining two peers using the following class of linear functions:

$$EB = \sum_{i=1}^{N} w_i \cdot BI_i, \tag{5.13}$$

where $BI_i$ is the bandwidth of the indirect path that passes by the landmark $L_i$, and $w_i$ is the normalized weight (i.e., $\sum_{i=1}^{N} w_i = 1$) assigned to this indirect path according to the location of its corresponding landmark with respect to the two peers. The idea behind this definition of the estimation function is as follows. We consider that the direct path shares the same tight link with the indirect path that passes by the landmark $L_i$ with some probability $w_i$. This probability depends on the location of the corresponding landmark with respect to the direct path or to one of its end points. $w_i$ is higher if for example $L_i$ is closer to one of the path end points (to be validated in the next section). By varying the expression of the probability $w_i$, we are able to cover different policies for bandwidth estimation ranging from the one that gives the same priority to all landmarks to the one that privileges the landmark that we deem the most suitable for the bandwidth inference.

For example, take the case of Figure 5.2, where peers $\{p_1, p_2, p_3, p_4\}$ are connected to the *network core* via a set of path edges. We suppose that there are three landmarks distributed in the network core as intermediates nodes for bandwidth inference. Thus, for the direct path joining $p_1$ to $p_3$ ($p_1p_3$), there are three associated indirect paths $\{1, 2, 3\}$. Each indirect path $i$ (i.e., $i = \{1,2,3\}$) is composed of the two IP path portions $p_1L_i$ and $L_ip_3$.

Suppose that the direct path $p_1p_3$ passes by the routers $\{R_{1a}, R_{1b}, R_{1e}, R_{1i}, R_{3i}, R_{3d}, R_{3b}, R_{3a}\}$ and by some set of routers in the network core. In this case, the direct path $p_1p_3$ overlaps with the

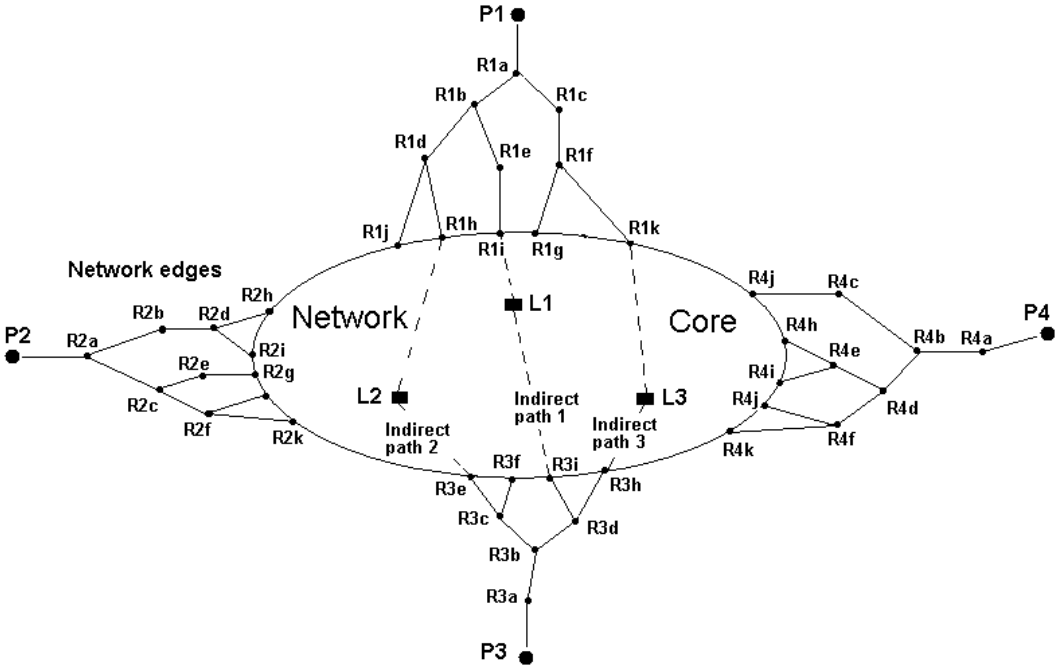



**Figure** 5.2: Direct versus Indirect paths

- Indirect path 1 by at least the two path portions $\{p_1\ R_{1a}\ R_{1b}\ R_{1e}\ R_{1i}\}$ and $\{R_{3i}\ R_{3d}\ R_{3b}\ R_{3a}\ p_3\}$,
- Indirect path 2 by at least the two path portions $\{p_1\ R_{1a}\ R_{1b}\}$ and $\{R_{3b}\ R_{3a}\ p_3\}$,
- Indirect path 3 by at least the two path portions $\{p_1, R_{1a}\}$ and $\{R_{3a}, p_3\}$.

Hence, the bandwidth of the indirect path 1 must be assigned the largest weight when estimating the bandwidth of the direct path $p_1p_3$ since these two paths have the most common portions and subsequently it is more probable that they share the same bottleneck link.

In our description, we focus on path edges since it is very probable that the bottleneck is located on these edges as shown in [7]; the authors of [7] obtain that it is almost 4 hops away from the path end points. But contrary to [7], our solution applies to other contexts where the bottleneck is not solely on the edge. The key point is that estimating the bandwidth on the path between two peers depends mainly on the location of the landmarks with respect to these peers. This dependency is explored in the next section.

### 5.4.2 Impact of landmarks' locations

To evaluate the impact of landmarks' locations on the bandwidth estimation accuracy, we consider the following real scenario conducted over Planetlab. We take 8 Planetlab nodes distributed in different European countries as landmarks. We also take 14 Planetlab nodes completely distributed in Europe as peers. We ask the question of whether the European landmarks can help to estimate accurately the bandwidth on the path between European peer and any other peer. Therefore, each of the European peers measures the RTT and the direct and reverse available bandwidth to 34 Planetlab nodes distributed worldwide. This leads to 476 measured paths. Then, we infer the bandwidth of these paths using Equation (5.13) and we compare the estimations with the measured values. This comparison is done for different weights in the estimation function (Equation 5.13).

Our landmark nodes are chosen with the main concern to have a high bandwidth connectivity to the Internet. This is an important requirement since we want to avoid having the bottleneck, of an indirect path, decided by the landmark itself.

We consider different forms of the probability $w_i$, and subsequently of the end-to-end bandwidth estimation function. By doing that, we are able to study the correlation between the estimation accuracy and the locations of the landmarks. We divide the study into two main parts:

- The estimation function depends on the delay closeness between the direct path and the indirect paths.
- The estimation function depends on the delay closeness between the landmarks and the path end points.

#### Estimating bandwidth based on indirect paths' delays

One possibility is to estimate the end-to-end bandwidth of a direct path using that of the indirect path having the shortest delay. Even though we found satisfactory results, we believe this method is not sufficient for providing accurate estimation since direct IP routing may lead to an end-to-end delay larger than the one of the shortest indirect path, with both paths having different sets of links and hence different bottlenecks. The accuracy could improve by considering more than one indirect path in the estimation function while assigning more weight to those having shorter delays. This considera-

tion is mainly recommended when there are more than one indirect path having delays on the order of that of the shortest one.

Thus, we consider all the N indirect paths in the bandwidth estimation function (Equation 5.13) with the following expression for the weight $w_i$:

$$w_i = \frac{C_i}{\sum_{i=1}^{N} C_i}, \quad \text{for } i = \{1, .., N\} \tag{5.14}$$

where,

$$C_i = \left(\frac{RI_{min}}{RI_i}\right)^{\alpha}, \tag{5.15}$$

$RI_i$ is the round trip delay of the indirect path that passes by the landmark $L_i$, $RI_{min}$ is that of the shortest indirect path among the N indirect paths, and $\alpha$ is a positive real number.

Hence, we obtain the following first expression of the estimation function:

$$EB_1 = \sum_{i=1}^{N} \frac{\left(\frac{RI_{min}}{RI_i}\right)^{\alpha}}{\sum_{i=1}^{N}\left(\frac{RI_{min}}{RI_i}\right)^{\alpha}} \cdot BI_i \tag{5.16}$$

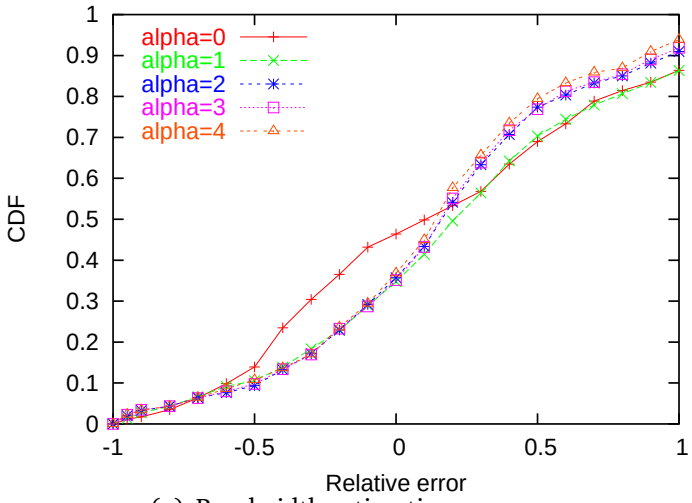
We draw in Figure 5.3(a) the CDF of the relative bandwidth estimation error which is calculated as:

$$\frac{EB_1 - A_{measured}}{A_{measured}}, \tag{5.17}$$

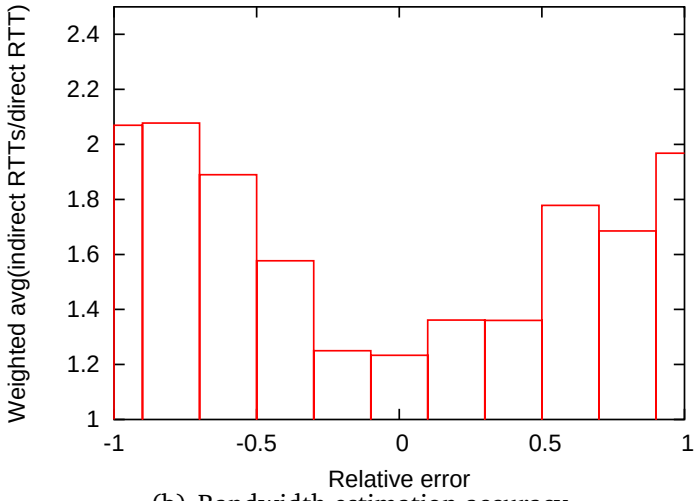
for all the bandwidth estimations and for different values of $\alpha$. The figure shows that when the $\alpha$ parameter increases, the estimation accuracy improves. This is expected since when $\alpha = 0$, the bandwidth component of all the indirect paths gets the same probability, and when $\alpha$ becomes large, the indirect paths having shorter delays, and hence better representation of the direct path, get more probability than those having larger delays. For $\alpha > 3$, we observe that the results become steady. This can be explained by the fact that the estimation becomes only dependent on the indirect paths having a delay on the order of that of the shortest indirect path. For $\alpha = 4$, the figure shows that approximatively 40% of the estimations are accurate within 25% and 70% of the estimations are accurate within 50%.

To show the correlation between the estimation accuracy and the difference in the delay between the direct and the indirect paths, we plot Figure 5.3(b) for the case $\alpha = 4$. For an estimation error interval (on the x axis) of length 0.2, the y axis shows

the sum $\sum_{i=1}^{N}(w_i \cdot RI_i)/R_d$, which is a weighted average of the ratio of the indirect paths' delays and the direct path delay ($R_d$). This sum is averaged over each interval of length 0.2. The figure shows a clear correlation between the two entities plotted on the x and y axis. This means that when some landmarks are located such that the delay of their corresponding indirect paths is close to that of the direct path, the estimation accuracy is high.




(a) Bandwidth estimation accuracy




(b) Bandwidth estimation accuracy

**Figure** 5.3: Bandwidth estimations based on indirect paths' delays

**Estimating bandwidth based on the delay distance between landmarks and peers**

Now, instead of relying our estimation on the end-to-end delay of the indirect paths, we focus on how close landmarks are to the direct path end points. Thus, for each pair of peers, we consider the N indirect paths in the bandwidth estimation model after assigning more weight for those going through landmarks that are closer to any of the two peers. Basically, we want to check if these latter indirect paths are more representative of the direct path than the ones having smaller end-to-end delays. We express the coefficients $C_i$ as:

$$C_i = \left(\frac{R_{min}}{R_i}\right)^{\alpha}, \tag{5.18}$$

where,

$$R_i = min(R_{xi}, R_{yi}), \tag{5.19}$$

$R_{xi}$ represents the round trip delay between the peer x and the landmark $L_i$, and

$$R_{min} = min_{i=1..N} R_i. \tag{5.20}$$

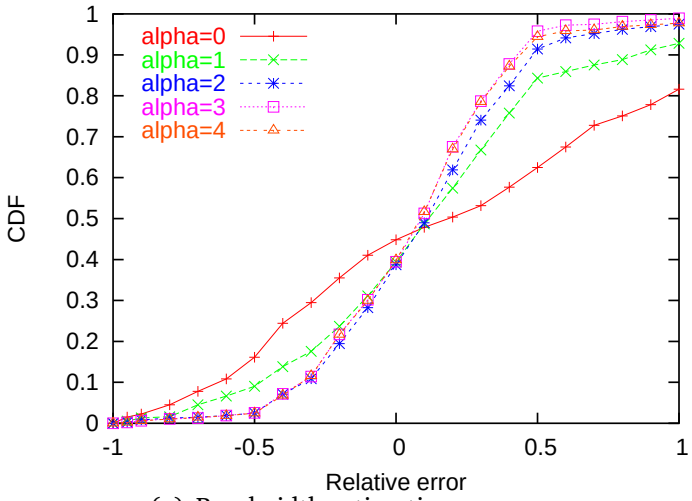
We recalculate the $w_i$ function (Equation 5.14) and subsequently the estimation function (Equation 5.13) with these new coefficients $C_i$. Thus, we obtain the following second expression of the estimation function:

$$EB_2 = \sum_{i=1}^{N} \frac{\left(\frac{R_{min}}{R_i}\right)^{\alpha}}{\sum_{i=1}^{N}\left(\frac{R_{min}}{R_i}\right)^{\alpha}} \cdot BI_i \tag{5.21}$$

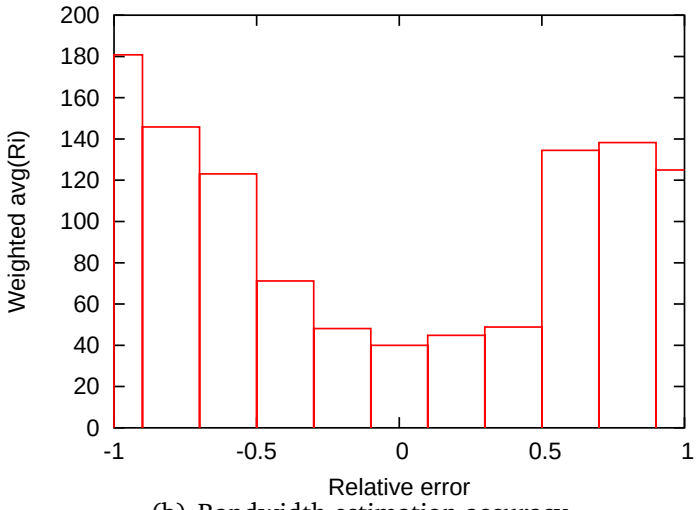
Then, we plot in Figure 5.4(a) the CDF of the relative estimation error which is calculated as:

$$\frac{EB_2 - A_{measured}}{A_{measured}}, \tag{5.22}$$

for all the bandwidth estimations and for different values of $\alpha$. As before, when $\alpha$ increases, the indirect paths having landmarks close to one of the two peers get more weight. The results shown in the figure become stationary for $\alpha > 3$. This is because the bandwidth estimations become only dependent on the few indirect paths having landmarks close to one of the peers. The figure shows better results comparing to the previous cases studied; around 57% of the estimations are accurate within 25% and 93% of the estimations are accurate within 50%.




(a) Bandwidth estimation accuracy




(b) Bandwidth estimation accuracy

**Figure** 5.4: Bandwidth estimations based on the delay distance between landmarks and peers

To show the correlation between the estimation accuracy and the landmarks' closeness to the extremities, we plot Figure 5.4(b) for the case $\alpha = 4$. For an estimation error interval (on the x axis) of length 0.2, the y axis shows $\sum_{i=1}^{N} w_i \cdot R_i$ averaged over the estimations inside the interval. The figure shows a clear correlation between the two entities in the x and y axis. This means that when some landmarks (among the N) are close to the path extremities, the estimation accuracy improves. Furthermore, it becomes better than the case where the estimation depends on the shortest indirect paths (see Figures 5.3(a) and 5.4(a)).

This is due to the fact that the indirect paths, going through landmarks which are close to path extremities, are more expected to provide better representation of the direct path since it is more probable that IP will route through them. In such cases, we recommend the use of the estimation function $EB_2$ presented in this section. But for the couples of peers that are relatively far from all the landmarks, the delay closeness between landmarks and peers is not expected to be anymore a good criterion for testing the representation of the indirect paths with respect to the direct path. In such situations, the location of the landmarks with respect to the direct path can be more helpful for this purpose. Therefore, we propose the use of $EB_1$ (described in Section 5.4.2) which depends on the shortest indirect paths and it is expected to estimate accurately the bandwidth when the delays of some indirect paths are close to that of the direct one (as obtained in Figure 5.3(b)); even if landmarks are far from path end points.

### 5.4.3 Bandwidth estimation function

Hence, for each couple of peers, we apply the following statement for determining the expression of the bandwidth estimation function (EB):

$$EB = \begin{cases} EB_2 & \text{if } R_{min} < R_{threshold}, \\ \\ EB_1 & \text{elsewhere,} \end{cases} \tag{5.23}$$

where $R_{min}$ is expressed in Equation 5.20, $R_{threshold}$ is a threshold delay for examining the closeness between landmark and peers, $EB_1$ and $EB_2$ are expressed respectively in Equations (5.16) and (5.21).

Thus, to estimate the bandwidth on the path between two peers, we first check the delay closeness between landmarks and peers by the statement $R_{min} < R_{threshold}$. We take $R_{threshold} = 80ms$ based on what we have obtained in Figure 5.4(b). In that figure,

the estimations, that are within the 50% accuracy, belong to the cases where at least one landmark is close to one of the path endpoints by a distance which is less than 80ms (i.e., $R_{min} < 80ms$). In such cases, the estimation function $EB_2$, which depends on the delay distance between landmarks and peers, is preferable to be considered for inferring the bandwidth. Otherwise, landmarks are relatively far from the path extremities and subsequently it becomes more helpful to rely on the location of the landmarks with respect to the direct path. In such cases, we use the estimation function $EB_1$ where the delay closeness between the direct and indirect paths is the criterion.

The challenge appears when the overlay network contains a large number of peers widely distributed. In this case, to infer accurately the bandwidth on the path between any pair of peers, a larger number of landmarks is obviously required for our estimation model. This issue is explored in the next section.

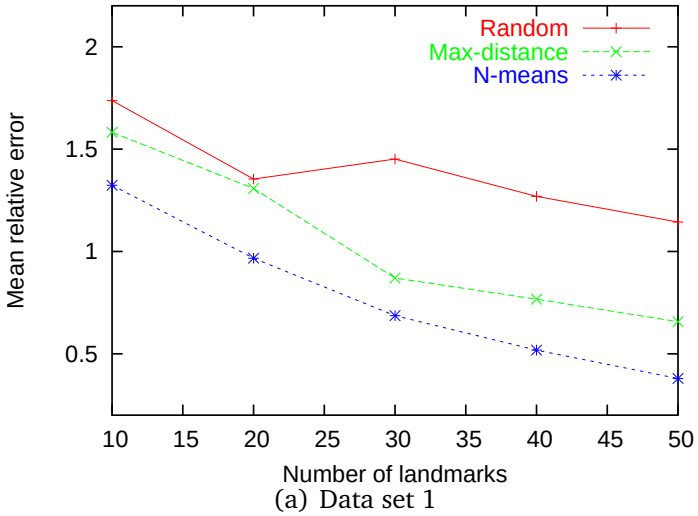
### 5.4.4 Impact of the number of landmarks

Instead of 8 landmarks distributed in Europe, we now infer the available bandwidth, on the paths joining a worldwide set of peers, using different landmark sets having the numbers $N = \{10, 20, 30, 40, 50\}$ distributed worldwide. Each landmark set is selected from 100 nodes having the highest bandwidth connectivity among the 127 Planetlab nodes[6]. Landmark selection is realized randomly, using the max-distance algorithm, and using the N-means algorithm. See Section 5.3 for more details concerning these algorithms.

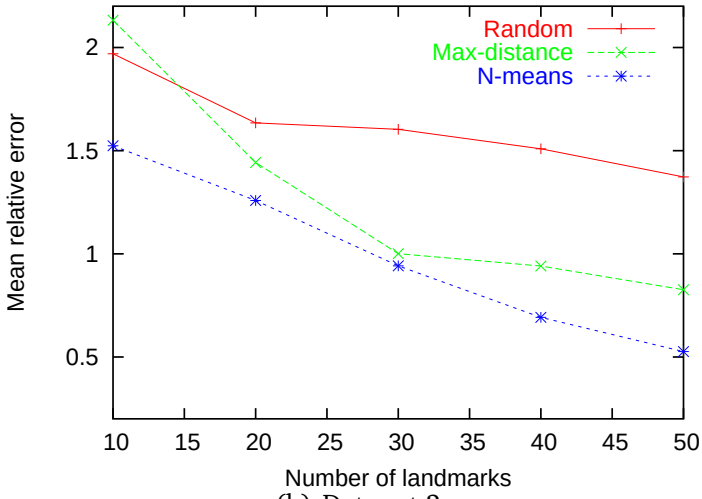
We plot in Figure 5.5 and 5.6 the mean (MRE) and the standard deviation (stdRE) of the bandwidth estimation error for the different landmark sets and over our three datasets. MRE is calculated as follows. For each set of N landmarks chosen from the 127 peers as described before, we infer the bandwidth on the paths joining the rest of peers of number $127 - N$. This is done using the bandwidth estimation function described in Section 5.4.3. Then, MRE (resp. stdRE) is computed as the mean (resp. the standard deviation) of the absolute relative errors of the estimation values with respect to the measured values (as computed in 5.6).

Figures 5.5 and 5.6 show that the bandwidth estimation error decreases when the number of landmarks increases. This is expected since with a wider coverage of the landmarks it becomes more probable to find indirect paths which better represent the

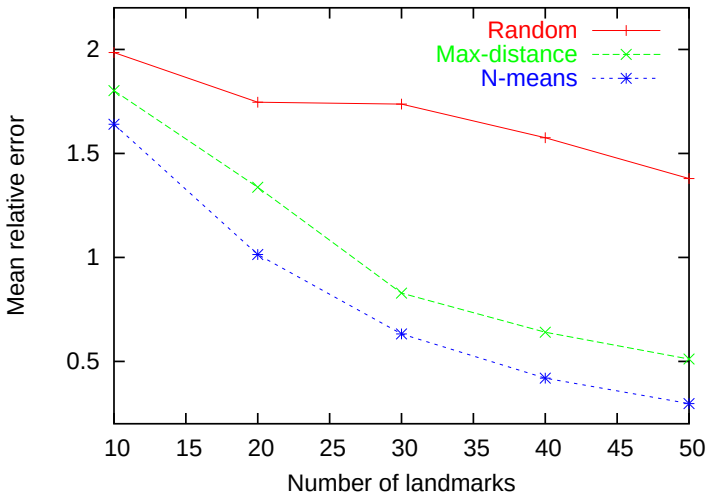
[6]We remind that this is an important requirement since we want to avoid having the bottleneck, of an indirect path, decided by the landmark itself.




(a) Data set 1

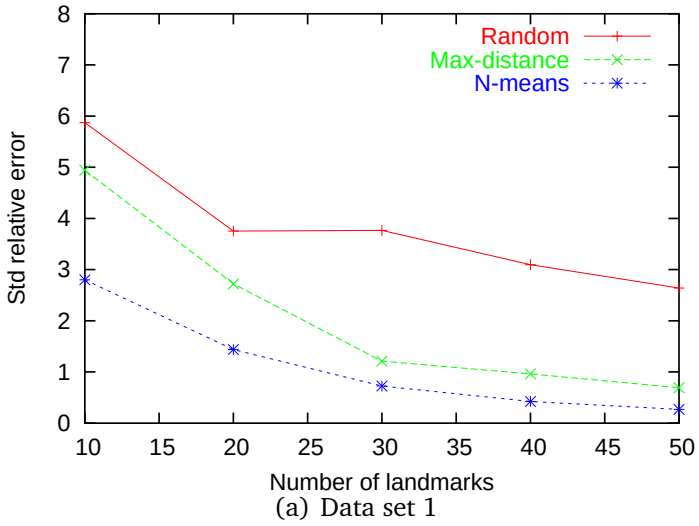



(b) Data set 2

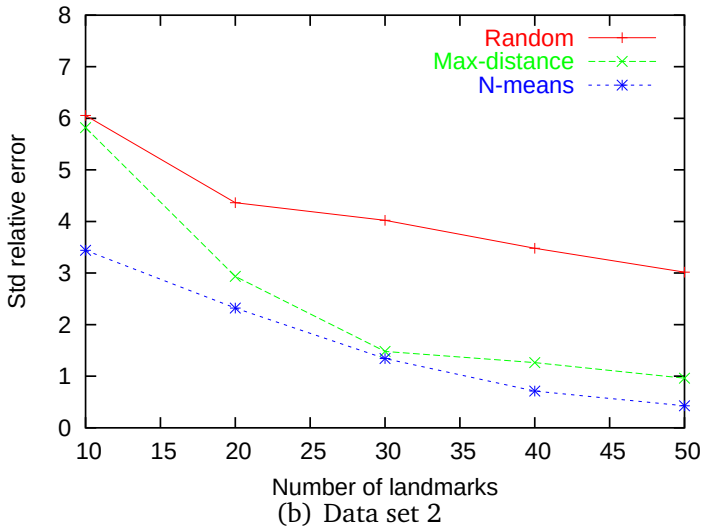



(c) Data set 3

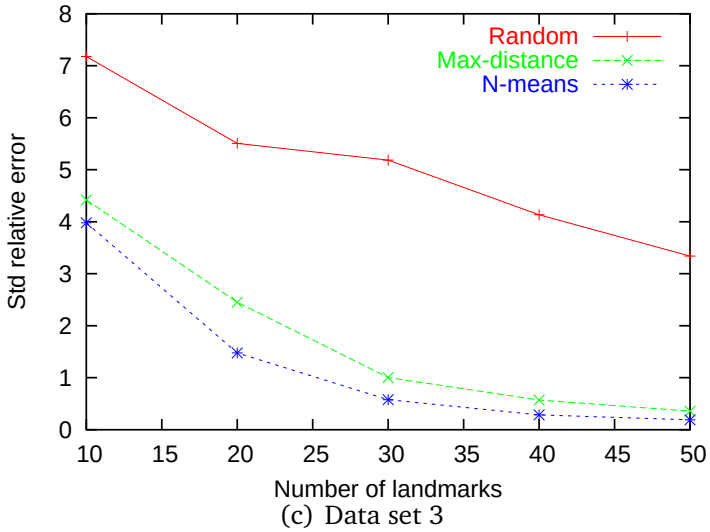
**Figure** 5.5: Mean estimation error obtained for different number of landmarks chosen in a random, max-distance, and N-means ways




(a) Data set 1




(b) Data set 2




(c) Data set 3

**Figure** 5.6: Standard deviation of the estimation error obtained for different number of landmarks chosen in a random, max-distance, and N-means ways

direct ones. Besides, the smaller bandwidth estimation error is obtained when using the N-means landmark sets. MRE (resp. stdRE) is between 1.3 and 1.6 (resp. between 3 and 4) for N equal 10 N-means landmarks and it decreases when N increases to become between 0.2 and 0.5 (resp. between 0.2 and 0.4) for N equal 50.

The estimations obtained when using the max-distance sets of landmarks are not as accurate as the case of the N-means sets of landmarks. In the former case, MRE (resp. stdRE) is between 1.6 and 2.1 (resp. between 5 and 7.1) for N equal 10 max-distance landmarks and it decreases when N increases to become between 0.5 and 0.9 (resp. between 0.4 and 1) for N equal 50. The worst case is obtained when using the random sets of landmarks. In this last case, MRE (resp. stdRE) is between 1.2 and 1.4 (resp. between 2.7 and 3.2) for N equal 50.

Moreover, we observe in the figures that it may happen that for some number of random landmarks, MRE and stdRE are larger than those obtained when a lower number of random landmarks is used. This can be caused by the fact that the smaller random set has a wider coverage than the larger random one. One may conclude that the distribution of the landmarks is as important as their number. A large number of randomly chosen set of landmarks may not be appropriate for bandwidth inference if they have a distribution covering a small portion of the network space while the peers cover all the space. In this case, a large number of direct and indirect paths will be disjoint and subsequently a large number of inaccurate estimations may occur. We note that when the indirect paths do not overlap with the direct one, the model provides accurate estimation only if the path bottleneck holds on the network connectivity of the peers. Hence, a random set of landmarks could be inappropriate for the bandwidth estimation model even if it contains a large number of nodes.

This is not the case when using the N-means landmark sets. N-means consists in grouping the peers into N clusters and then choosing the closest peers to the clusters' centroids as landmarks. Such delay minimization between landmarks and peers leads to better estimate the bandwidth on the paths joining peers as obtained in Section 5.4.2.

Thus, in our settings, a set of 40 to 50 landmarks having an N-means distribution is appropriate for estimating the bandwidth among a worldwide distributed set of peers. In the next section, we evaluate the performance gain achieved when considering the proximity in CHESS, determined using our bandwidth estimation model, instead of the delay proximity.

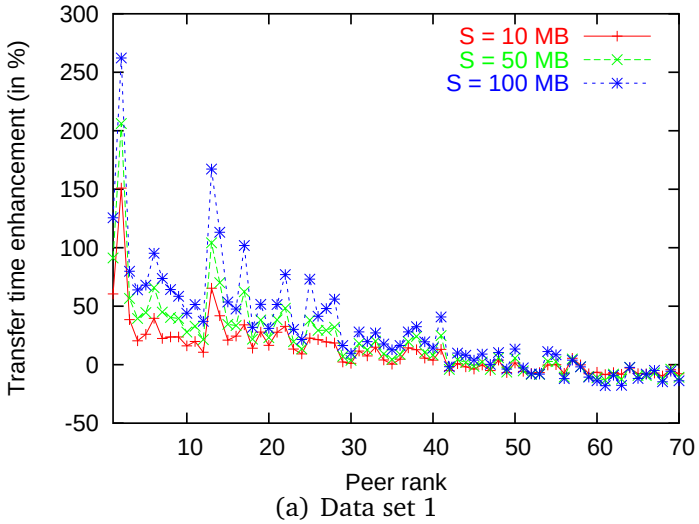
## 5.5 Enhanced proximity perceived by the application

From the standpoint of a certain peer, we recall that peers are ranked within the CHESS space in a decreasing order of the utility function. Thus, close peers in the CHESS space are those providing the best application quality despiting their network locality. In the previous chapter, we have obtained that the proximity in CHESS outperforms the basic delay proximity for different applications. This is obtained using the measured values of the network parameters. However, determining the proximity of peers using direct probing among them is not efficient due to its heavy overhead in large scale networks. Therefore, we use in this section our scalable bandwidth estimation model (described in Section 5.4.1) for characterizing the proximity in CHESS. On the other hand, we use the measured values of the delay and loss rate parameters in order to focus on the impact of our bandwidth estimation approach on the proximity characterization.

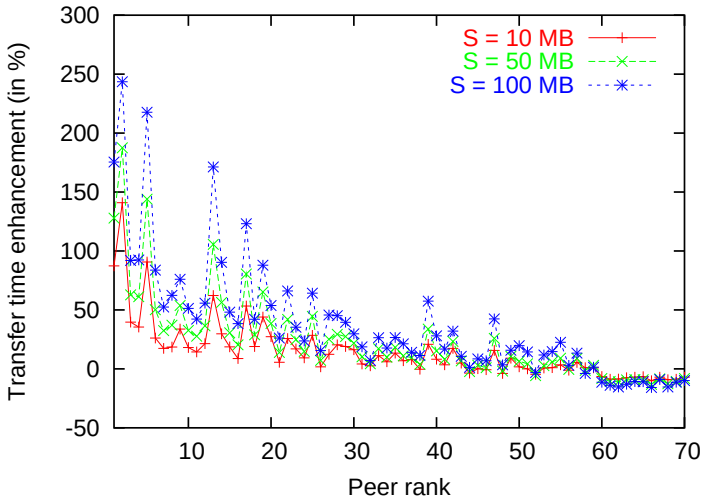
To this end, we consider as before a file transfer application over the TCP protocol. We consider the case of large TCP transfers due to its sensitivity to the different network parameters as we have obtained in the previous chapter. We evaluate the enhancement of the TCP latency when the proximity in CHESS is used instead of the delay proximity to perform the ranking of peers from the best to the worst.

This is computed similarly to Section 4.3.3 with the difference of using the bandwidth values estimated on the paths between a peer $p$ and the 86 other peers using the 40 N-means landmarks. Besides, we recall that we use the measured values of the delay RTT and the loss rate P instead of the landmark-based estimated ones in order to focus on the impact of our bandwidth estimation approach on the proximity characterization. We determine the latency enhancement for large files transfer (i.e., $S = \{50\text{MB}, 75\text{MB}, 100\text{MB}\}$) on each path. Then, we average all enhancement values at rank $r$ over the 87 peers. This study allows to evaluate how much the proximity in CHESS, determined using our bandwidth estimation model, outperforms the delay-based one on average at the application level.

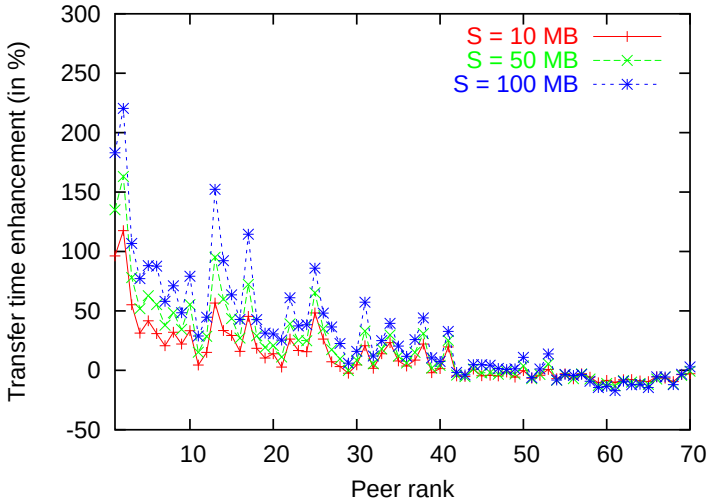
We plot in Figure 5.7 the transfer time enhancement as a function of the rank $r$ for the different file sizes. The enhancement is computed for our three datasets and drawn separately in Figure 5.7. The figure shows that, for low values of the rank $r$, the enhancement improves considerably when file size increases. It is between 50% and 200% when the transfer is achieved from the closest peer in the CHESS space. When the rank $r$ increases, the enhancement decreases and it becomes negative at some




(a) Data set 1




(b) Data set 2




(c) Data set 3

**Figure** 5.7: Transfer time enhancement when using the proximity in CHESS instead of the delay proximity

high ranks. As before, this is due to the fact that the peers having high ranks in the CHESS space (resp. delay space) are those having low ranks in the delay space (resp. CHESS space). The reason is that the paths with farer peers have better performance (e.g., larger available bandwidth) than those with close peers as obtained in Chapter 3. Hence, considering the delay alone for proximity characterization is far from being a good approximation of the proximity characterized in CHESS; this concerns long file transfer applications. The proximity in CHESS provides much better quality even when using the estimated values of the bandwidth parameter.

## 5.6 Conclusions

We show, in this chapter, that the proximity in CHESS can be deployed in large scale networks without complexity. This can done by inferring the network parameters, including the utility functions, in an easy and scalable way. While network embedding approaches may be appropriate for estimating the delay and loss parameters, it is not the case for the bandwidth parameter. Therefore, we propose a model that infers easily and scalably the bandwidth among peers using the bandwidth of the indirect paths that join them via a set of landmark nodes. Our model depends on the location of the landmarks with respect to the direct path and to the path endpoints. We find that a set of 40 to 50 landmarks could be enough for estimating the bandwidth among a world-wide distributed set of peers. Finally, we show that the proximity in CHESS, which is determined using our bandwidth estimation model, provides much better quality than that obtained when using the delay proximity for large file transfer applications.

# 6

# CONCLUSIONS AND FUTURE WORK

This dissertation has addressed the problem of proximity characterization among network nodes from application point of view. A better definition of this proximity can improve the quality of service perceived by clients for many applications. Particularly, for file sharing applications, this may lead to smaller transfer latency. In this chapter, we first summarize the contributions of this work. Then, we present some perspectives on our future research.

In Chapter 2, we presented an overview of the active approaches proposed in the literature for inferring the topology of the Internet. Then, we introduced in Chapter 3 the concept of application-aware space that we call CHESS. The characterization of the proximity in this space relies on the measurements of the network path characteristics and the prediction of application requirements. The need for this notion of proximity is deduced from the low correlation observed among path characteristics. Particularly, a proximity in the delay space does not automatically lead to a proximity in another space as the bandwidth one.

We described in Chapter 4 how application quality can be improved when proximity is defined using utility functions that model application performance. For file transfer applications, this is done using metric that estimates the transfer time by considering the critical parameters on the network paths together with limitations on the client and server sides. Also, we used the E-model for predicting the speech quality for interactive audio applications. For each application, we showed with Planetlab experiments how determining the proximity in the CHESS space instead of the delay-based one, permits to enhance the application quality perceived by peers.

In Chapter 5, we showed that the proximity in CHESS can be deployed in large scale networks at low complexity. This is done by inferring the network parameters, including the utility functions, in an easy and scalable way. While network embedding approaches may be convenient for estimating the delay and loss parameters, it is not the case for the bandwidth parameter. Therefore, we proposed a model that infers easily and scalably the bandwidth among peers using the bandwidth of the indirect paths that join them via a set of landmark nodes. Our model depends on the location of the landmarks with respect to the direct path and to the path endpoints. We found that a set of 40 to 50 landmarks could be enough for estimating the bandwidth among a worldwide distributed set of peers. Finally, we showed that the proximity in CHESS, which is determined using our bandwidth estimation model, provides much better quality than that obtained when using the delay proximity for large file transfer applications.

Regarding our future work, we plan to evaluate the profit of characterizing the proximity in CHESS for further applications. Particularly, we will consider the case of constructing an overlay multicast tree for video distribution. In such case, peers can be distributed along the tree not simply based on their geographical location or their delay closeness, but rather by considering the network parameters (e.g., peers' fan-in and fan-out limitations, network path characteristics) impacting the application quality. This will be examined using our scalable methods for estimating the network parameters.

On another venue, we will investigate further the problem of scalable bandwidth estimation. In particular, we will study the variability of bandwidth measurements and the persistency of bottlenecks [83]. This is important for determining the measurement frequency of the bandwidth vectors necessary for recomputing and updating the end-to-end bandwidth estimations.

# A

# PRÉSENTATION DES TRAVAUX DE THÈSE EN FRANÇAIS

## A.1 Introduction

Dans cette thèse, nous proposons une approche active qui caractérise de la meilleure façon la proximité entre les noeuds d'un réseau *Overlay*. Notre approche consiste à déterminer la proximité entre les noeuds du réseau au niveau applicatif en se basant sur des critères propres à l'application. Ceci est dans le but d'augmenter la qualité du service perçue par les utilisateurs. Nous commençons la dissertation en présentant, dans Chaiptre 2, une observation sur l'ensemble des approches actives qui ont été proposées dans la littérature pour l'inférence de la topologie du réseau. Puis, nous montrons la motivation, la description et l'expérimentation de notre proposition [1, 2, 3, 4, 5] dans les Chapitres 3, 4 et 5.

### A.1.1 Domaine de la thèse: Inférence de la topologie de l'Internet

L'énorme croissance du nombre des réseaux et des utilisateurs de l'Internet, ainsi que l'absence d'une administration centralisée, rendent la topologie de l'Internet de plus en plus complexe. Un grand nombre de routeurs sont installés dans le coeur et les périphéries de l'Internet dans le but de fournir un degré élevé de connectivité entre les différents noeuds (par exemple, utilisateurs, serveurs) qui sont répartit partout dans le monde. En outre, le contenu numérique devient partagé entre les utilisateurs de l'Internet et entreposé dans des serveurs dupliqués et distribués à travers l'Internet. Ceci vise à améliorer la qualité du service qui est offerte aux utilisateurs et de fournir une disponibilité élevée des données. Une telle duplique de l'information fait du choix de son endroit un problème défiant qui exige une connaissance de la topologie pour le rendre optimal.

Il y a deux approches principales pour inférer la topologie d'un réseau: l'approche active et l'approche passive. L'approche active est basée sur l'envoi des messages (par exemple, messages d'écho ICMP) entre les noeuds du réseau (par exemple, utilisateurs, serveurs, etc.) afin d'inférer la topologie du réseau (par exemple, [7, 8, 9]). En principe, l'approche active peut être utilisée pour inférer la topologie de tout l'Internet, d'un ISP particulier ou d'un AS particulier. En outre, l'approche active peut être aussi utilisée pour caractériser la proximité (par exemple, en termes du délai) entre les noeuds du réseau aussi bien que les paramètres de la performance sur les chemins qui les joignent. L'approche que nous proposons dans cette thèse appartient à la classe de l'approche active. Dans le chapitre suivant, nous passons en revue le corps de la littérature qui entre dans cette classe d'approches.

L'approche passive [10, 11, 12, 13, 14, 15, 16, 17] consiste à inférer la topologie du réseau sans y injecter de nouveaux paquets (les paquets de mesure qui sont utilisés dans l'approche active). L'approche passive sert à inférer la topologie en observant le

trafic de données qui circulent à travers le réseau. Ceci est fait en regardant les tables de routage (par exemple, BGP [18]), ou en observant les messages de contrôle échangés entre les routeurs. Cette classe d'approches n'est pas explorée dans cette thèse.

Nous définissons brièvement les entités principales du réseau qui sont fréquemment mentionnées dans la dissertation:

- Système Autonome (AS), qui est une collection de routeurs qui sont sous la même autorité administrative et qui utilisent 'Interior Gateway Protocol (IGP)' commun comme un protocole de routage de paquets.

- Fournisseur d'accès Internet (ISP), est le directeur et l'administrateur du réseau qui procure l'accès d'Internet à une partie ou à tous les utilisateurs d'un certain AS.

- Réseau CDN, où les demandes des clients sont distribuées par des machines relais à des serveurs miroirs stockant la même information et géographiquement répartis dans l'Internet. Beaucoup de compagnies, comme Akamai [19], procurent CDN aux fournisseurs d'information.

- Réseau Pair à Pair (e.g., BitTorrent [20]), où les pairs se comportent en tant que clients et serveurs.

- Réseau Overlay qui peut être un réseau Pair à Pair ou un réseau CDN.

## A.2 Contribution de la thèse: proximité applicative

L'utilisation répandue naissante des réseaux Pair à Pair et Overlay justifie la nécessité d'optimiser la performance perçue par les utilisateurs au niveau applicatif. Ceci revient à définir une fonction de proximité qui évalue combien deux pairs sont proches l'un de

l'autre. La caractérisation de la proximité aide à identifier le meilleur pair à contacter ou à prendre comme voisin.

Différentes fonctions ont été proposées dans la littérature pour caractériser la proximité entre les pairs, mais la plupart d'entre eux [8, 9, 21, 22, 23, 24, 25, 26, 27, 28, 29] sont basés sur une métrique simple telle que le délai, le nombre des sauts, et l'endroit géographique. Nous pensons que ces métriques ne sont pas optimales pour caractériser la proximité à cause de l'hétérogénéité de l'Internet en termes de caractéristiques des chemins, la vitesse des liens, et la diversité des conditions d'application. Quelques applications (par exemple, transfert de grands fichiers et service audio interactif) sont sensibles à d'autres paramètres de réseau comme la bande passante disponible et le taux de perte.

Par conséquent, la proximité doit être définie au niveau applicatif en prenant en compte les paramètres du réseau qui ont un impact sur la performance de l'application. À cet effet, nous présentons dans cette thèse la notion de *CHESS* (une abréviation de *an application-aware spaCe for enHancEd Scalable Services in overlay networks*), qui est un espace applicatif construit en fonction des besoins d'application. Dans ce nouvel espace, la proximité se caractérise selon une fonction d'utilité qui modélise la qualité de service perçue par les pairs au niveau applicatif. Un pair est plus proche qu'un autre d'un certain troisième pair s'il fournit une meilleure fonction d'utilité, même si le chemin qui le relie au troisième pair est plus long.

Nous commençons la dissertation en étudiant si la proximité du délai est une bonne approximation de la proximité dans CHESS [4]. Nous essayons de répondre à cette question par des mesures étendues effectuées au-dessus du réseau expérimental mondial *Planetlab*. À cet effet, nous considérons un grand ensemble de pairs et nous mesurons les caractéristiques des chemins qui les joignent. Nous considérons par la suite deux applications typiques: un transfert de fichier fonctionnant au-dessus du pro-

tocole TCP, et un service audio interactif. Pour chaque application, nous proposons une métrique qui modélise sa qualité en considérant les paramètres critiques du réseau (par exemple, le délai, la largeur de la bande passante disponible, le taux de perte) qui ont un impact sur la performance de l'application [5]. En outre, nous évaluons le perfectionnement de la performance perçue par les pairs quand ils choisissent leurs voisins en se basant sur la proximité dans CHESS, qui est déterminé en utilisant les fonctions de qualité proposées au lieu de celle basée sur le délai.

Notre observation principale est la suivante: le délai, la largeur de la bande passante disponible et le taux de perte sont légèrement corrélés, ce qui signifie qu'on ne peut pas baser sur un de ces métriques pour définir la proximité quand l'application est plus sensible aux autres. Par exemple, si nous utilisons le délai pour décider du pair le plus proche et ceci dans le but de le contacter pour un transfert de fichier, la performance de l'application dégrade par rapport au scénario optimal où les voisins sont identifiés en se basant sur la prédiction du temps de transfert des fichiers. En outre, si nous contactons le pair le plus proche en termes de délai pour un service audio interactif, la qualité de la parole n'est pas aussi bonne que celle obtenue quand le pair contacte celui avec qui on prévoit le meilleur facteur de qualité de la parole.

La contrainte principale pour déterminer la proximité dans CHESS est l'inférence des paramètres du réseau qui sont incluent dans la fonction de qualité d'une manière qui passe à l'échelle. En d'autres termes, l'estimation des paramètres de réseau sur le chemin joignant deux pairs quelconques dans un grand système devrait être réalisée avec la moindre quantité de mesure et une coopération limitée entre les pairs.

Nous commençons par montrer que la méthode de factorisation de matrice [30, 31], qui est proposé initialement pour estimer le délai, est appropriée pour estimer aussi le taux de perte. Cependant, ce n'est pas le cas pour la bande passante disponible qui n'est pas du tout une métrique additive. La bande passante disponible dépend du

goulot d'étranglement qui peut apparaître sur n'importe quel lien de chemin. Alors, un modèle d'estimation de la bande passante disponible doit identifier l'itinéraire reliant chaque couple de pairs pour pouvoir mesurer le lien de goulot d'étranglement. Le défi est qu'un tel modèle doit passer à l'échelle et être facile à déployer.

A partir de ce résultat, nous proposons un modèle pour estimer la bande passante disponible entre les pairs d'une façon qui passe à l'échelle [3]. Notre modèle estime la bande passante disponible sur les chemins directs (c-à-d, chemins IP) qui joignent les pairs en utilisant la bande passante disponible sur les chemins indirects. Ces chemins les relient par l'intermédiaire d'un ensemble de relais bien définis que nous appelons landmarks. En principe, les chemins direct et indirect se partagent le même goulot avec une certaine probabilité qui dépend de l'endroit du landmark correspondant par rapport au chemin direct ou à une des extrémités du chemin. Cette probabilité est plus haute si le landmark est plus proche d'une des extrémités de chemin. Elle peut être également plus haute si le délai du chemin indirect est plus proche de celui du chemin direct. C'est pourquoi nous supposons que la bande passante disponible sur chaque chemin indirect contribue à l'estimation de la bande passante du chemin direct suivant une certaine probabilité que nous définissons.

En utilisant encore des mesures sur Planetlab, nous évaluons la solution et analysons l'impact de l'endroit et du nombre des landmarks sur l'exactitude de l'estimation [1]. Nous découvrons que notre modèle d'estimation peut inférer exactement la bande passante disponible sur les chemins, ceux qui joignent un ensemble de pairs mondialement distribués en utilisant un nombre entre 40 et 50 landmarks.

Finalement, nous comparons la proximité caractérisée dans l'espace CHESS à celle obtenue dans l'espace du délai du point de vue application [2]. Une application typique de transfert de fichier est considérée pour évaluer la qualité du service perçue par les pairs quand ils choisissent leurs voisins en se basant sur ces deux notions distinguées de

proximité. Nous déterminons la proximité dans CHESS entre un ensemble de noeuds de Planetlab mondialement distribués en utilisant notre métrique [5] et ceci dans le but de prévoir le temps de transfert. Nous observons que la caractérisation de cette proximité, en utilisant notre modèle d'estimation de la bande passante, mène à une qualité bien meilleure que celle obtenue en utilisant seulement le délai.

### A.2.1 L'organisation de la thèse

La dissertation est structurée comme le suivant: Dans le chapitre suivant, nous présentons une vue d'ensemble des approches actives proposées dans la littérature pour inférer la topologie de l'Internet. Nous nous concentrons en particulier sur ceux qui sont basés sur les systèmes de coordonnés de réseau, et ceux qui utilisent les mesures de traceroute. Dans le Chaiptre 3, nous motivons d'abord le besoin de caractériser la proximité dans CHESS en étudiant la corrélation entre les différentes caractéristiques des chemins de réseau. Ensuite, nous expliquons comment une telle notion de proximité peut être particulièrement utile pour le choix du meilleur serveur et la construction du réseau Overlay.

Dans le Chapitre 4, nous établissons une comparaison entre la proximité dans l'espace de délai et celle caractérisée dans CHESS. Ceci est fait en évaluant l'impact de ces deux notions de proximité sur la performance de l'application. C'est pourquoi, nous considérons deux applications typiques: un transfert de fichier fonctionnant audessus du protocole TCP et un service audio interactif. Pour chaque application, nous développons d'abord la fonction d'utilité qui est capable de prédire la qualité d'application. Ensuite, nous évaluons le perfectionnement de la performance perçue par les pairs quand ils choisissent leurs voisins en se basant sur la proximité dans CHESS au lieu de celle basée sur le délai.

Dans le Chapitre 5, nous décrivons en premier lieu l'approche de factorisation de

matrice et étudions sa capacité pour estimer le délai, le taux de perte et la bande passante disponible. Nous proposons en second lieu un modèle pour estimer la bande passante disponible sur les chemins joignant les pairs d'une manière facile et passante à l'échelle. Dans la dernière partie de ce chapitre, nous évaluons le profit de performance réalisé quand nous considérons les estimations de la bande passante pour déterminer la proximité dans l'espace CHESS. En dernier lieu, nous concluons la dissertation dans le Chapitre 6 par quelques perspectives sur la future recherche dans ce secteur.

## A.3 Chapitre 2: Une vue d'ensemble des approches inférant la topologie

Inférer la topologie de l'Internet est un énorme domaine de recherche qui devient d'intérêt plus élevé avec l'évolution croissante de l'Internet. Il s'agit de fournir des informations spécifiques sur la topologie du réseau telles que la localisation des noeuds de réseau, la caractérisation de leur connectivité et la détermination de la performance sur les noeuds de réseau et sur les chemins qui les joignent. Ceci peut être réalisé selon les trois niveaux principaux du réseau: (i) le niveau de routeur, pour un ISP particulier ou pour tout l'Internet, (ii) le niveau de système autonome (AS) et (iii) le niveau applicatif.

L'inférence de la topologie de l'Internet est intéressante de différents points de vue comme présentés ci-dessous:

- Un utilisateur d'Internet (resp. pair) peut profiter de l'inférence de la topologie du réseau dans différents buts. Voici quelques exemples :
  - Identification du meilleur serveur (resp. le meilleur pair) pour télécharger un contenu en point à point ou le meilleur ensemble de serveurs (resp. les

meilleurs pairs) pour télécharger le contenu en parallèle.

- Optimisation de la construction du réseau Pair à Pair pour minimiser le temps de la recherche des contenus.
- Optimisation de la construction des arbres multicast applicatif pour la distribution d'un contenu (par exemple, distribution vidéo) afin d'augmenter la qualité du service perçue par les utilisateurs.
- Optimisation de la performance obtenue par les protocoles fonctionnant de bout en bout (par exemple, protocoles de collecte des données) [32].
- Identification du meilleur accès Internet quand il y en a plusieurs (comme le cas des machines ayant plusieurs interfaces IP).

■ Un fournisseur de service peut utiliser l'information inférée de la topologie pour améliorer la qualité de ses applications. Parmi les applications, nous pouvons trouver:

- Une meilleure distribution de ses serveurs à travers l'Internet suivant la distribution des clients qui demandent le service.
- Une meilleure politique pour pousser les documents et pour mettre à jour les mémoires 'caches'.
- Ajustez le déploiement de l'infrastructure d'un réseau Overlay pour améliorer la performance de la distribution du contenu (par exemple, distribution vidéo).
- Une meilleure re-direction des clients au meilleur serveur contenant l'information demandée.

■ Un ISP peut utiliser l'information de la topologie inférée pour optimiser son réseau. Nous mentionnons les objectifs principaux suivants:

- Analyser les protocoles de routage, la robustesse et la performance du réseau. Par exemple, nous mentionnons le bénéfice d'une restauration efficace au niveau IP: pour commuter l'envoi des paquets d'une interface à l'autre en se basant sur certains critères comme la panique des liens ou la dégradation de la qualité de service.
- Meilleur connexion avec les autres ISP, et un meilleur arrangement des tables de routage de BGP.
- Vérification si la qualité de service promis à un ISP voisin est satisfaisante.
- Découvert les préfixes qui sont responsables d'un comportement irrégulier tel qu'une attaque, une anomalie, etc.; avec connaître les endroits et les ISP de ces préfixes.
- Pour l'ingénierie du trafic à l'intérieur du réseau d'une ISP. Par exemple, nous mentionnons le bénéfice de localiser les points indésirables et de les éviter en rectifiant le routage du trafic par un autre chemin à l'intérieur du réseau.

■ Un chercheur scientifique peut se servir de la topologie du réseau pour étudier l'évolution de l'Internet et pour obtenir des scénarios plus réalistes dans ses recherches. Ceci peut aider à résoudre les problèmes réels comme ceux qui sont liés au comportement des protocoles du réseau.

Dans ce chapitre, nous présentons un briefing sur les approches actives qui ont été proposées dans la littérature pour l'inférence de la topologie du réseau. La structure du chapitre est comme suit. Tout d'abord, nous présentons une vue d'ensemble des outils actifs de mesure qui sont utiles pour l'inférence de la topologie du réseau. Ensuite, nous décrivons les différents travaux proposés qui peuvent être classifiés en tant que approches actives pour inférer la topologie du réseau. Nous nous concentrons en particulier sur ceux qui comptent sur les systèmes de coordonnés du réseau, et ceux

qui utilisent les mesures de traceroute. Finalement, nous concluons le chapitre dans Section 5.6.

## A.4 Chapitre 3: Motivation de la proximité dans CHESS

Dans les réseaux Pair à Pair et Overlay, la qualité du service, celle qui est perçue par les utilisateurs peut être optimisée au niveau applicatif en identifiant le meilleur pair à contacter ou à prendre comme voisin. Ceci appelle à définir une fonction de proximité qui évalue combien deux pairs sont proches l'un de l'autre du point de vue application.

Différentes fonctions sont présentées dans la littérature pour caractériser la proximité entre les pairs, mais la plupart d'entre eux [9, 23, 25, 28] sont basés sur des simples métriques telles que le délai, le nombre des sauts ou l'endroit géographique. Nous croyons que cette métrique n'est pas propre à caractériser la proximité vue l'hétérogénéité de l'Internet en termes de caractéristiques de chemins du réseau, et la diversité des conditions d'application. Quelques applications (par exemple, transfert de grands dossiers) sont sensibles à d'autres paramètres du réseau comme la bande passante disponible.

Par conséquent, la proximité devrait être définie au niveau applicatif avec la prise en compte des métriques du réseau qui décident de la performance de l'application. À cet effet, nous présentons la notion de l'espace CHESS. Dans ce nouvel espace, la proximité est caractérisée selon une fonction d'utilité qui modélise la qualité perçue par les pairs au niveau applicatif. Un pair est plus proche qu'un autre d'un certain troisième pair s'il est capable de fournir une meilleure fonction d'utilité, même si le chemin qui le lie au troisième pair est plus long.

En nous basant sur des mesures étendues au-dessus du réseau expérimental mondial Planetlab [69], nous motivons le besoin de notre nouvelle notion de proximité.

Nous montrons combien l'arrangement des pairs qui est fait suivant la proximité du délai dévie de celui réalisé en considérant d'autres paramètres du réseau (par exemple, la bande passante disponible, le taux de perte). Notre observation principale est la suivante. Le délai, la bande passante disponible et le taux de perte sont légèrement corrélés, ce qui signifie que, dans notre arrangement, nous ne pouvons pas se fonder sur une de ces métriques en définissant la proximité quand l'application est plus sensible aux autres. En Particulier, le meilleur pair à contacter n'est pas toujours le plus proche. Par conséquent, la connaissance des différents paramètres du réseau aide à améliorer la performance des applications et ceci en permettant de définir de meilleurs modèles de proximité.

Le chapitre est structuré comme le suivant. Tout d'abord, nous présentons la mise en oeuvre de nos mesures. Ensuite, dans Section 3.4, nous étudions la corrélation entre les différentes caractéristiques mesurées du réseau. Par la suite, nous présentons dans Section 3.5 quelques applications potentielles qui peuvent profiter de la caractérisation de la proximité dans CHESS. Finalement, nous concluons le chapitre dans Section 5.6.

## A.5 Chapitre 4: L'amélioration de la performance dû à la proximité dans CHESS

La caractérisation de la proximité dans CHESS nécessite la prédiction de la qualité perçue par les pairs. Ceci appelle à développer pour chaque application la fonction d'utilité correspondante qui modélise la qualité perçue par les pairs. Ceci peut être fait en considérant les paramètres critiques de la performance qui décident de la qualité de l'application.

Dans ce chapitre, nous considérons deux applications typiques: un transfert de fichier fonctionnant au-dessus du protocole de transport TCP, et un service audio in-

teractif. Pour l'application de transfert de fichier, nous développons une métrique qui correspond à une prédiction du temps de transfert d'un contenu du serveur au client. L'originalité de cette métrique est dans le fait qu'elle considère les caractéristiques des chemins (c.-à-d., le délai, la bande passante disponible, le taux de perte) entre les serveurs et le client ainsi que la charge du serveur et la fenêtre maximale de congestion du client. Nos résultats de mesure prouvent que notre métrique peut prévoir le temps de transfert des fichiers sur des chemins ayant de différentes performances avec une exactitude élevée. Pour le service audio interactif, nous utilisons le 'E-model' pour prévoir la qualité objective de la parole. L'efficacité du E-model est prouvée dans la littérature.

Pour chaque application, nous déterminons la proximité dans CHESS entre un grand nombre de noeuds de Planetlab en utilisant la fonction d'utilité correspondante. Ensuite, nous évaluons le perfectionnement de la performance perçue par les pairs quand ceux-ci choisissent leurs voisins en basant sur la proximité dans CHESS au lieu du celle basée sur le délai.

Nous remarquons que si nous utilisons le délai pour sélectionner le pair le plus proche à contacter pour un transfert de fichier, la performance de l'application se détériore par rapport au cas où les voisins sont identifiés en basant sur la prédiction du temps de transfert. En outre, si nous contactons le pair le plus proche en délai pour un service audio interactif, la qualité de la parole n'est pas aussi bonne que celle obtenue quand le pair à contacter est celui avec qui nous prévoyons avoir la meilleure qualité de la parole. Le même résultat se prolonge aux autres voisins au delà du plus proche.

Le chapitre est organisé comme le suivant. Dans la première partie du chapitre, nous explorons le cas de l'application de transfert de fichier. Dans Section 4.3.1, nous développons notre métrique pour la prédiction du temps de transfert. Puis, nous étudions l'impact de la définition de la proximité sur le temps de transfert dans Sec-

tion 4.3.3. Le cas du service audio interactif est élaboré dans Section 4.4. Finalement, nous concluons le chapitre dans Section 5.6.

## A.6 Chapitre 5: Déploiement à grande échelle de la proximité dans CHESS

La caractérisation de la proximité dans CHESS nécessite l'estimation des paramètres du réseau contenant la fonction d'utilité d'une manière facile et qui passe à l'échelle. En d'autres termes, l'estimation des paramètres de performance sur le chemin entre deux pairs quelconques dans un grand système, devrait être réalisée par un petit amont de mesure et une coopération limitée entre les pairs.

Alors qu'il y a récemment plusieurs modèles passant à l'échelle pour estimer le délai [8, 9, 21, 23, 24, 25, 26, 27], nous trouvons une seulement, appelée *BRoute* [7], pour estimer la bande passante disponible. BRoute suppose que les goulots d'étranglement du réseau sont situés sur les bords des chemins du réseau (c.-à-d., les premiers et derniers 4 liens d'un chemin IP complet) et sont partagés le plus probablement par de différents chemins. L'outil procède d'abord en mesurant la bande passante disponible sur les liens des bords. Il estime ensuite la bande passante disponible sur un chemin comme le minimum de la bande passante des liens de bord qu'elle identifie. L'identification des liens de bords d'un chemin joignant deux noeuds est réalisée par la détermination des arbres AS de source et de destination et par l'identifica- tion du plus court chemin AS, celui qui peut les joindre. Nous croyons que la réalisation de ces taches est un défi [81, 82, 14] et qu'elle ajoute beaucoup de complexité à l'approche.

Dans ce travail, nous commençons par prouver que l'approche de factorisation de matrice [30, 31], proposée pour estimer le délai d'un chemin complet entre deux noeuds, est propre à estimer aussi le taux de perte du chemin. Ce n'est pas le cas pour

la bande passante disponible qui est loin d'être une métrique additive. La bande passante disponible dépend du lien de goulot d'étranglement qui peut apparaître n'importe où sur le long d'un chemin complet. Un modèle d'estimation de la bande passante disponible doit identifier l'itinéraire joignant chaque couple de pairs pour pouvoir mesurer le lien de goulot d'étranglement. Le défi est qu'un tel modèle doit passer à l'échelle et être facile à déployer.

Par conséquent, nous proposons un modèle qui est capable d'estimer la bande passante disponible sur le chemin entre deux pairs. Ceci est fait d'une façon qui passe à l'échelle et qui n'exige pas la détermination des arbres AS de la source et de la destination. Elle ne suppose non plus que les goulots d'étranglement du réseau soient sur les bords des chemins. Notre modèle estime la bande passante sur les chemins joignant les pairs en utilisant la bande passante disponible sur les chemins indirects. Ces chemins les relient par l'intermédiaire d'un ensemble de noeuds bien définis que nous appelons des landmarks. En principe, les chemins direct et indirect se partagent le même goulot avec une certaine probabilité qui dépend de l'endroit du landmark correspondante par rapport au chemin direct ou à une des extrémités de chemin. Cette probabilité est plus haute si le landmark est plus proche d'une des extrémités du chemin. Elle peut être également plus haute si le délai du chemin indirect est plus proche de celui du chemin direct. Ainsi, la bande passante de tout chemin indirect contribue à l'estimation de la bande passante du chemin direct selon sa probabilité assignée. De cette façon, que le goulot d'étranglement soit sur les liens de bord du chemin joignant les deux pairs ou qu'il soit au milieu ne nous intéresse pas. D'ailleurs, un pair n'a pas besoin de déterminer ses arbres AS source et destination pour déduire les liens de bord qui le joignent aux autres pairs comme c'est le cas dans BRoute. Bien au contraire, les pairs doivent identifier les chemins indirects qui représentent le mieux les chemins directs. Cette tache est beaucoup plus facile à réaliser puisque nous proposons d'obtenir une

telle information en utilisant le délai des chemins qui relient les pairs aux landmarks.

En utilisant des mesures sur Planetlab, nous évaluons encore une fois la solution et analysons l'impact de l'endroit et du nombre de landmarks sur l'exactitude de l'évaluation. Nous obtenons que notre modèle d'estimation peut inférer exactement la bande passante entre un ensemble de pairs mondialement distribués en utilisant un nombre qui est entre 40 et 50 landmarks.

Finalement, nous déterminons la proximité entre les pairs dans les espaces du délai et du CHESS afin d'évaluer leur impact sur la performance d'application. Une application typique de transfert de fichier est considérée pour évaluer la qualité du service, celle qui est perçue par les pairs quand ils choisissent leurs voisins en se basant sur ces deux notions distinguées de proximité. Nous déterminons la proximité dans CHESS entre un ensemble de noeuds de Planetlab mondialement distribué en utilisant notre fonction de prédiction de temps de transfert. Nous observons que la caractérisation de cette proximité, en utilisant notre modèle d'estimation de la bande passante, mène à une qualité bien meilleure que celle obtenue en utilisant juste le délai.

Le chapitre est organisé comme le suivant. Tout d'abord, nous décrivons l'approche de factorisation de matrice que nous utilisons pour estimer le délai et le taux de perte d'une façon qui passe à l'échelle. Puis, nous présentons dans Section 5.4 un modèle pour estimer la bande passante disponible d'une façon qui passe aussi à l'échelle. Dans Section 5.5, nous évaluons le gain de la performance obtenue quand nous utilisons les estimations de la bande passante disponible pour déterminer la proximité dans l'espace CHESS. Finalement, nous concluons le chapitre dans Section 5.6.

## A.7 Chapitre 6: Conclusions et travaux futurs

Cette dissertation a soulevé le problème de la caractérisation de la proximité entre les noeuds du réseau de point de vue application. Une meilleure définition de cette proximité peut améliorer la qualité du service, celle qui est perçue par les utilisateurs pour beaucoup d'applications. En particulier, pour les applications de partage des fichiers, ceci peut mener au temps de transfert le plus réduit. Dans ce chapitre, nous récapitulons d'abord les contributions de ce travail. Ensuite, nous présentons quelques perspectives sur notre future recherche.

Dans Chapitre 2, nous avons présenté une vue d'ensemble des approches actives proposées dans la littérature pour inférer la topologie de l'Internet. Puis, nous avons défini dans Chapitre 3 le concept de l'espace CHESS qui peut être construit en fonction de l'application. La caractérisation de la proximité dans cet espace est basée sur les mesures des paramètres de performance sur les chemins du réseau et la prédiction de la fonction d'utilité de l'application. Le besoin de cette notion de proximité est déduit de la légère corrélation observée entre les différents paramètres de performance. En particulier, une proximité dans l'espace du délai ne mène pas automatiquement à une proximité dans un autre espace comme c'est le cas de celle qui est basée sur la bande passante disponible.

Nous avons expliqué dans Chapitre 4 comment la qualité de l'application peut être améliorée quand la proximité est définie par l'utilisation des fonctions d'utilité qui modélisent la performance de l'application. Pour les applications de transfert de fichier, cela est fait par l'emploi d'une métrique qui estime le temps de transfert en considérant les paramètres de performance critiques sur les chemins du réseau ainsi que les limitations aux côtés du client et du serveur. En outre, nous avons utilisé le E-model pour prévoir la qualité de la parole pour des applications audio interactives. Pour chaque application, nous avons montré avec des expérimentations sur Planetlab comment la

détermination de la proximité dans l'espace CHESS et non pas celle basée sur le délai, fournit une amélioration de la qualité de l'application, celle qui est perçue par les pairs.

Dans Chapitre 5, nous avons prouvé que la proximité dans CHESS peut être déployée dans des réseaux à grande échelle avec une légère complexité. Ceci est fait par l'inférence des paramètres de performance du réseau, ceux qui contiennent les fonctions d'utilités d'une manière facile et qui passe à l'échelle. Alors que les approches qui se basent sur les systèmes de coordonnées peuvent être propres à estimer le délai et le taux de perte, la bande passante disponible ne possède pas cette faculté. Par conséquent, nous avons proposé un modèle qui peut estimer facilement la bande passante disponible sur les chemins directs joignant les pairs en utilisant la bande passante disponible sur les chemins indirects qui les joignent par l'intermédiaire d'un ensemble de landmarks. Notre modèle dépend de l'endroit des landmarks par rapport au chemin direct et aux extrémités du chemin. Nos résultats montrent qu'un ensemble de 40 à 50 landmarks peut être suffisant pour estimer la bande passante entre un ensemble de pairs mondialement distribués. Finalement, nous avons montré que la proximité dans CHESS, qui est déterminée en utilisant notre modèle d'évaluation de la bande passante, fournit une qualité bien meilleure que celle obtenue par l'utilisation de la proximité de délai pour des applications de transfert de grands fichiers.

Pour nos futurs travaux, nous évaluerons le bénéfice de la caractérisation de la proximité dans CHESS pour d'autres applications. Nous considérerons en particulier le cas de la construction d'un arbre applicatif pour la distribution d'un contenu vidéo. En ce cas, les pairs peuvent être distribués tout au long de l'arbre non seulement en se basant sur leur endroit géographique ou la proximité de délai, mais aussi et surtout en considérant les paramètres critiques du réseau (par exemple, la bande passante de la connexion réseau des pairs, et les caractéristiques sur les chemins du réseau), ceux qui ont un impact sur la qualité de l'application. Ceci sera examiné en utilisant notre

méthode d'estimation de la bande passante.

Par ailleurs, nous étudierons plus profondément le problème de l'estimation de la bande passante. Nous étudierons en particulier la variabilité des mesures de la bande passante et de la persistance des goulots [83]. Ce sera important pour déterminer la fréquence de mesure des vecteurs de la bande passante qui est nécessaire pour déterminer de nouveaux et mettre à jour les estimations de la bande passante disponible.

# BIBLIOGRAPHY


[1] M. Malli, C. Barakat, and W. Dabbous. *CHESS: An Application-aware Space for Enhanced Scalable Services in Overlay Networks*. Submitted for publication, 2006. vii, 1, 4, 99, 104

[2] M. Malli, C. Barakat, and W. Dabbous. *An Enhanced Scalable Proximity Model*. In IEEE International Workshop on Quality of Service (short paper), 2006. vii, 1, 5, 99, 104

[3] M. Malli, C. Barakat, and W. Dabbous. *Landmark-based End-to-End Bandwidth Inference*. In IEEE Infocom Student Workshop, 2006. vii, 1, 4, 99, 104

[4] M. Malli, C. Barakat, and W. Dabbous. *Application-level versus Network-level Proximity*. In Asian Internet Engineering Conference, 2005. vi, 1, 3, 99, 102

[5] M. Malli, C. Barakat and W. Dabbous. *An Efficient Approach for Content Delivery in Overlay Networks*. In IEEE Consumer Communications and Networking Conference, 2005. vi, vii, 1, 3, 5, 99, 103, 105

[6] Matthias Scheidegger, Torsten Braun, Florian Baumgartner. *Endpoint Cluster Identification for End-to-End Distance Estimation*. in IEEE International Conference on Communications, 2006. 2

[7] N. Hu, P. Steenkiste. *Exploiting Internet Sharing for Large Scale Available Bandwidth Estimation*. In ACM Internet Measurement Conference, 2005. 2, 72, 81, 82, 100, 112

[8] B. Wong, A. Silvkins, and E. G. Sirer. *Meridian: A Lightweight Network Location Service without Virtual Coordinates*. In ACM SIGCOMM, 2005. vi, xiii, 2, 3, 13, 24, 25, 72, 100, 102, 112

[9] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. *Vivaldi: A Decentralized Network Coordinate System*. In ACM SIGCOMM, 2004. vi, 2, 3, 13, 14, 30, 32, 72, 100, 102, 109, 112

[10] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley, *Inferring TCP Connection Characteristics Through Passive Measurements*. In IEEE Infocom, 2004. 2, 100

[11] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, *Measuring the Effects of Internet Path Faults on Reactive Routing*. In Sigmetrics, 2003. 2, 100

[12] Y. Zhang and L. Breslau and V. Paxson and S. Shenker. *On the characteristics and origins of internet flow rates*. In ACM SIGCOMM, 2002. 2, 100

[13] D. Andersen, N. Feamster, S. Bauer, and H. Balakrishnan. *Topology Inference from BGP Routing Dynamics*. In ACM Internet Measurement Workshop, 2002. 2, 100

[14] L. Gao. *On Inferring autonomous system relationships in the Internet*. In Transactions on Networking, 2001. 2, 72, 100, 112

[15] H. Jiang and C. Dovrolis. *Passive estimation of TCP round-trip times*. Technical report, 2001. 2, 100

[16] H. Martin, A. McGregor, and J. Cleary. *Analysis of Internet delay times*. In Passive and Active Measurement Workshop, 2000. 2, 100

[17] K. Thompson, G. Miller and R. Wilder. *Wide-area internet traffic patterns and characteristics*. In IEEE Network Magazine, 1997. 2, 100

[18] Y. Rekhter and T. Li. *A Border Gateway Protocol 4*. RFC 1771, 1995. 2, 101

[19] Akamai, `http://www.akamai.com`. 2, 101

[20] BitTorrent, `http://www.bittorrent.com`. 2, 101

[21] Y. Shavitt and T. Tankel. *On the Curvature of the Internet and its usage for Overlay Construction and Distance Estimation*. In IEEE Infocom, 2004. vi, 3, 13, 71, 72, 102, 112

[22] Y. Shavitt and T. Tankel. *Big-bang simulation for embedding network distances in Euclidean space*. In IEEE Infocom, 2003. vi, 3, 102

[23] L. Tang and M. Crovella. *Virtual Landmarks for the Internet*. In ACM Internet Measurement Conference, 2003. vi, 3, 13, 14, 30, 32, 71, 72, 102, 109, 112

[24] H. Lim, J. Hou, and C. Choi. *Constructing Internet Coordinate System Based on Delay Measurement*. In ACM Internet Measurement Conference, 2003. vi, 3, 13, 14, 72, 102, 112

[25] E. Ng and H. Zhang. *Predicting Internet network distance with coordinates-based approaches*. In IEEE Infocom, 2002. vi, 3, 13, 14, 30, 71, 72, 102, 109, 112

[26] S. Ratnasamy and M. Handly and R. Karp and S. Shenker. *Topologically-Aware Overlay Construction and Server Selection*. In IEEE Infocom, 2002. vi, 3, 13, 21, 22, 71, 72, 102, 112

[27] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang. *IDMaps: A Global Internet Host Distance Estimation Service*. In IEEE/ACM Transactions on Networking, 2001. vi, 3, 13, 24, 72, 102, 112

[28] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida. *Constraint-Based Geolocation of Internet Hosts*. In ACM Internet Measurement Conference, 2004. vi, 3, 25, 30, 32, 102, 109

[29] V. Padmanabhan and L. Subramanian. *An Investigation of Geographic Mapping Techniques for Internet Hosts*. In ACM SIGCOMM, 2001. vi, 3, 25, 32, 102

[30] Y. Mao, and L. Saul, and J. Smith: IDES: An Internet Distance Estimation Service for Large Networks, to appear in IEEE JSAC, 2006. vii, 4, 15, 72, 73, 76, 103, 112

[31] Y. Mao, and L. Saul. *Modelling distances in large-scale networks by matrix factorization*. In ACM Internet Measurement Conference, 2004. vii, 4, 14, 15, 72, 73, 76, 79, 103, 112

[32] Chadi Barakat, Mohammad Malli, and Naomichi Nonaka. *TICP: Transport Information Collection Protocol*. In Annals of Telecommunications Journal, vol. 61, no. 1-2, pp. 167-192, 2006. 8, 107

[33] TCPdump, `http://www.tcpdump.org/`. 10, 11

[34] K. Claffy, T. Monk, and D. McRobb. *Internet tomography*. In Nature, January 1999. 16

[35] S. Moon and T. Roscoe. *Metadata Management of Terabyte Datasets from an IP Backbone Network: Experience and Challenges*. In Workshop on Network-Related Data Management, 2001. 10, 11

[36] S. Katti, D. Katabi, C. Blake, E. Kohler, and J. Strauss, *A Passive Toolkit for Measuring, Tracking, and Correlating Path Characteristics*. Technical report, 2004. 10, 11

[37] J. Case, M. Fedor, M. Shoffstall, and J. Davin. *Simple Network Management Protocol*. RFC 1157, 1990. 10, 11

[38] Netflow. See `http://www.cisco.com/en/US/products/ps6601/products_ios_protocol_group_home.html`. 10, 11

[39] N. Hu and P. Steenkiste. *Evaluation and Characterization of Available Bandwidth Techniques*. In IEEE JSAC Special Issue in Internet and WWW Measurement, Mapping, and Modeling, 2003. 12

[40] J. Strauss, D. Katabi and F. Kaashoek. *A Measurement study of Available Bandwidth Estimation Tools*. In ACM Internet Measurement Conference, 2003. 12

[41] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell. *PathChirp: Efficient Available Bandwidth Estimation for Network Paths*. In Passive and Active Measurement Workshop, 2003. 12

[42] K. Lai and M. Baker. *Nettimer: A Tool for Measuring Bottleneck Link Bandwidth*. In Usenix, 2001. 12

[43] B. Melander, M. Bjorkman, and P. Gunningberg. *A New End-to-End Probing and Analysis Method for Estimating Bandwidth Bottlenecks*. In Global Internet Symposium, 2000. 12

[44] V. Ribeiro and M. Coates, R. Riedi, S. Sarvotham and R. Baraniuk. *Multifractal Cross Traffic Estimation*. In ITC, 2000. 12

[45] V. Jacobson. *Pathchar: A Tool to Infer Characteristics of Internet Paths*. `ftp://ftp.ee.lbl.gov/pathchar/`, 1997. 12

[46] S. Keshav. *A Control-Theoretic Approach to Flow Control*. In ACM SIGCOMM, pages 3-15, September 1991. 12

[47] R. Carter and M. Crovella. *Dynamic Server Selection using Bandwidth Probing in Wide-Area Networks*. Technical Report, Boston University, 1996.

[48] R. L. Carter and M. E. Crovella. *Measuring Bottleneck Link Speed in Packet-Switched Networks*. In Performance Evaluation, vol. 27,28, pp. 297-318, 1996. 12

[49] G. Jin, G. Yang, B. Crowley, and D. Agarwal. *Network Characterization Service (NCS)*. In the 10th IEEE Symposium on High Performance Distributed Computing, 2001. 12

[50] M. Jain and C. Dovrolis. *End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput*. In ACM SIGCOMM, Aug. 2002, pp. 295-308.

[51] M. Jain and C. Dovrolis. *Pathload: A Measurement Tool for End-to-End Available Bandwidth*. In Passive and Active Measurements, 2002. 12

[52] J. Nelder, and R. Meed. *A Simplex Method for Function Minimization*. In Computer Journal, 7:308-313, 1965. 14

[53] M.Costa, M. Castro, A. Rowstron, and P. Key. *PIC: Practical Internet Coordinates for Distance Estimation*. In ICDCS, 2004. 14

[54] S. Banerjee, T. Griffin, and M. Pias. *The interdomain connectivity of Planetlab nodes*. In Passive and Active Measurement Workshop, 2004. 14

[55] K. Lakshminarayanan, and V. Padmanabhan. *Some findings on network performance of broadband hosts*. In ACM Internet Measurement Conference, 2003. 14, 24

[56] E. Lua, T. Griffin, M. Pias, H. Zheng, and J. Crowcroft. *On the Accuracy of Embeddings for Internet Coordinate Systems*. In ACM Internet Measurement Conference, 2005. 14

[57] H. Zheng, E. Lua, M. Pias, T. Griffin. *Internet Routing Policies and Round-Trip-Times*. In Passive and Active Measurement Workshop, 2005. 14, 24

[58] N. Spring and R. Mahajan and D. Wetherall. *Measuring ISP Topologies with Rocketfuel*. In ACM SIGCOMM, 2002. 16, 18

[59] K. Claffy and T. Monk and D. McRobb. *Internet tomography*. In Nature, 1999. 16

[60] Cooperative Association for Internet Data Analysis, `http://www.caida.org/`. xiii, 16, 18, 53

[61] R. Govindan and H. Tangmunarunkit. *Heuristics for Internet map discovery*. In IEEE Infocom, 2000. 20

[62] S. Hotz. *Routing information organization with heterogeous path requirements*. Ph.d. thesis, 1994. 23

[63] S. Ratnasamy, P. Francis, M. Handly, R. Karp, and S. Shenker. *A Scalable Content-Addressable Network*. In SIGCOMM, 2001. 22

[64] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, Hari Balakrishnan. *Chord: A Scalable Peertopeer Lookup Service for Internet Applications*. In SIGCOMM, 2001. 24

[65] A. Rowstron and P. Druschel. *Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems*. In IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), 2001. 24

[66] B. Zhao, J. Kubiatowicz, and A. Joseph. *Tapestry: An infrastructure for fault-tolerant wide-area location and routing*. Technical Report UCB/CSD-01-1141. UC Berkeley. April 2001. 24

[67] P. Enge and P. Misra. *Special issue on global positioning system*. Proceedings of the IEEE, 87(1):3-15, Jan. 1999. 25

[68] L. Peterson, V. Pai, N. Spring, and A. Bavier. *Using PlanetLab for Network Research: Myths, Realities, and Best Practices*. Technical report, 2005. 31

[69] An open, distributed platform for developing, deploying and accessing planetary-scale network services, see `http://www.planet-lab.org/`. 30, 109

[70] J. Navratil, and L. Cottrell, `http://www-iepm.slac.stanford.edu/tools/abing/`, 2004. 31

[71] J. Navratil, L. Cottrell. *ABwE: A Pratical Approach to Available Bandwidth Estimation*. In Passive and Active Measurement Workshop, 2003. 31

[72] J. Navratil, `http://www.slac.stanford.edu/~jiri/PLANET`, 2004. 31

[73] A. Edwards, *An Introduction to Linear Regression and Correlation*, 1976. 33

[74] ITU-T Recommendation G.107. The E-model, a computational model for use in transmission planning, 2000. 45, 65

[75] Leonard Kleinrock, *Queueing systems volume I: theory*, 1975. 54

[76] OpenBSD FTP, `http://www.openbsd.org/ftp.html`. 60

[77] R. Cole, and J. Rosenbluth. *Voice over IP Performance Monitoring*. In ACM SIGCOMM Computer Communication Review, v. 31 , p. 9 - 24, 2001. 65, 66

[78] L. Ding, and R. Goubran. *Speech Quality Prediction in VoIP Using the Extended E-Model*. In IEEE Globecom, 2003. 65

[79] ITU-T Recommendation G.711. Pulse Codec Modulation (PCM) of voice frequencies, 1988. 66

[80] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. *Modeling TCP Throughput: a Simple Model and its Empirical Validation*. In proceedings of ACM SIGCOMM, 1998. 58

[81] Z. Mao, L. Qiu, J. Wang, and Y. Zhang. *On AS-level Path Inference*. In Sigmetrics, 2005. 72, 112

[82] G. Battista, M. Patrignani, and M. Pizzonia. *Computing the types of the relationships betweeen autonomous systems*. In IEEE Infocom, 2003. 72, 112

[83] A. Akella and S. Seshan and A. Shaikh. *An Empirical Evaluation of Wide-Area Internet Bottlenecks*. In ACM Internet Measurement Conference, 2003. 98, 117

# ABSTRACT

We introduce in this thesis the notion of *CHESS*, an application-aware space for enhanced scalable services in overlay networks. In this new space, the proximity is a function of network parameters (e.g., delay and bandwidth) that decide on the application performance. We motivate the need for this new notion by showing that the network parameters are slightly correlated.

Then, we consider two typical applications: a file transfer running over the TCP protocol, and an interactive audio service. For each application, we first propose a metric that models the application quality by considering the critical network parameters (e.g., delay, bandwidth, loss rate) affecting the application performance. Then, we evaluate the enhancement of the performance perceived by peers when they choose their neighbors based on the proximity in CHESS instead of the delay-based one determined using the proposed utility functions.

Our major contribution is a model for inferring the bandwidth among peers in an easy and scalable manner. It consists of estimating the bandwidth among peers using the bandwidth of the indirect paths that join them via a set of well defined proxies or relays that we call landmark nodes. Our idea is that an indirect path shares the same tightest link with the direct path with a probability that depends on the location of the corresponding landmark with respect to the direct path or any of the two peers subject to bandwidth inference. We evaluate the impact of the location, number, and distribution of the landmarks on the bandwidth estimation accuracy. We obtain that the proximity in CHESS, which is determined using our bandwidth estimation model, provides much better quality than that obtained using the delay proximity for large file transfer applications. The whole study is supported by extensive measurements carried out over the worldwide experimental network 'Planetlab'.



**Keywords**: Network Topology, Overlay Network, Measurement, Performance

# RÉSUMÉ

Nous introduisons dans cette thèse la notion de proximité applicative. Cette proximité est fonction des paramètres du réseau (par exemple, délai, bande passante, taux de perte) qui affectent les performances de l'application. Nous justifions le besoin pour cette nouvelle notion par la légère corrélation des paramètres réseau que nous avons observée sur l'Internet.

Dans la première partie de la thèse, nous considérons deux applications typiques qui sont le transfert des fichiers au dessus du protocole TCP, et un service audio interactif. Pour chaque application, nous proposons d'abord une métrique qui modélise sa qualité en fonction des paramètres critiques du réseau. Puis, nous évaluons l'amélioration de la qualité perçue par les pairs utilisateurs de ces applications lorsqu'ils choisissent leurs voisins en fonction de la proximité applicative que nous proposons au lieu de celle basée sur le délai.

Notre majeure contribution est un modèle qui sert à déployer la proximité applicative d'une façon qui passe à l'échelle. Ceci passe par une estimation de la bande de passante entre deux pairs, le délai étant fait par les autres. Le modèle consiste à estimer la bande passante entre les pairs en utilisant celles des chemins indirects qui les connectent via un ensemble de relais bien définis que nous appelons *landmarks*. Notre idée est qu'un chemin indirect partage le même goulot que le chemin direct avec une probabilité qui dépend de l'endroit où se trouve le landmark correspondant par rapport au chemin direct. Nous évaluons l'impact de l'endroit, du nombre, et de la distribution des landmarks sur l'exactitude des estimations de la bande passante. Notre étude montre que la proximité déterminée par notre modèle d'estimation de la bande passante fournisse une meilleure qualité applicative que celle obtenue en utilisant la proximité de délai et ceci pour les applications de transfert des grands fichiers, le transfert des grands fichiers étant une application plus sensible à la bande passante qu'au délai. Les expérimentations qui ont servi à cette étude sont basées sur des mesures approfondies effectuées au dessus du réseau expérimental mondial *Planetlab*.

**Mots-clés**: La Topologie du Réseau, Réseau Applicatif, Mesure, Performance