Rating Network Paths for Locality-Aware Overlay Construction and Routing

Wei Du, Yongjun Liao, Narisu Tao, Pierre Geurts, Xiaoming Fu and Guy Leduc

***Abstract*—This paper investigates the rating of network paths, i.e. acquiring quantized measures of path properties such as round-trip time and available bandwidth. Comparing to fine-grained measurements, coarse-grained ratings are appealing in that they are not only informative but also cheap to obtain.**

Motivated by this insight, we firstly address the scalable acquisition of path ratings by statistical inference. By observing similarities to recommender systems, we examine the applicability of solutions to recommender system and show that our inference problem can be solved by a class of matrix factorization techniques. A technical contribution is an active and progressive inference framework that not only improves the accuracy by selectively measuring more informative paths but also speeds up the convergence for available bandwidth by incorporating its measurement methodology.

Then, we investigate the usability of rating-based network measurement and inference in applications. A case study is performed on whether locality awareness can be achieved for overlay networks of Pastry and BitTorrent using inferred ratings. We show that such coarse-grained knowledge can improve the performance of peer selection and that finer granularities do not always lead to larger improvements.

***Index Terms*—rating-based network measurement, network inference, recommender system, matrix factorization.**

I. Introduction

Network measurement is a fundamental problem in the heart of the networking research. Over the years, various tools have been developed to acquire path properties such as round-trip time (RTT), available bandwidth (ABW) and packet loss rate, etc [1]. As in most scientific disciplines, the common sense in the field is that a measurement should be made as fine-grained and accurate as possible. This is considered necessary to enable quantitative analysis of network performance.

However, recent advances in emerging Internet services have created numerous situations where coarse-grained measurements can be leveraged and may even be preferred. A typical example is Peer-to-Peer (P2P) Overlay Networks [2] where a common design is to use *Intelligent Peer Selection* to improve traffic locality [3], i.e. encourage more communications between "nearby" nodes, with connections of small RTT or high ABW. The objective is thus to find "good-enough" paths for overlay construction and routing, which can be well served by using coarse-grained network measurement.

Motivated by this insight, this paper investigates the rating of network paths, i.e. acquiring quantized path properties represented by ordinal numbers of $1, 2, 3, \ldots$, with larger value indicating better performance. Although coarse-grained, ordinal ratings are appealing for the following reasons:

- Ratings carry sufficient information that already fulfills the requirements of many applications.
- Ratings are rough measures that are cheaper to obtain than exact property values.
- Ratings can be encoded in a few bits, saving storage and transmission costs.

A practical issue of rating-based network measurement is the efficient acquisition on large networks. While cheap for a single path, it is still infeasible to rate all paths in a network by active probing due to the quadratic complexity. The scalability issue has been successfully addressed by statistical inference that measures a few paths and predicts the properties of the other paths where no direct measurements are made [4]–[12]. Inspired by these studies, a particular focus of this paper is **network inference of ratings**: how ratings of network paths can be accurately predicted. An interesting observation is that the inference problem resembles the problem of *recommender systems* which studies the prediction of preferences of users to items [13]. If we consider a path property as a "friendship" measure between end nodes, then intelligent peer selection can be viewed as a "friend" recommendation task. This seemingly trivial connection has the great benefit to leverage the rapid progresses in machine learning and investigate the applicability of various solutions to recommender systems for network inference. Our studies show that a class of matrix factorization techniques are suitable for network inference and achieved good results that are known to be acceptable for recommendation tasks.

Furthermore, we also observe an important difference between the two inference problems. Unlike recommender systems where users rate items voluntarily, we have more control over data acquisition in network inference, i.e. nodes may actively select a few other nodes to probe. This insight enables the design of an *active* inference framework that, under a given measurement budget, improves prediction accuracy by selectively measuring those more informative paths. In addition, a

This work was partially supported by the EU under project FP7-Fire ECODE, FP7-ICT mPlane and FP7-ICT GreenICN, by a grant from German Academic Exchange Service (DAAD) and a grant from the Lindemann Foundation, Germany, and by the European Network of Excellence PASCAL2 and the Belgian network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office.

Wei Du, Yongjun Liao and Guy Leduc are with Research Unit in Networking (RUN), University of Liège, Belgium. Email: {wei.du, yongjun.liao, guy.leduc}@ulg.ac.be.

Narisu Tao and Xiaoming Fu are with Computer Networks, University of Göttingen, Germany. Email: {narisu.tao, xiaoming.fu}@informatik.uni-goettingen.be.

Pierre Geurts is with Systems and Modeling, University of Liège, Belgium. Email: p.geurts@ulg.ac.be.

progressive inference framework is designed for a particular path property, namely ABW, by utilizing the principle of *self-induced congestion* [1] which has been widely incorporated in popular ABW measurement tools such as Pathload and Pathchirp [14], [15]. This allows us to perform inference using partial rating information (i.e., the rating of a path is larger or smaller than a value), which significantly speeds up the convergence of the inference.

Another practical issue on rating-based network measurement is **the usability in applications**. Two questions need to be answered, the first of which is whether the inference of ratings is accurate enough to be exploited by applications and the second of which is how to determine a proper granularity. While a coarser granularity means rougher and thus cheaper measurement, it also means more information losses which may hurt the performance of applications. Answers to these questions are critical in the design of system architecture, particularly for P2P applications where the knowledge of locality plays an important role [3], [16], [17].

Thus, we answer these two questions by investigating quantitatively the impacts of both the inaccuracy of the inference and the granularity. For the case study, we consider locality-aware overlay construction and routing where locality refers to the proximity between network nodes according to some path property such as RTT or ABW. More specifically, we performed the study on Pastry [16] and BitTorrent [17], which are typical structured and unstructured overlay networks and are known to enjoy the property of locality awareness, and evaluated the performance of overlay construction and routing, with the knowledge of locality obtained via network inference of ratings. Our studies show that while the knowledge of inferred ratings can improve the performance of peer selection, finer granularities do not always lead to larger improvements. For example, our simulations on various datasets show that the performance of peer selection improves very little when the rating level reaches 2^4.

Thus, this paper makes the following contributions:

- We investigate the rating-based network measurement that acquires quantized path properties represented by ordinal numbers. Such representation not only is informative but also reduces measurement, storage and transmission costs.
- We investigate the scalable acquisition of ratings by network inference. We highlight similarities between network inference and recommender systems and examine the applicability of solutions from this latter domain to network inference. In particular, we show that our inference problem can be solved by a class of matrix factorization techniques.
- We design the active and progressive inference schemes that improve not only the prediction accuracy but also the convergence speed.
- We perform a case study on locality-aware overlay construction and routing to demonstrate the usability of rating-based network measurement and inference in P2P applications.

The rest of the paper is organized as follows. Section II introduces related work. Section III describes the properties and the rating of network paths. Section IV introduces network inference by a class of matrix factorization techniques. Section V describes an active and progressive inference framework. Section VI introduces the case study on locality-aware overlay construction and routing. Section VII gives conclusions and future work.

II. Related Work

A. Inference of Network Path Properties

Network inference has been studied for over a decade, and numerous approaches have been developed. For example, GNP [4] and Vivaldi [5] solved the inference of RTT by Euclidean embedding, while tomography-based approaches such as TOM [6], Network Kriging [7] and NetQuest [8] used linear algebraic methods and the routing tables to infer additive properties such as RTT and packet loss rate. Although interesting in different aspects, these approaches are limited only to certain path properties and/or rely on the routing information of the network which is expensive to obtain for applications.

To overcome these constraints, a recent advance was made in [10], [12] that formulated network inference as a matrix completion problem. A class of algorithms called DMFSGD was proposed and showed improved accuracy and flexibility in dealing with various properties including the non-additive ABW. Matrix completion is suitable for network inference because it exploits the correlations between measurements of different paths [6], [7], [12], [18] and has also been used for other problems such as network traffic estimation [19]–[21].

All above-mentioned work focused on the prediction of exact measurement values, except [10] which classified path properties into binary classes of either “good” or “bad”. This paper goes further and is the first to investigate rating-based network measurement and inference. Ratings are measures in between property values and binary classes: on the one hand, they are more informative than binary classes; on the other hand, they are advantageous over property values due to the coarse-grained measurement.

B. Locality-Aware Overlay Networks

There are two classes of overlay networks: structured and unstructured [2]. The former places content deterministically at specified locations according to the Distributed Hash Table (DHT) which makes subsequent queries efficient. The latter organizes peers more randomly and uses flooding or random walks to query content. Examples of structured systems are Kademlia [22], Chord [23] and Pastry [16], and those of unstructured are Freenet [24], Gnutella [25] and BitTorrent [17].

For both structured and unstructured overlay networks, a desired property is locality-awareness which makes communications as local as possible. Such a property is important for P2P applications because it can reduce cross-domain traffic, avoid congestions and improve service performance [3]. It is recognized that locality-awareness can be achieved via intelligent peer selection whereby dense connections are enforced between nodes that are well connected to each other, i.e., short in terms of RTT or wide in terms of ABW.

While the effectiveness of intelligent peer selection has been repeatedly justified [3], [26], [27], this paper is the first to study whether locality-awareness can be achieved, in both structured and unstructured overlay networks, using rough and inaccurate knowledge of network proximity obtained via network inference of ratings.

III. Properties and Rating of Network Paths

A. Properties of Network Paths

While ultimately the performance of a network path should be judged by the quality of service (QoS) perceived by end users, it is important to define an objective metric that is directly measurable without user interventions. Among the commonly-used path properties mentioned earlier, this paper only discusses round-trip time (RTT) and available bandwidth (ABW) due to the availability of such data and the relatively rare occurrence of packets losses in the Internet [1].

1) Round-Trip Time (RTT): The RTT is one of the oldest and most commonly-used network measurements due partially to its ease of acquisition by using ping with little measurement overhead, i.e., few ICMP echo request and response packets between the sender and the target node. Although the one-way forward and reverse delays are not exactly the same due e.g. to the routing policies, the RTTs between two network nodes can be treated as symmetric.

2) Available Bandwidth (ABW): The ABW is the remaining capacity of the bottleneck link on a path. Many comparative studies have shown that tools based on self-induced congestion are generally more accurate [1], [28]. The principle states that if the probing rate exceeds the available bandwidth over the path, then the probe packets become queued at some router, resulting in an increased transfer time. The ABW can then be estimated as the minimum probing rate that creates congestions or queuing delays. To this end, pathload [14] sends trains of UDP packets at a constant rate and adjusts the rate from train to train until congestions are observed at the target node, while pathchirp [15] reduces the probe traffic by varying the probe rate within a train exponentially. The ideas of the two tools are illustrated in Figure 1.

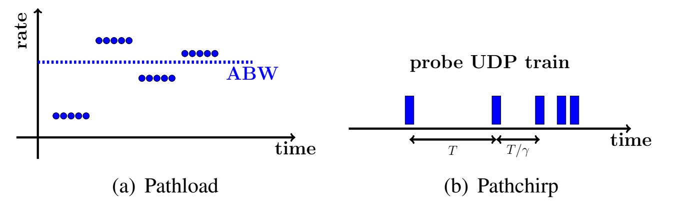


Fig. 1. Two popular ABW measurement tools. In (a), Pathload sends UDP trains (blue dots) at a constant rate, and increases the probing rate if no congestion is observed and decreases otherwise, until convergence to the ABW (dashed line). In (b), Pathchirp varies the probing rate within a train (blue rectangles) exponentially, i.e., the packets in a train are spaced exponentially.

Comparing to the RTT, the ABW measurement is much more costly and less accurate. For example, many studies revealed the pitfalls and the uncertainties of the ABW measurement [28]–[30]. In this paper, we ignore these measurement issues and investigate how network inference can be performed[1] for ABW using a publicly available dataset [31] in which the measurements were acquired by pathchirp, a fairly accurate and widely-used tool [28]. In contrast to the RTT, the ABW is asymmetric and its measurement is inferred at the target node.

B. Rating of Network Paths

Generally, ratings can be acquired by vector quantization that partitions the range of the path property into R bins using $R-1$ thresholds, denoted by $\tau = \{\tau_1, \ldots, \tau_{R-1}\}$, and determines to which bins property values belong, as illustrated in Figure 2. The thresholds can be chosen evenly or unevenly according to the data distribution or the requirements of applications. For example, Google TV requires a broadband speed of 2.5Mbps or more for streaming movies and 10Mbps for High Definition contents [32]. Accordingly, $\tau = \{2.5\text{Mbps}, 10\text{Mbps}\}$ can be defined for ABW to separate paths of rating 1, 2 and 3.

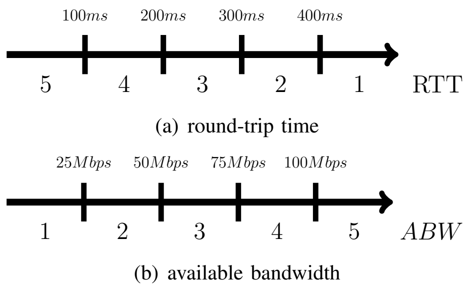
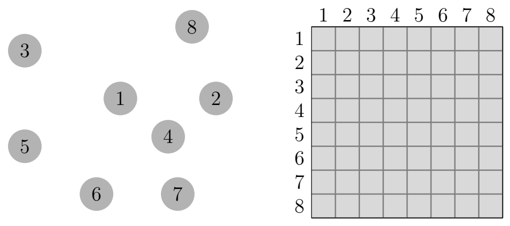


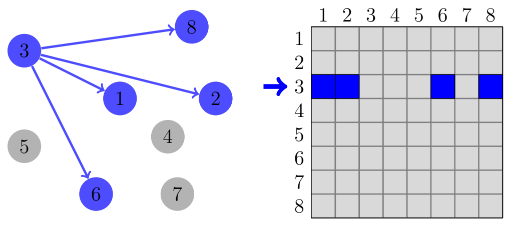
Fig. 2. Rating by quantization. The thresholds are chosen as an example.

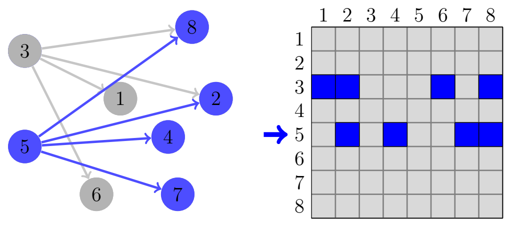
Rating by quantization is directly applicable to RTTs whose measurements are cheap. For the ABW, it is preferable to directly obtain the rating measures without the explicit acquisition of the values in order to reduce measurement overhead. We will show how this is possible by utilizing the principle of self-induced congestion in Section V-B. In general, as data acquisition undergoes the accuracy-versus-cost dilemma, stating that accuracy always comes at a cost, coarse-grained measurements are always cheaper to obtain than fine-grained ones. The advantage of the low measurement cost holds particularly for ABW whose measurement is costly.

Note that when $R = 2$, rating degenerates to binary classification of network paths which was studied in [10]. When R is very large, rating approaches the measurement of property values, which was studied in [12]. Thus, R results from a trade-off between the granularity and the measurement costs. A larger R means finer granularity but at the cost of more measurement overheads, and vice versa. Nevertheless, we will fix $R = 5$ in Section IV where we examine the applicability of solutions to recommender systems. Such a choice is natural because it corresponds to the commonly-used 5-star rating system which categorizes performance into 5 discrete levels of “very poor”, “poor”, “ordinary”, “good” and “very good” quality . We will study the impact of different choices of R in Section VI.

[1]Strictly speaking, we do not infer the property of a path. Instead, we infer the property value that should have been obtained by some measurement tool.


(a) A network of 8 nodes.


(b) Node 3 probes node 1, 2, 6 and 8.


(c) Node 5 probes node 2, 4, 7 and 8.

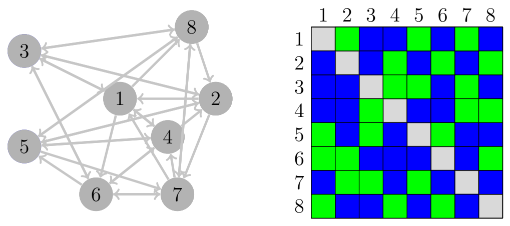

(d) A matrix with some missing entries is formed.

Fig. 3. A matrix completion view of network inference. In (d), the blue entries are measured path properties and the green ones are missing. Note that the diagonal entries are empty as they represent the performance of the path from each node to itself which carries no information.

IV. Network Inference of Ratings

This section describes the scalable acquisition of ratings on large networks by statistical inference.

A. Problem Statement

Formally, the inference problem is stated as follows. Consider a network of n end nodes, and define a path to be a routing path between a pair of end nodes. There are $O(n^2)$ paths in the network, and we wish to monitor a small subset of paths so that the performance of all other paths can be estimated. Note that here we seek to infer ratings, instead of values, of path properties. Besides reducing the measurement overheads, another motivation of network inference is that we can estimate the properties of those paths which are not measurable due e.g. to the lack of experimental controls.

This inference problem has a matrix completion form where a partially observed matrix is to be completed [33]. In this context, a matrix, denoted by X, is constructed, with its entry x_{ij} being the rating of the path from node i to node j. Each node randomly probes a few other nodes and measures the ratings of the paths to them. The measurements are put in the corresponding entries of X, and the missing entries are the ratings of those unmeasured paths and need to be estimated, illustrated in Figure 3. Note that X is a square matrix when it contains the performance of all pairwise paths in a network. However, for situations where one set of nodes in a network probe another set of nodes in the same or a different network, X becomes non-square. Such situations arise in content distribution networks (CDNs) where a set of users probe a set of servers [34].

In order for network inference to be feasible, network measurements on different paths have to be correlated with or dependent on each other. Otherwise, the inference would be impossible or suffer from large errors, regardless the inference schemes used. The correlations between network measurements exist, because Internet paths with nearby end nodes often overlap or share bottleneck links due to the simple topology of the Internet core. The redundancy of link usage across paths causes the performance of many paths to be correlated [6]–[8]. For example, the congestion at a certain link causes the delays of all paths that traverse this link to increase jointly. A direct consequence of such correlations is that the related performance matrix occurs to have a low-rank characteristic [12], [18] which further enables the inference problem to be solved by matrix factorization techniques, shown in the following sections. The low-rank characteristic of performance matrices of RTT and ABW is illustrated by the spectral plots in Figure 4. It can be seen that the singular values of these matrices decrease fast, which is an empirical justification of the low-rank phenomenon.

B. Connections to Recommender Systems

The problem of network inference bears strong similarities to the problem of recommender systems which predicts the preference of users to items such as music, books, or movies [13], [36]. Table I shows an example of a recommender system. Consider that network nodes are users and that each node treats other nodes as items or “friends”. The performance of a network path is then actually a “friendship” measure of how one end node would like to contact the other end node

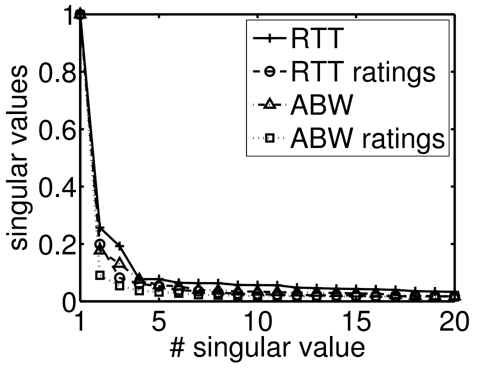


Fig. 4. The singular values of a 2255×2255 RTT matrix, extracted from the Meridian dataset [35], and a 201×201 ABW matrix, extracted from the HP-S3 dataset [31], and of their corresponding rating matrices. The rating matrices are obtained by thresholding the measurement matrices with $\tau = \{20\%, 40\%, 60\%, 80\%\}$ percentiles of each dataset. The singular values are normalized so that the largest ones equal 1.

of the path. In particular, the task of intelligent peer selection can be viewed as a "friend" recommendation task. Guided by this insight the rapid advances made for recommender systems can be applied to our inference problem.

TABLE I
AN EXAMPLE OF A RECOMMENDER SYSTEM.

	item1	item2	item3	item4	item5	item6
user1	5	3	4	1	?	?
user2	5	3	4	1	5	2
user3	5	?	4	1	5	3
user4	1	3	2	5	1	4
user5	4	?	4	4	4	?

In particular, research on recommender systems was largely motivated by the *Netflix* prize which was an open competition initiated in 2006 for algorithms that predict the preference, quantized by the 5-star rating system, of users to movies [37]. A grand prize of one million dollar was to be given to the first algorithm that improves the accuracy of Netflix's own algorithm *cinematch* by 10%. The prize had attracted a lot of efforts and attempts before it was given to the BellKor's Pragmatic Chaos team in 2009 which achieved an improvement of 10.21%. In what follows, the prize-winning solution is called *BPC*. Specifically, BPC integrated two classes of techniques based on neighborhood models and on matrix factorization. Neighborhood models exploit the similarities between users and between items. For example, two users are considered similar if they rate a set of items similarly. Meanwhile, two items are considered similar if they are given similar ratings by a set of users. Although interesting, neighborhood models are infeasible for network inference, because, to compute the similarities between network nodes, they must probe a number of common nodes, which is impossible when the nodes being probed are randomly selected[2]. Thus, we focus below on the matrix factorization models in BPC.

[2]However, it is possible to deploy some landmark nodes in the network for other nodes to probe. Thus, each node knows its performance information to/from the common landmarks. In such situations, the neighborhood models become applicable and are interesting to study, which is left as future work.

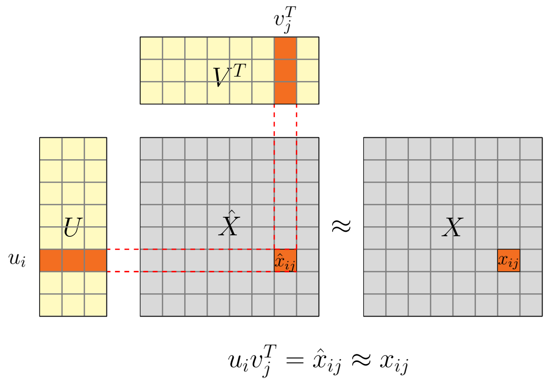


Fig. 5. Matrix factorization.

C. Matrix Factorization

Matrix factorization (MF) exploits the low-rank nature of matrices of real-world data. Mathematically, a n by n matrix of rank r, where $r \ll n$, has only r non-zero singular values and it can be factorized as

$$X = UV^T, \tag{1}$$

where U and V are matrices of size $n \times r$. In practice, due to data noise, X is often full-rank but with a rank r dominant component. That is, X has only r significant singular values and the other ones are negligible. In this case, a rank-r matrix $\hat{X}$ can be found that approximates X with high accuracy, i.e.,

$$X \approx \hat{X} = UV^T. \tag{2}$$

MF can be used for solving the problem of matrix completion, which generally minimizes an objective function of the following form:

$$\min \sum_{ij \in \Omega} l(x_{ij}, u_i v_j^T), \tag{3}$$

where Ω is the set of observed entries, x_{ij} is the ijth entry of X, and u_i and v_j are the ith and jth row of U and of V respectively. l is a loss function that penalizes the difference between the two inputs. In words, we search for (U, V) so that $\hat{X} = UV^T$ best approximates X at the observed entries in Ω. The unknown entries in X are predicted by

$$\hat{x}_{ij} = u_i v_j^T, \text{ for } ij \notin \Omega. \tag{4}$$

Note that $\hat{x}_{ij}$ is real-valued and has to be rounded to the closest integer in the range of $\{1, R\}$ for ordinal rating. Figure 5 illustrates MF for matrix completion.

Below, we introduce various MF models that were integrated in BPC including RMF, MMMF and NMF.

1) RMF: Regularized matrix factorization (RMF) [13] adopts the widely-used square loss function and solves

$$\min \sum_{ij \in \Omega} (x_{ij} - u_i v_j^T)^2 + \lambda \sum_{i=1}^{n} (u_i u_i^T + v_i v_i^T). \tag{5}$$

The second term is the regularization which restricts the norm of U and V so as to prevent overfitting, and λ is the regularization coefficient.

2) MMMF: Max-margin matrix factorization (MMMF) solves the inference problem by ordinal regression [38] which relates the real-valued estimate $\hat{x}_{ij}$ to the ordinal rating x_{ij} by using $R-1$ thresholds $\{\theta_1, \ldots, \theta_{R-1}\}$. More specifically, MMMF requires the following constraint to be satisfied for each x_{ij}, $ij \in \Omega$,

$$\theta_{c-1} < \hat{x}_{ij} = u_i v_j^T \leqslant \theta_c, \quad \text{for } x_{ij} = c, \quad 1 \leqslant c \leqslant R. \quad (6)$$

For simplicity of notation $\theta_0 = -\infty$ and $\theta_R = +\infty$. In words, the value of $\hat{x}_{ij}$ does not matter, as long as it falls in the range of $\{\theta_{c-1}, \theta_c\}$ for $x_{ij} = c$, $1 \leqslant c \leqslant R$. Here we set the thresholds as $\{1.5, 2.5, 3.5, 4.5\}$ for $R = 5$. Thus, the constraint in eq. 6 means that, for example, if $x_{ij} = 2$, then it is required that $1.5 < \hat{x}_{ij} < 2.5$ so that $\hat{x}_{ij}$ will be rounded to 2. Whether $\hat{x}_{ij}$ is 2, 2.2 or 1.6 makes no difference.

Thus, we penalize the violation of the constraint in eq. 6 for each x_{ij} and minimize the following objective function

$$\min \sum_{ij \in \Omega} \sum_{c=1}^{R-1} l(T_{ij}^c, \theta_c - u_i v_j^T) + \lambda \sum_{i=1}^{n} (u_i u_i^T + v_i v_i^T), \quad (7)$$

where $T_{ij}^c = 1$ if $x_{ij} \geqslant c$ and -1 otherwise. Essentially, the objective function in eq. 7 consists of $R-1$ binary classification losses, each of which compares an estimate $\hat{x}_{ij}$ with a threshold θ_c in $\{\theta_1, \ldots, \theta_{R-1}\}$. For example, for $x_{ij} = 2$, it is required that $\hat{x}_{ij} > 1.5$ and $\hat{x}_{ij} < 2.5$, $\hat{x}_{ij} < 3.5$, $\hat{x}_{ij} < 4.5$. Each violation of these four constraints is penalized by $l(T_{ij}^c, \theta_c - u_i v_j^T)$, with

$$T_{ij}^c = \begin{cases} -1 & \text{for } c = 1 \\ 1 & \text{for } 2 \leqslant c \leqslant 4 \end{cases} \quad (8)$$

indicating the correct sign of $(\theta_c - u_i v_j^T)$.

Note that l in eq. 7 can be any classification loss function, among which the smooth hinge loss function is used [38], defined as

$$l(x, \hat{x}) = \begin{cases} 0 & \text{if } x\hat{x} \geqslant 1 \\ \frac{1}{2}(1 - x\hat{x})^2 & \text{if } 0 < x\hat{x} < 1 \\ \frac{1}{2} - x\hat{x} & \text{if } x\hat{x} \leqslant 0 \end{cases} \quad (9)$$

3) NMF: Non-negative matrix factorization (NMF) [39] incorporates an additional constraint that all entries in U and V have to be non-negative so as to ensure the non-negativity of $\hat{X}$. Besides, NMF uses the divergence as the loss function, defined as

$$D(X||\hat{X}) = \sum_{ij \in \Omega} (x_{ij} \log \frac{x_{ij}}{\hat{x}_{ij}} - x_{ij} + \hat{x}_{ij}). \quad (10)$$

Thus, NMF solves

$$\min D(X||UV^T) + \lambda \sum_{i=1}^{n} (u_i u_i^T + v_i v_i^T), \text{ s.t. } (U, V) \geqslant 0. \quad (11)$$

4) MF Ensembles: Instead of using one of these MF models, BPC integrated all of them in an ensemble framework, which is the root of its success. The idea is to learn multiple predictors simultaneously, by using different MF models (RMF, MMMF and NMF) and by setting different parameters for each MF model, and combines their outputs, by voting or averaging, for prediction [40], [41]. Intuitively, the power of ensemble methods comes from the "wisdom of the crowd", which says that a large group's aggregated answer to a question is generally found to be as good as, and often better than, the answer given by any of the individuals within the group [42].

Besides improving accuracy, ensemble models allow to compute the variance of each prediction, by computing the variance of the predictions by different predictors. Variance indicates the uncertainty of the prediction: large variance means that different predictors disagree with each other and we are thus less certain about the combined prediction[3]. Such information can be exploited in e.g. intelligent peer selection so that we can choose, among nodes with high ratings, those with small variance as they are more certain. Moreover, it further enables the design of an active inference framework, which will be introduced in Section V.

D. MF for Network Inference

1) Inference By Stochastic Gradient Descent: We adopted Stochastic Gradient Descent (SGD) for solving all MF models. In short, we pick x_{ij} in Ω randomly and update u_i and v_j by gradient descent to reduce the difference between x_{ij} and $u_i v_j^T$. SGD is particularly suitable for network inference, because measurements can be acquired on demand and processed locally at each node. We refer the interested readers to [10], [12] for the details of the inference by SGD.

2) Neighbor Selection: We also adopted the common architecture that each node randomly selects k nodes to probe, called neighbors in the sequel. That is, each node measures the properties of the paths from itself to the k neighbors and predicts the other paths by using MF-based inference schemes.

The choice of k is the result of a trade-off between accuracy and measurement overhead. On one hand, increasing k always improves accuracy as we measure more and infer less. On the other hand, the more we measure, the higher the overhead is. Thus, we vary k for networks of different sizes so as to control the number of paths being monitored to be a certain percentage of the total number of paths in a network. In particular, we require k to be no smaller than 10 so that a certain amount of information about the performance at each node is guaranteed. This leads to less than 5% of available measurements for a network of about two hundred nodes, which is the smallest dataset we used in the paper. For large networks of a few thousand nodes, we increase k so that about 1% of the paths are monitored. As the largest dataset we used in this paper has less than 5000 nodes, k is no larger than 50. We consider that such setting of k leads to sparse available measurements that is affordable for large networks.

3) Rank r: The rank r is an important parameter and can only be determined empirically. On the one hand, r has to be large enough so that the dominant components in X are kept. On the other hand, a higher-rank matrix has less redundancies and requires more data to recover, increasing measurement

[3]While we build our confidence on the variance of a prediction, it does not mean that smaller variance leads to better accuracy. Instead, it means that the combined prediction makes more sense if the variance is small, i.e. if different predictors agree with each other.

overheads. Our experiments show that empirically, $r = 10$ is a good choice, given the sparse available measurements.

E. Comparison of Different MF Models

This section compares different MF models on our network inference problem. In the evaluations, we set $R = 5$, which was also used in the Netflix prize.

The comparison was then performed on the following publicly available datasets:

- **Harvard** contains dynamic RTT measurements, with timestamps, between 226 Azureus clients deployed on PlanetLab [43];
- **Meridian** contains static RTT measurements between 2500 network nodes obtained from the Meridian project [35];
- **HP-S3** contains static ABW measurements between 231 PlanetLab nodes [31].
- **YouTube** contains static RTT measurements from 441 PlanetLab nodes to 4816 YouTube servers [34].

In the simulations, the static measurements in Meridian, HP-S3 and YouTube are used in random order, whereas the dynamic measurements in Harvard are used in time order according to the timestamp of each measurement.

We adopted the evaluation criterion, *Root Mean Square Error* (RMSE), given by

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}{n}}. \tag{12}$$

which was used in the Netflix prize. As RMSE is the average estimation error, the smaller it is, the better.

1) Obtaining Ratings: The first step is to obtain ratings on a scale of $\{1, 5\}$ from the raw measurements. To this end, the range of a path property is partitioned by the rating threshold $\tau = \{\tau_1, \ldots, \tau_4\}$. τ is set by two strategies:

- Strategy 1: set τ by the $\{20\%, 40\%, 60\%, 80\%\}$ percentiles of each dataset.
  - Harvard: $\tau = \{48.8, 92.2, 177.2, 280.3\}$ms
  - Meridian: $\tau = \{31.6, 47.3, 68.6, 97.9\}$ms
  - HP-S3: $\tau = \{12.7, 34.5, 48.8, 77.9\}$Mbps
  - YouTube: $\tau = \{38.1, 91.1, 131.3, 192.4\}$ms
- Strategy 2: partition evenly the range between 0 and a large value manually selected for each dataset.
  - Harvard: $\tau = \{75, 150, 225, 300\}$ms
  - Meridian: $\tau = \{25, 50, 75, 100\}$ms
  - HP-S3: $\tau = \{20, 40, 60, 80\}$Mbps
  - YouTube: $\tau = \{50, 100, 150, 200\}$ms

2) Results: Throughout the paper, the MF parameters are set as follows: for RMF, MMMF and NMF, the regularization coefficient $\lambda = 0.1$ and the rank $r = 10$. For MF ensembles, we generated for each MF model (RMF, MMMF and NMF) 6 predictors using different parameters, i.e. r ranges from 10 to 100 and λ ranges from 0.01 to 1, as described in [41]. For the neighbor number, $k = 10$ for Harvard of 226 nodes, leading to about 4.42% available measurements; $k = 32$ for Meridian of 2500 nodes, leading to about 1.28% available measurements; $k = 10$ for HP-S3 of 231 nodes, leading to about 4.33% available measurements; $k = 50$ for YouTube of 4816 servers, leading to about 1.04% available measurements. Thus, we collect k measurements at each node and perform the inference using different MF models. The evaluation was done by comparing the inferred ratings of those unmeasured paths with their true ratings, calculated by RMSE defined above.

Table II shows the RMSEs achieved by different MF models and on different datasets. We can see that while RMF generally outperforms MMMF and NMF, MF ensembles perform the best at the cost of more computational overheads due to the maintenance of multiple MF predictors. Note that all MF models achieved fairly accurate results with the RMSE less than 1. In comparison, for the dataset in the Netflix prize, the RMSE achieved by the Netflix's cinematch algorithm is 0.9525 and that by BPC is 0.8567 [37]. While the RMSEs on different datasets are not comparable, it shows that in practice, the prediction with an accuracy of the RMSE less than 1 for ratings on a scale of $\{1, 5\}$ is already accurate enough to be used for recommendation tasks. Note that from Table II, it appears that Strategy 2 which partitions the range of the property evenly produced smaller RMSEs than Strategy 1 which set τ by certain percentiles of the data. However, the RMSEs by different strategies are not comparable, because the evaluations were performed on different rating data generated by different strategies. Nevertheless, Strategy 2 may create unbalanced portions of ratings. For example, we may have no path of rating 1 but a lot of paths of rating 2, which will never occur for Strategy 1. For this reason, Strategy 1 is used by default in the rest of the paper.

TABLE II
RMSE ON DIFFERENT DATASETS.

τ: Strategy 1	Harvard	Meridian	HP-S3	YouTube
RMF	0.934	0.831	0.675	0.923
MMMF	0.969	0.863	0.686	0.957
NMF	0.977	0.904	0.682	0.969
MF Ensembles	0.920	0.821	0.661	0.901
τ: Strategy 2	Harvard	Meridian	HP-S3	YouTube
RMF	0.920	0.776	0.669	0.910
MMMF	0.919	0.810	0.670	0.944
NMF	0.932	0.829	0.674	0.961
MF Ensembles	0.904	0.766	0.653	0.873

Overall, RMF is lightweight and suits well for online deployment in P2P applications, and is thus used in Section VI for the case study on overlay construction and routing. Table III shows the confusion matrices achieved by RMF on the four datasets. In these matrices, each column represents the predicted ratings, while each row represents the actual ratings. Thus, the diagonal entries represent the percentage of the correct prediction, and the off-diagonal entries represent the percentage of "confusions" or mis-ratings. For example, the entry at $(2, 2)$ represents the percentage of the rating-2 paths which are correctly predicted as rating-2, and the entry at $(2, 3)$ represents the percentage of the rating-2 paths which are wrongly predicted as rating-3, i.e. the confusions from rating-2 to rating-3. It can be seen that while there are mis-ratings, most of them have a small error of $|x_{ij} - \hat{x}_{ij}| = 1$, marked as shaded entries in the confusion matrices in Table III. This means that the mis-ratings are under control. For example, a rating-5 path may be wrongly predicted as 4, but seldom as 3, 2 or 1, since the entries at $(5, 3)$, $(5, 2)$ and $(5, 1)$ in all confusion matrices are small.

TABLE III
CONFUSION MATRICES.

Harvard		Predicted				
		1	2	3	4	5
Actual	1	**68%**	28%	2%	1%	1%
	2	18%	**60%**	20%	1%	1%
	3	3%	13%	**66%**	17%	1%
	4	3%	4%	16%	**67%**	10%
	5	3%	3%	5%	43%	**46%**

Meridian		Predicted				
		1	2	3	4	5
Actual	1	**78%**	18%	3%	1%	0%
	2	8%	**59%**	30%	3%	0%
	3	1%	18%	**60%**	20%	1%
	4	1%	3%	33%	**59%**	4%
	5	1%	1%	12%	59%	**27%**

HP-S3		Predicted				
		1	2	3	4	5
Actual	1	**92%**	6%	1%	1%	0%
	2	11%	**65%**	22%	2%	0%
	3	1%	20%	**68%**	11%	0%
	4	0%	3%	33%	**58%**	6%
	5	0%	1%	10%	55%	**34%**

YouTube		Predicted				
		1	2	3	4	5
Actual	1	**80%**	17%	1%	1%	1%
	2	13%	**66%**	20%	1%	0%
	3	2%	13%	**69%**	15%	0%
	4	3%	8%	33%	**52%**	4%
	5	2%	7%	12%	46%	**32%**

Note that we also evaluated another matrix completion method, namely LMaFit[4], which was used in [19] for traffic matrix completion and found that it performed much worse than the MFs used in this paper. For example, the RMSE by LMaFit on Meridian, HP-S3 and YouTube are 1.357, 1.139 and 1.422 respectively. Note also that many general matrix completion methods including LMaFit take as input an incomplete matrix and make updates on the entire matrix. This means that they cannot process the dynamic measurements in the Harvard dataset, nor can they be decentralized. It is however worth mentioning that LMaFit runs much faster than the MFs based on SGD.

3) Observations: Our experiments reveal a useful effect of ordinal rating. In practice, MF is known to be sensitive to outliers such as those unusually large and small values in the datasets. However, as ratings are quantized measures, measurement outliers are naturally handled by truncating the large and small property values to 1 and R. A direct consequence is that MF becomes insensitive to parameter settings, as the inputs are always in a relatively small and fixed range.

In the experiments, we observed that there exist nodes which have a poor rating with all their k neighbors. The likely reason is that those nodes have a poor access link, i.e. the link by which a node is connected to the Internet core is the bottleneck, which causes all connected paths to perform poorly. Thus, we calculate the mean rating and the standard deviation of the measured paths associated with each node, denoted by μ_i and σ_i. If σ_i is small, we do not consider that node i provides useful information about the performance of the network, and will simply predict the ratings of all paths of node i as μ_i. In our datasets, the non-informative nodes are rare (i.e. no more than 10 in each dataset), and thus the "mean rating" trick improved the accuracy only slightly. However, we incorporate it to detect non-informative nodes so that they won't pose a problem in any case.

[4]The source code is downloaded from http://lmafit.blogs.rice.edu/.

4) Discussions on Scalability: The MF models have proved to work well for recommender systems on extremely large matrices such as the one in the Netflix prize. Thus, there is no scalability issue when running MFs on performance matrices constructed on networks even with millions of nodes. However, two practical questions need to be answered when deploying MFs on real, large networks:

- How many measurements are required to make predictions with a decent accuracy, i.e., an RMSE at least smaller than 1?
- How fast do MFs run on such a large matrix?

Regarding the first question, a theory in matrix completion states that a $n \times n$ matrix of rank r can be exactly or accurately recovered, with high probability, from as few as $O(nr\log(n))$ randomly sampled entries [33]. This means that each node would have to probe $O(r\log(n))$ neighbors, which scales fairly well in theory. Nevertheless, we are interested in evaluating whether such a bound holds or is required for MF-based network inference on large networks. Regarding the second question, it is known that MFs based on SGD are computationally efficient, as SGD involves only vector operations [10], [12], and empirically converge fast for large matrices, as demonstrated in BPC. We leave the study of these issues as future work, because it would require really large-scale network measurement data.

V. Active and Progressive Inference

A. Active Inference

Despite the strong similarities, there is one important difference between network inference and recommender systems: unlike recommender systems where users rate items voluntarily, here we have more control over data acquisition because network nodes may actively select a few other nodes to probe. This insight enables to design an active sampling scheme that selectively measures a set of paths so that the inference of the other unmeasured paths is more accurate.

The key is to find which paths are more informative, which is difficult without additional knowledge such as the topology and routing of the network. However, we can exploit the uncertainty of a prediction, measured by the variance. Intuitively, if we are not certain about a prediction, measuring the path is probably the simplest and the best way to reduce this uncertainty. In our problem, MF ensembles allow us to compute the variance of each prediction, by computing the variance of the predictions by different MF predictors, as mentioned in Section IV-C4. Thus, we incorporate an uncertainty sampling strategy [44] that selects those paths of which our inference is the least certain, i.e. with large variance. To this end, we perform active inference in three steps.

1) Learn MF ensembles with a small number of measurements that are randomly sampled and compute the predictions and their variance.
2) Choose those predictions with large variance and actively measure the paths corresponding to them.
3) Learn another MF ensembles with all the available measurements including those randomly and actively sampled.

It is clear that, under a given measurement budget, i.e. a given number of paths to be measured, active inference has the doubled computational overheads over standard inference with random sampling due to the twice computations of MF ensembles. However, we will show in Section V-C that empirically the extra overheads lead to an improvement on prediction accuracy by $4\% - 8\%$ on our datasets.

B. Progressive Inference

Besides active inference, we also develop a progressive inference framework for ABW that incorporates the measurement principle of self-induced congestion.

1) ABW Rating by Self-Induced Congestion: As mentioned in Section III, popular ABW measurement tools such as pathload [14] and pathchirp [15] are based on the principle of self-induced congestion the utilization of which enables the direct rating without knowing the actual value. The idea is that, by sending a constant-rate flow, each probe performs naturally a binary test that checks whether the ABW is larger or smaller than the probe rate, illustrated in Figure 6. Thus, we can set the probe rate of each flow to one of the quantization thresholds in $\tau = \{\tau_1, \ldots, \tau_{R-1}\}$ and according to the responses, determine between which pairs of thresholds the ABW value is.

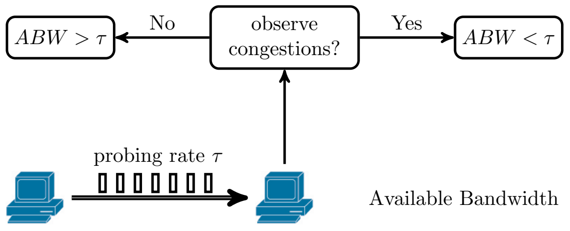


Fig. 6. A binary test based on self-induced congestion.

For each path, such rating procedure takes $O(\log R)$ probes in a binary search manner, which is fairly cheap. Moreover, when performing ABW rating on multiple paths of a network simultaneously, we can place the binary tests randomly over time and across paths, instead of carrying continuously a series of binary tests on the same paths until the exact rating of a path is reached. In doing so, we may further reduce the impact of the probe traffic on network usage and the collisions of the probe traffic on the common links of different paths. While interesting, we leave the measurement study of this ABW rating methodology as future work.

2) ABW Inference by Progressive Learning: The above rating procedure enables to design a progressive learning framework which can speed up the convergence of the inference. The key is to utilize the fact that each binary test in a rating measurement returns some partial information of the form of either $x_{ij} < c$ or $x_{ij} \geqslant c$ which is already usable for inference. For example, when we find that the rating of the path from node i to node j is no smaller than 3, i.e., $x_{ij} \geqslant 3$, although we don't know the exact rating, we can still constrain the inference system so that it tries to make a prediction of $\hat{x}_{ij}$ no smaller than 3. Thus, the constraint put on the inference system is that $\hat{x}_{ij} \geqslant 3$. If such a constraint cannot be satisfied, we penalize the violation of the constraint, which is done by modifying the loss function l in MF in eq. 3 to be asymmetric, i.e. we make an asymmetric loss function for a partial rating of the form of either $x_{ij} < c$ or $x_{ij} \geqslant c$, given by

$$l_p(x_{ij}, \hat{x}_{ij}) = \begin{cases} 0 & \text{if } \hat{x}_{ij} \text{ and } x_{ij} \text{ are consistent} \\ l(c, \hat{x}_{ij}) & \text{otherwise} \end{cases} \tag{13}$$

For the example above, our knowledge on x_{ij} is that $x_{ij} \geqslant 3$. Thus, if MF makes a prediction of $\hat{x}_{ij}$ no smaller than 3, the loss function returns 0, because the prediction and the partial rating are consistent. Otherwise, it returns the loss of $l(3, \hat{x}_{ij})$.

The progressive learning of MF hence solves the following minimization problem:

$$\min \sum_{ij \in \Omega_{\text{exact}}} l(x_{ij}, \hat{x}_{ij}) + \sum_{ij \in \Omega_{\text{partial}}} l_p(x_{ij}, \hat{x}_{ij}), \tag{14}$$

where Ω_{exact} is the set of exact rating measurements and Ω_{partial} is the set of partial rating measurements. Note that over time, as more probe flows are sent into the network, more and more partial measurements become exact and move from Ω_{partial} to Ω_{exact}. The optimization can be done by the same SGD algorithm mentioned in Section IV-D1.

The reason why progressive inference converges faster than standard inference with exact ratings only is that the inference can already start before a full rating measurement is completed. For example, node i firstly sends a constant-rate flow to node j to test whether the rating of the ABW is smaller than 3, and node j returns a "yes" or "no" response. At the next time, node i adjusts the probe rate so as to test whether the rating is larger than 4 or smaller than 2 according to the response of the previous probe. For progressive inference, node i can perform an update when receiving a response from node j, while for standard inference, node i has to wait until it knows exactly the value of the rating.

Progressive inference is particularly useful when dealing with dynamic measurements, because we can maintain the rating information by performing binary tests, each of which checking alternatively whether the rating has increased or decreased. Moreover, in our measurement studies on the PlanetLab network, we observed that some PlanetLab nodes were configured with constrained maximum packet transfer rates which were below the ABW of the paths connected to them. Consequently, we cannot measure the exact ABW ratings of certain paths, because it requires to send large traffic at a rate that exceeds the constraints. However, we still get partial information about those paths, i.e., we know that the ABW is larger than the constrained rate, which can be exploited by the progressive inference framework. Note that progressive inference is not useful for RTT, because RTT can be easily obtained using ping-type tools with fairly low costs. There is thus no partial rating information about RTT that could be used in a progressive manner.

C. Evaluations

The evaluations of both the active and the progressive inference framework were performed on the same datasets as described in Section IV-E.

1) Active Inference: We fixed the measurement budget by setting k, the number of neighbors to be probed at each node, as the following: $k = 20$ for Harvard and HP-S3, 64 for Meridian and 100 for YouTube. We then performed active inference as described earlier: first learn MF ensembles with half of the neighbors of each node randomly selected; then select actively the other half whose variances are the largest; learn other MF ensembles with all available measurements. We compared the RMSEs of active inference and of standard inference with random neighbor selection, i.e. each node randomly selects all its k neighbors, shown in Table IV. It can be seen that active inference achieved about $4\% - 8\%$ improvements over standard inference on different datasets. Note that while looking small, such improvements are non-trivial if compared with the requested improvement of 10% in the Netflix prize.

TABLE IV
RMSE OF ACTIVE AND STANDARD INFERENCE.

	Harvard	Meridian	HP-S3	YouTube
active inference	0.595	0.757	0.521	0.588
standard inference	0.652	0.790	0.566	0.637

2) Progressive Inference: We performed experiments to highlight the advantage of progressive inference using partial ratings over standard inference using exact ratings on the convergence speed. To this end, we ran RMF with progressive and standard inference respectively on the ABW dataset, HP-S3. Note that we simulated the ABW measurement using the pathload strategy whereby the rate of a probe flow is set to one of the quantization thresholds in $\tau = \{\tau_1, \ldots, \tau_{R-1}\}$ and is adjusted according to the feedback of the previous probe. Thus, each probe flow returns a partial rating of whether the rating is larger or smaller than a certain value. For progressive inference, we utilize a partial rating immediately after it is obtained, whereas for standard inference, we wait until the exact rating is known which requires $O(\log R)$ probes. Figure 7 plots the RMSE with respect to the number of probe flows sent from each node. It can be seen that the RMSEs of both progressive and standard inference decrease monotonically with the number of probe flows and the accuracy improvement becomes smaller and smaller. Clearly, progressive inference converges much faster than standard inference. Note also that progressive inference achieves the same accuracy as standard inference, because overall the inference in both cases is based on the same set of rating measurements.

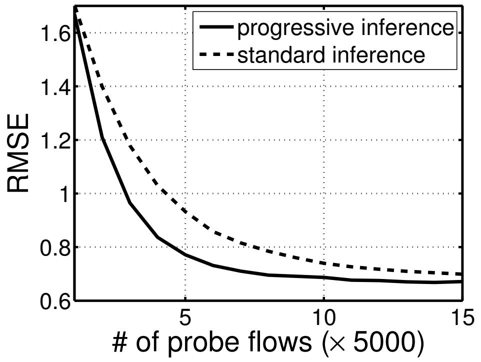


Fig. 7. Comparison on the HP-S3 dataset of progressive inference using partial ratings and standard inference using exact ratings on the convergence speed.

VI. CASE STUDY: LOCALITY-AWARE OVERLAY CONSTRUCTION AND ROUTING

With the techniques of rating-based network measurement and inference presented above, the remaining issue is the usability in Internet applications. There are two questions to be answered:

- While Section IV-E shows that the inference by various MF models can achieve at least an accuracy of RMSE less than 1, it is natural to ask whether such an accuracy is acceptable for applications.
- It is critical to choose a proper granularity for rating network paths, because although a finer granularity leads to more informative measurement, it also means more measurement overheads which may overweigh the benefit of exploiting the knowledge of network proximity. Thus, it is also natural to ask whether more fine-grained ratings always improve the performance of applications.

To answer these questions, we perform a case study on locality-aware overlay construction and routing and investigate whether locality-awareness can be achieved by using inferred ratings of path properties such as RTT and ABW. More specifically, we consider Pastry [16] and BitTorrent [17] which are typical structured and unstructured overlay networks and are known to enjoy the property of locality-awareness. Both Pastry and BitTorrent rely on an outside program that acquires network path properties. For example, Pastry uses measurement tools such as traceroute for hop count or ping for RTT [16], and BitTorrent uses Vivaldi to infer RTTs [43]. Here, we are interested in knowing whether our MF-based inference schemes can serve as the outside program in Pastry and BitTorrent[5] and the impact of the rating granularity on their performance. To simplify the evaluation, we only employ RMF in this section due to its accuracy and simplicity which facilitates its deployment in P2P applications.

A. Pastry

Pastry is a classic structured overlay network for the implementation of a DHT where the key-value pairs are stored

[5]Note that investigating how MF-based inference can actually be incorporated in the Pastry and BitTorrent protocols is beyond the scope of this paper. However, [10], [12] showed that MF can be implemented in a fully decentralized manner, with the same architecture as Vivaldi [5] which has been incorporated in BitTorrent [43]. This indicates that our MF-based inference schemes can be seamlessly used in BitTorrent, with no extra overhead required. We refer the interested readers to [10], [12] for the details of the decentralized architecture and implementations of our MF-based inference schemes.

in a P2P network in an organized manner so that queries can be achieved within $O(\log n)$ hops of routing, where n is the number of nodes. Here, we drop the description but refer the interested readers to [16] for the construction and routing algorithms of Pastry. We mention that Pastry determines the best routes by using the proximity knowledge.

In the simulations, we predicted ratings using RMF on the same datasets and with the same configuration as described in Section IV-E, except that we varied the rating levels from $R = 2$, $R = 2^2$,..., to $R = 2^8$, instead of $R = 5$ in the previous sections. For comparison, we also ran RMF to predict values of path properties, which is the most fine-grained measure. We then built the routing tables in Pastry using respectively no proximity knowledge, inferred ratings, inferred values, as well as the true measurements in the original datasets. We refer to Pastry using no proximity knowledge as $R = 0$ and using inferred values as $R = \infty$. Pastry using true measurements is the ideal case where the best routes can be found at the cost of $O(n^2)$ active measurements. In the implementation of Pastry, considering that there are at most 2500 nodes in our datasets, the node Id space is $N = 2^{14}$. Other parameters are: the base is $B = 2$, the leaf set size is $L = 2^B$, the neighbor set size is $M = 2^{B+1}$, and the routing table size is $T = (\log_L N, L - 1) = (7, 3)$.

After construction, we simulated $100,000$ lookup messages, i.e. queries, which were routed from randomly chosen nodes with randomly chosen keys. Then, we compared the routing performance of Pastry under $R = \{0, 2^1, \ldots, 2^8, \infty\}$ respectively with that of using true measurements. We measured the routing performance as the average of a routing metric, RTT for Harvard and Meridian and ABW for HP-S3, of each query and calculated the ratio between the different cases of R and the ideal case of using true measurements, called *stretch*. As Pastry can find the best routes when using true measurements, stretch for the routing performance is greater than 1 for RTT and smaller than 1 for ABW. The closer it is to 1, the better.

Figure 8 shows the stretches of both the routing performance and the hop count. We make the following observations. First, the stretch of the hop count is very close to 1, which is expected because Pastry always produces routes of $O(\log n)$ hops, with or without using the proximity knowledge. Second, also as expected, the routing performance of using inferred knowledge is worse than that of using true measurements, but is better than using no knowledge at all ($R = 0$), even for the case of $R = 2$ which leads to the most coarse-grained ratings. This shows that network inference of ratings is accurate enough to be used in Pastry. Third, increasing R after R reaches 2^3 or 2^4 cannot further improve the routing performance of Pastry. This shows that rating with finer granularities is not always beneficial for applications, let alone the extra costs due to more fine-grained measurements. Last, the routing performance of using inferred values ($R = \infty$) is a little worse than that of using inferred ratings with $R = 2^4$. This seems to suggest that the inference of property values was affected by measurement outliers, which further degraded the performance of Pastry. In contrast, outliers were naturally filtered out by rating-based measurements.

B. BitTorrent

BitTorrent is perhaps the most popular unstructured overlay network and is widely used for file sharing [17]. The protocol adopts a random peer selection procedure which causes large inter-domain traffic and degrades application performance. It is well known that these issues can be overcome by intelligent peer selection with the use of the proximity knowledge [3], [26], [27].

In the simulations, we inferred ratings and values of path properties using RMF as described in the above section. Then, we constructed overlay networks in BitTorrent with a standard scheme whereby a tracker node randomly proposes to each peer 50 peers in the network, among which 32 are selected. Ideally, we would like to connect each peer with those best performing peers, i.e. shortest for RTT and widest for ABW. However, such best peer selection strategy leads to the problem of reachability that some nodes are not reachable by some other nodes[6]. Our simulations showed that there could be $10\% \sim 15\%$ of the end-to-end obstructed paths. To overcome this problem, we came up with a mixed strategy whereby $p\%$ of the peers execute best peer selection and the other ones execute random peer selection, referred to as *best $p\%$ peer selection*[7]. We tested $p = \{100, 90, 50, 0\}$, with $p = 100$ being essentially the strategy of best peer selection and $p = 0$ being random peer selection. Note that the reachability problem is already avoided when $p = 90$.

We measured the performance of peer selection by the average performance of the links between each pair of peers selected by the peer selection procedure, shown in Figure 9. It can be seen that the performance of peer selection improves as more nodes execute the strategy of best peer selection. This shows that we can reduce download time or avoid congestions in BitTorrent by exploiting inferred ratings of RTT or ABW. It can also be seen that, while $R = 2^4$ or 2^5 appear to be optimal, which is similar to Pastry, the granularity seems to have less impact on BitTorrent. This is probably due to the protocol that requires each peer to select 32 peers out of 50 candidates, thus leaving us with limited choices for peer selection. Note that BitTorrent is particularly concerned with the performance of peer selection, because chunks of files are frequently exchanged between direct neighbors.

C. Remarks

Our case study is encouraging, showing that inferred ratings are accurate enough to be beneficial to applications such as Pastry and BitTorrent. Empirically, a coarse granularity of $R = 2^4$ appears to be satisfactory for our datasets, which is acceptable for RTT whose measurement is cheap. However, for

[6]In an overlay network, the links between nodes are directed. For example, there may be a link from A to B, because A chooses B as its neighbor. However, there may not be a link from B to A, because B may not choose A as its neighbor. Thus, the problem of reachability doesn't mean that the overlay network is disconnected. Instead, it means that some end-to-end paths are obstructed and that packets can only be routed in one direction between some end nodes.

[7]An alternative strategy for best $p\%$ peer selection is to let each node select $p\%$ best peers and $(100 - p)\%$ random peers. We tested both strategies and found that they achieved statistically similar results.

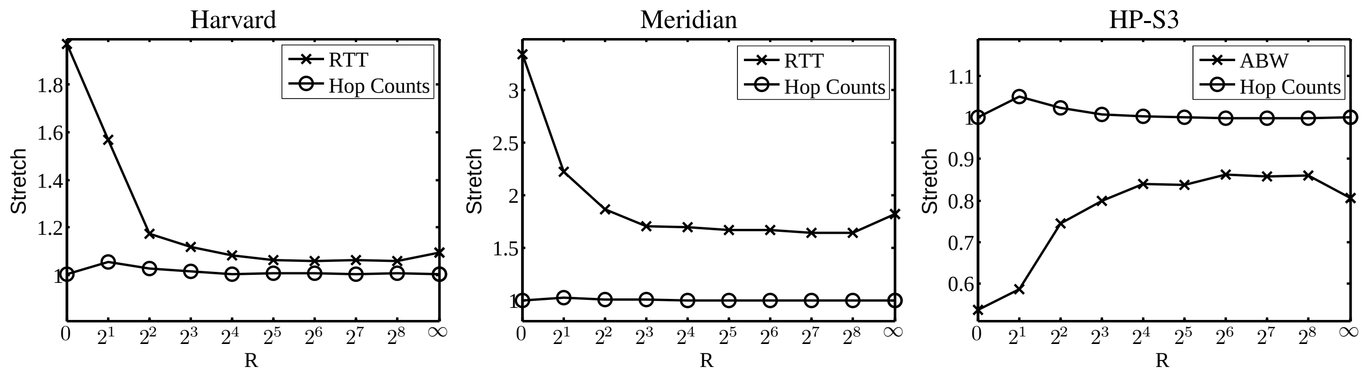


Fig. 8. Stretch of the routing performance of Pastry, defined as the ratio between the routing metric of Pastry using inferred ratings and of Pastry using true measurements. Note that $R = 0$ means that no proximity knowledge is used, and $R = \infty$ means that inferred values are used.

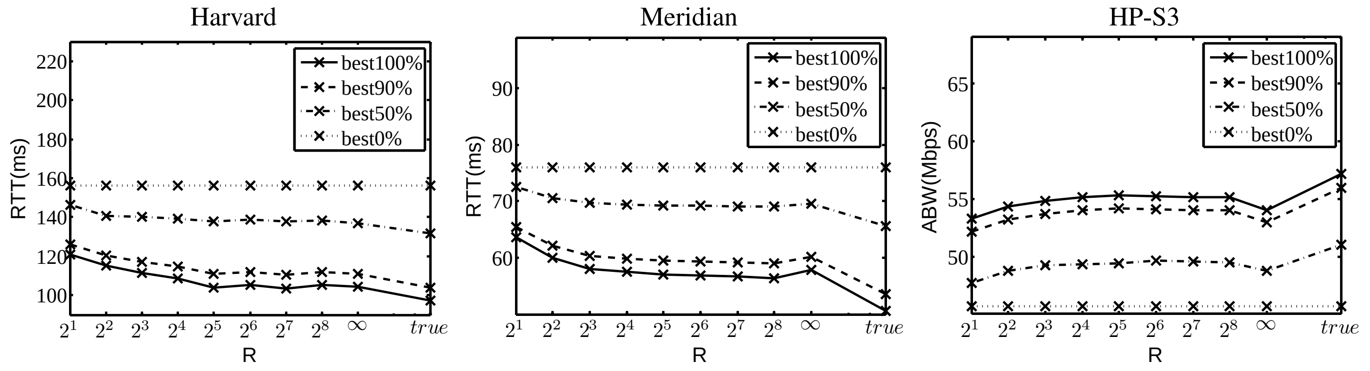


Fig. 9. Performance of peer selection for BitTorrent, calculated as the average link performance between each pair of selected peers. Note that "*true*" means that true measurements are used.

ABW, the improvement by increasing R from 2^3 to 2^4 comes at the cost of more probe flows and thus more overheads, which may not be considered worthy. In practice, the choice of R has to take into account not only the performance of applications but also the measurement budget.

We measured the inference accuracy by RMSE which reflects the overall accuracy of the prediction for each path. Such metric is more meaningful for applications where the inference accuracy of every path is equally important. For the task of intelligent peer selection where it is only important to ensure that the selected paths have a high rating, it may not be worth pushing the metric of RMSE to the limit. This insight shows that a prediction system should be evaluated from the applications' point of view, which highlights the importance of carrying out case studies on real applications.

Note that the case study focused on the impact of the accuracy of rating inference and the impact of the granularity in rating-based network measurement. Thus, we performed no comparison with other inference approaches, as none can deal with ratings of different path properties. For example, Vivaldi [5] infers RTT values by Euclidean embedding and was already shown to be less accurate than our MF-based approach when using the same amount of RTT measurements [12]. Tomography-based approaches [6]–[8] do not work on non-additive properties such as ABW and require the routing information of the network which is not available in our scenarios. In contrast, our MF-based approach is flexible in dealing with both additive and non-additive properties and in dealing with both property values and ratings of different levels, which is a unique feature that distinguishes it from all previous approaches.

VII. Conclusions and Future Work

This paper presents a novel concept of rating network paths, instead of measuring values of path properties, and investigates the inference of ratings by solutions designed for recommender systems, particularly by a class of matrix factorization techniques, which were found to work well. The case study on locality-aware overlay construction and routing highlights the usability of rating-based network measurement and inference by Internet applications. These studies reveal the advantages of ratings: they are informative, have low measurement cost and are easy to process in applications.

There are several avenues for future work. First, while we focused, in the case study, on the impact of the inference accuracy and the rating granularity, we are interested in the design of new protocols that can incorporate schemes of network inference of ratings. Second, we wish to apply our techniques on other inference problems such as traffic matrix estimation [21] and investigate the benefits of coarse-grained packet counting for the application of traffic engineering. In addition, we are also interested in a CDN case study, whereby users are redirected to nearby servers with good network performance. A particular benefit of rating-based user redirection is that load balancing can be naturally achieved by random server selection among servers with top-rating network connections. This can avoid the overuse of a particular server that has the best connection to a lot of users. This paper has already evaluated the accuracy of rating prediction on a CDN (YouTube) dataset, shown in Table III, which highlights the potentials of the techniques for CDNs. We wish to further justify the benefit on load balancing by investigating the server selection models in various CDNs.

References

[1] M. Crovella and B. Krishnamurthy, *Internet Measurement: Infrastructure, Traffic and Applications*. New York, NY, USA: John Wiley & Sons, Inc., 2006.

[2] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A survey and comparison of peer-to-peer overlay network schemes," *IEEE Communications Surveys and Tutorials*, vol. 7, pp. 72–93, 2005.

[3] E. Marocco, A. Fusco, I. Rimac, and V. Gurbani, "Improving peer selection in peer-to-peer applications: myths vs. reality," Internet Research Task Force Working Group, Dec. 2012, http://tools.ietf.org/html/rfc6821.

[4] T. S. E. Ng and H. Zhang, "Predicting Internet network distance with coordinates-based approaches," in *Proc. of IEEE INFOCOM*, 2002, pp. 170–179.

[5] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," in *Proc. of ACM SIGCOMM*, Portland, OR, USA, Aug. 2004, pp. 15–26.

[6] Y. Chen, D. Bindel, H. Song, and R. H. Katz, "An algebraic approach to practical and scalable overlay network monitoring," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 55–66, Aug. 2004.

[7] D. B. Chua, E. D. Kolaczyk, and M. Crovella, "Network kriging," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2263–2272, Dec. 2006.

[8] H. H. Song, L. Qiu, and Y. Zhang, "NetQuest: A flexible framework for large-scale network measurement," in *Proc. of ACM SIGMETRICS*, 2006.

[9] Y. Mao, L. Saul, and J. M. Smith, "IDES: An Internet distance estimation service for large networks," *IEEE Journal On Selected Areas in Communications*, vol. 24, no. 12, pp. 2273–2284, Dec. 2006.

[10] Y. Liao, W. Du, P. Geurts, and G. Leduc, "Decentralized prediction of end-to-end network performance classes," in *Proc. of CoNEXT*, Tokyo, Japan, Dec. 2011.

[11] Y. Liao, P. Geurts, and G. Leduc, "Network distance prediction based on decentralized matrix factorization," in *Proc. of IFIP Networking Conference*, Chennai, India, May 2010, pp. 15–26.

[12] Y. Liao, W. Du, P. Geurts, and G. Leduc, "DMFSGD: A decentralized matrix factorization algorithm for network distance prediction," *IEEE/ACM Transactions on Networking*, vol. 21, no. 5, pp. 1511–1524, Oct. 2013.

[13] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[14] M. Jain and C. Dovrolis, "End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 537–549, Aug. 2003.

[15] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, "pathchirp: Efficient available bandwidth estimation for network paths," in *Proc. of the Passive and Active Measurement*, 2003.

[16] A. Rowstron and P. Drusche, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *Proc. IFIP/ACM ICDSP*, 2001.

[17] Vuze Bittorrent, http://www.vuze.com/.

[18] L. Tang and M. Crovella, "Virtual landmarks for the Internet," in *Proc. of ACM/SIGCOMM Internet Measurement Conference*, Miami, FL, USA, Oct. 2003, pp. 143–152.

[19] G. Gürsun, N. Ruchansky, E. Terzi, and M. Crovella, "Inferring visibility: Who's (Not) talking to whom?" in *Proc. of ACM SIGCOMM*, 2012, pp. 151–162.

[20] G. Gürsun and M. Crovella, "On traffic matrix completion in the internet," in *Proc. of ACM/SIGCOMM Internet Measurement Conference*, 2012, pp. 399–412.

[21] M. Roughan, Y. Zhang, W. Willinger, and L. Qiu, "Spatio-temporal compressive sensing and internet traffic matrices," *IEEE/ACM Transactions on Networking*, vol. 20, no. 3, pp. 662–676, Jun. 2012.

[22] P. Maymounkov and D. Mazières, "Kademlia: A peer-to-peer information system based on the xor metric," in *International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002, pp. 53–65.

[23] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proc. of ACM SIGCOMM*, 2001, pp. 149–160.

[24] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: a distributed anonymous information storage and retrieval system," in *International workshop on Designing privacy enhancing technologies: design issues in anonymity and unobservability*, 2001, pp. 46–66.

[25] Gnutella, http://www.gnu.org/philosophy/gnutella.html/.

[26] D. R. Choffnes and F. E. Bustamante, "Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems," in *Proc. of ACM SIGCOMM*, 2008, pp. 363–374.

[27] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz, "P4p: provider portal for applications," in *Proc. of ACM SIGCOMM*, 2008, pp. 351–362.

[28] A. Shriram, M. Murray, Y. Hyun, N. Brownlee, A. Broido, M. Fomenkov, and K. Claffy, "Comparison of public end-to-end bandwidth estimation tools on high-speed links," in *Proc. of the Passive and Active Measurement*, 2005, pp. 306–320.

[29] M. Jain and C. Dovrolis, "Ten fallacies and pitfalls on end-to-end available bandwidth estimation," in *Proc. of ACM/SIGCOMM Internet Measurement Conference*, 2004.

[30] J. Sommers, P. Barford, and W. Willinger, "Laboratory-based calibration of available bandwidth estimation tools," *Microprocess. Microsyst.*, vol. 31, no. 4, pp. 222–235, Jun. 2007.

[31] P. Yalagandula, P. Sharma, S. Banerjee, S. Basu, and S.-J. Lee, "S3: a scalable sensing service for monitoring large networked systems," in *Proc. of the SIGCOMM workshop on Internet network management*, 2006, pp. 71–76.

[32] Google TV, http://www.google.com/tv/.

[33] E. J. Candès and Y. Plan, "Matrix completion with noise," *Proc. of the IEEE*, vol. 98, no. 6, pp. 925–936, 2010.

[34] V. K. Adhikari, S. Jain, Y. Chen, and Z.-L. Zhang, "Vivisecting youtube: An active measurement study," in *INFOCOM*, 2012.

[35] B. Wong, A. Slivkins, and E. Sirer, "Meridian: A lightweight network location service without virtual coordinates," in *Proc. of ACM SIGCOMM*, Philadelphia, Pennsylvania, USA, Aug. 2005, pp. 85–96.

[36] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Scalable collaborative filtering approaches for large recommender systems," *Journal of Machine Learning Research*, vol. 10, pp. 623–656, Jun. 2009.

[37] Netflix Prize, http://www.netflixprize.com/.

[38] J. D. M. Rennie and N. Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *International Conference on Machine Learning*, 2005, pp. 713–719.

[39] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2001, pp. 556–562.

[40] T. G. Dietterich, "Ensemble learning," in *The Handbook of Brain Theory and Neural Networks*. The MIT Press, 2002.

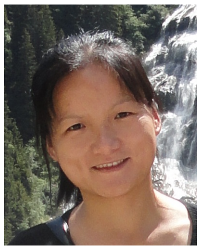
[41] M. Wu, "Collaborative filtering via ensembles of matrix factorizations," in *KDD Cup and Workshop at the 13th ACM SIGKDD Conference*, 2007.

[42] J. Surowiecki, *The Wisdom of Crowds*. Anchor, 2005.

[43] J. Ledlie, P. Gardner, and M. I. Seltzer, "Network coordinates in the wild," in *Proc. of USENIX Symposium on Networked Systems Design and Implementation*, Cambridge, Apr. 2007, pp. 22–35.

[44] B. Settles, "Active Learning Literature Survey," University of Wisconsin–Madison, Tech. Rep. 1648.

Wei Du is a postdoctoral researcher with the Research Unit in Networking (RUN), University of Liège, Belgium. He received his B.S. degree from Tianjin University, China, in 1997, and PhD degree from the Institute of Computing Technology, Chinese Academy of Sciences, China, in 2002. Since graduation, he has worked as postdoctoral researcher at INRIA Rocquencourt, France, Hamburg University, Germany, University of Liège, Belgium, University of Innsbruck, Austria, and University of Göttingen, Germany. His main research interests are computer networking and machine learning.



Yongjun Liao is a postdoctoral researcher with the Research Unit in Networking (RUN), University of Liège, Belgium. She received her B.S. and M.S. degrees from Guangxi University, China, in 1999 and 2002, and her PhD degree in Engineering Science from the University of Liège, Belgium, in 2013. Before joining University of Liège in 2007, she had worked as a software engineer in Hopen Software Engineering Co., Ltd. in Beijing, China. Her main research interests are the applications of machine learning techniques to computer networking problems such as the scalable monitoring of large networks.

Narisu Tao is a PhD candidate with the networking group, University of Göttingen, Germany. He received his B.S. and M.E. from Nanjing University, China, in 2007 and 2010. Since April 2012 he has been working as a research assistant at the networking group in University of Göttingen, Germany. His main research interests are computer networking and machine learning.

Pierre Geurts is an assistant professor with the EECS department, University of Liège, Belgium. He graduated as an electrical (computer science) engineer in 1998 and received the PhD degree in applied sciences in 2002. From 2006 to 2011, he was research associate of the FNRS (Belgium). His research interests concern the design of, computationally and statistically efficient, supervised and semi-supervised learning algorithms in order to exploit structured input and output spaces (sequences, images, time-series, graphs), with applications in bioinformatics, computer vision, and computer networks.

Dr. Xiaoming Fu (M'02 - SM'09) received his bachelor degree in automatic instrumentation in 1994 and master degree in computer science in 1997, respectively, from Northeastern University, China, and his Ph.D. degree in computer science from Tsinghua University, Beijing, China in 2000. He was a research staff at the Technical University Berlin until joining the University of Göttingen, Germany in 2002, where he has been a professor in computer science and heading the Computer Networks Group since 2007.

Dr. Fu's research interests include network architectures, protocols, and applications. He is currently an editorial board member of IEEE Communications Magazine, IEEE Transactions on Network and Service Management, Elsevier Computer Networks, and Computer Communications, and has published over 100 papers in journals and international conference proceedings. He has served as secretary (2008-2010) and vice chair (2010-2012) of the IEEE Communications Society Technical Committee on Computer Communications (TCCC), and chair (2011-2013) of the Internet Technical Committee (ITC) of the IEEE Communications Society and the Internet Society. He has been involved in FP6 ENABLE, VIDIOS, Daidalos-II and MING-T projects and is the coordinator of FP7 GreenICN and MobileCloud projects.

Dr. Fu is the recipient of the IEEE LANMAN 2013 Best Paper Award and the 2005 University of Göttingen Foundation Award for Exceptional Publications by Young Scholars.

Guy Leduc (S'81 - M'83 - SM'13) is a full professor with the EECS department, University of Liège, Belgium, and is since 1997 the head of the Research Unit in Networking (RUN). He received the MSc degree in electrical engineering and the PhD degree in engineering science, with professorial accreditation, from the University of Liège, in 1983 and 1991 respectively. His main research interests are the application of machine learning techniques to networking problems, traffic engineering, resilience and new network architectures. His research unit is or has been involved in European projects such as mPlane on measurements, ECODE on cognitive networking, ResumeNet on resilient networking, ANA on autonomic networking, TOTEM on an open-source toolbox for traffic engineering, and the E-NEXT European network of excellence. He is a former chairman of the IFIP Technical Committee (TC6) on Communications Systems, a former steering committee member of the IFIP Networking Conference, and an area editor of IEEE Transactions on Network and Service Management and Elsevier Computer Communications journals. He received the IFIP Networking 2006 and 2010 Best Paper Awards.