Available online at www.sciencedirect.com
ScienceDirect
Performance Evaluation 64 (2007) 838–855
PERFORMANCE EVALUATION
An International Journal
www.elsevier.com/locate/peva

Random graphs as models of hierarchical peer-to-peer networks☆

R. Gaeta, M. Sereno*

Dipartimento di Informatica, Università di Torino, Torino, Italy

Available online 29 June 2007

Abstract

This paper proposes the development and application of random graphs-based performance evaluation techniques to understand design trade-offs for hierarchical unstructured peer-to-peer networks. In particular, the connections between lower and higher level peers (that are known as *leaves* and *ultra-peers* in the Gnutella jargon) are modeled as a *bipartite random graph* while the overlay network used by ultra-peers to forward queries is modeled as a *generalized random graph*. Both the random graph models consider peers of either level as partitioned into classes; this feature is included in the model description to consider the mismatch between the logical topology of the application and the physical deployment of peers throughout the Internet. To assign realistic values to the input model parameters and to validate the model predictions we obtained snapshots of the Gnutella application topology at both levels and conducted simulation experiments on these snapshots. The paper highlights a few exploitations of the modeling technique with a particular focus on the evaluation of the impact of locality awareness on user and network performance measures.

Keywords: Hierarchical peer-to-peer networks; Bipartite random graphs; Generalized random graphs; Topology mismatch

1. Introduction

Peer-to-peer (P2P) paradigm has recently emerged as a new model for distributed networked services and applications. P2P applications have been deployed in many different areas, such as distributed grid computing, storage, web cache, Internet telephony, streaming, conferencing, content distribution, and so on. But file sharing applications are perhaps the most popular P2P applications: many different file sharing systems, such as Gnutella, Kazaa, Edonkey, Emule, BitTorrent, exist and collect millions of users. These kinds of applications are characterizing a great fraction of the Internet traffic nowadays and several statistics on IP traffic have recently provided evidence that P2P traffic is starting to dominate the bandwidth in certain segments of the Internet [1].

In a P2P-based application participants are termed as *peers* and play the dual role of both provider and requester of a service. Services are the location and transfer of (part of) a *resource* that can be owned by several peers thus defining the resource *popularity*. Peers organize in an overlay (logical) network on top of the physical network. Each peer

☆ This work has been partially supported by the Italian Ministry for University and Research (MIUR) within the frameworks of the FAMOUS (PRIN) and of the PROFILES (PRIN) projects.

* Corresponding author. Tel.: +39 011 6706718; fax: +39 011 751603.
E-mail address: matteo.sereno@di.unito.it (M. Sereno).

0166-5316/$ - see front matter
doi:10.1016/j.peva.2007.06.020

establishes application level connections only to a subset of known peers (its *neighbors*). Management of the overlay network is done at the application level: different management schemes define different classes of P2P networks.

In this paper we consider searching in *hierarchical unstructured* P2P networks. In a hierarchical unstructured P2P network a two-tier overlay structure divides peers into two groups: lower and higher level peers. Examples of P2P application adopting this architectural paradigm are for instance Gnutella [2], FastTrack [3] (Kazaa [4]), DirectConnect [5] and for non-file sharing P2P applications Skype [6].

Borrowing from the Gnutella protocol [2] in the rest of the paper we adopt the terms *ultra-peer* and *leaf* to denote a high level and a low level peer, respectively. The ultra-peers form a *top-level* overlay while the leaf peers are connected to the top-level overlay through one or multiple connections. The ultra-peer two-tier structure has been found effective to improve the scalability of the system. A leaf keeps only a small number of connections open, and that is to the ultra-peers. An ultra-peer acts as a proxy to the Gnutella network for the leaves connected to it. This has the effect of making the Gnutella network scalable by reducing the number of peers on the network involved in message handling and routing, as well as by reducing the actual traffic among them. Although the two-tier architecture helped in reducing the total number of messages sent following a request, a problem remains open: in general, the logical topologies created by P2P applications do not match well with the underlying Internet topology [7], leading to ineffective use of the physical networking infrastructure.

In this paper we develop a mathematical model to analyze the performance of search in hierarchical P2P networks. We exploit the model to gain insight into the effect that locality-aware topology creation and maintenance algorithms have on the availability of a resource that is inserted into the network for the first time by one peer connected from a particular region of the Internet.

We consider the problem of analyzing searching techniques in hierarchical unstructured P2P networks very important for two reasons:

(1) despite the growing popularity of BitTorrent-like P2P protocols unstructured P2P networks remain extremely popular these days as witnessed by recent measurements reporting that the number of different participants is over 2,000,000 users [8]. Therefore we believe that applicability of the results we obtain is not limited.
(2) although most of the traffic generated by P2P applications is for data transfer searching still represents a non-negligible communication and processing burden on peers. This is particularly true in a hierarchical network application where ultra-peers manage all the requests issued by leaves.

In the model we develop the connections between ultra-peers and leaves are represented as a *bipartite random graph* while the overlay network used by ultra-peers to forward queries is modeled as a *generalized random graph* [9]. This choice is motivated by the observation that at any point in time a snapshot of the overlay network can be represented by a finite graph of size N where a node represents a peer and application-level connections among peers are modeled as edges. Although P2P networks are highly dynamic and result in a constantly and randomly changing topology, if we assume that the time scale of search operations is much shorter than the time scale of the P2P network topology evolution, we can reasonably assume that at any instant in time the snapshot of the P2P network topology can be viewed as an instance of a finite graph of size N. Both the random graph models consider nodes of either level as partitioned into classes; this feature is included in the model description to consider the mismatch between the logical topology of the application and the physical deployment of peers throughout the Internet. Furthermore, we assume that a resource is characterized by a popularity that we define as the probability that a randomly chosen peer owns a copy of the resource.

Thanks to the model we developed we also analyze the impact of locality awareness in algorithms to create and maintain the overlay topology of a hierarchical P2P application. In locality-aware algorithms an effort is made to create logical connections among peers that minimize the cost of message transmission over the physical network. Locality awareness is thus an important issue at the system level to optimize the performance of the application but we show that pushing the exploitation of locality to the extreme could have unexpectedly negative consequences on the possibility of a resource to spread fast and widely enough to become accessible to most users with high probability.

The model analysis is carried out by deriving the generating function of the distribution of the number of query messages sent throughout the network starting from a query originator that belongs to a given class. The generating function is successively refined to account for the probability for a leaf in class i of successfully finding a resource in class j. The solution is developed with the assumption that some characteristics of the two-tier overlay structure are described by independent random variables. We test the validity of the independence assumption in Section 5 and in

Section 6.2 where the model predictions are thoroughly validated against simulations conducted on real snapshots of the overlay networks of Gnutella [8].

The main results of this paper are:

- the development of an analytical model of a complex P2P network application by means of the combination of two random graph-based models. Furthermore, this is the first paper where bipartite random graphs are extended with the partition of nodes into classes.
- the exploitation of the model results in providing insights into the topology mismatch problem in hierarchical P2P networks.

To the best of our knowledge this is the first paper that represents a complex hierarchical network application with two complex random graphs and the first paper to consider nodes of a bipartite random graph as partitioned into classes.

The paper is organized as follows: Section 2 summarizes work that is somehow related to our contribution, Section 3 describes the features of the Gnutella 0.6 protocol that are most relevant to our work, Section 4 contains the technical part of the paper where the random graph models are derived and performance indexes are defined. Section 5 contains a discussion on the validity of the main modeling assumptions we made, Section 6 discusses results obtained from the model solution: the validity of the proposed model is discussed by extensively comparing its predictions against simulation on real overlay networks topologies in Section 6.2, while results showing the impact of locality awareness in topology creation algorithms on the hit probability as well as on the number of messages are presented in Section 6.3. Section 7 summarizes the paper content and outlines future developments.

2. Related works

In this section we briefly illustrate previous works on random graphs-based modeling of P2P network as well as papers that deal with the mismatch between logical (overlay) and physical topologies in P2P networks. Several papers have analyzed the behavior of P2P-based applications by means of random graphs although *none* of them focused on hierarchical P2P networks.

The work in [10] introduces a number of local search strategies based on random walks that utilize high degree nodes in power-law graphs and that have costs scaling sub-linearly with the size of the graph. The authors use generalized random graphs and the generating function analysis technique to demonstrate the utility of these strategies on flat P2P networks. The paper differs from our since the goal is to devise efficient search strategies in P2P networks with a particular degree distribution.

An evolving random graph-based model for studying the evolution of flat P2P communities is defined in [11]; this model more accurately represents the process of network creation but can only be used for analyzing basic properties such as reachability from a given node in the network. To conduct more sophisticated analysis (average queue length at the nodes, average delay for responses) the same model must be used in conjunction with a discrete event simulation approach. This paper differs from our's since locality-aware network creation is not considered and because quantitative analysis is conducted by simulation.

The work in [12] exploits generalized random graphs to represent overlay networks but only focuses on simple homogeneous scenarios where peers are always available, resources are uniformly distributed among peers, and flooding-based search strategies are probabilistic but uniform across peers. Furthermore, validation of the model predictions is only done by simulations on synthetic instances of random graphs.

The authors of [13] introduce a scalable searching protocol for locating contents in random networks with heavy-tailed degree distributions. The algorithm is capable of finding any content in the network with probability one in time $O(\log N)$, with a total traffic that provably scales sub-linearly with the network size, N. The analysis of the size of the giant connected component of a random graph with heavy-tailed degree distributions under bond percolation is at the heart of their main results.

The work in [14] exploits the theory of random graphs to prove the properties of a generalization of the search that combines flooding and random walks. The authors discuss the properties of normalized flooding on classes of random graphs and evaluate the more general hybrid strategy by simulation on several types of topologies. In this paper random graphs are used to prove theorems on the topology properties of special classes of networks.

As for the mismatch between logical (overlay) and physical topologies in P2P networks, the paper [7] pointed out that application-level topologies generated by P2P applications tend to poorly match the underlying Internet

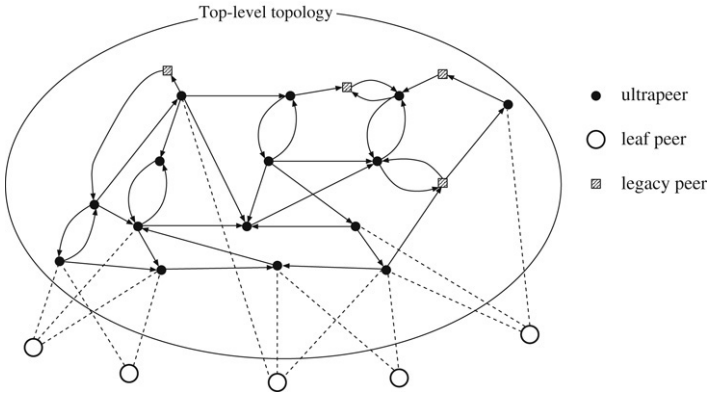

Fig. 1. Gnutella overlay topology.

topology leading to ineffective use of the network resources. This effect together with the increasing popularity of P2P applications (see for instance [15]) imply several consequences both on network operators and on final users (P2P and non-P2P users). The paper [16] discusses several different strategies/policies that network operators can apply to manage P2P traffic. An important class of the strategies presented in [16] concerns policies that allow the construction of network-aware overlay topologies.

Proposals on network-aware overlay building strategy appeared in [17] and in [18] where the authors proposed mechanisms for building the overlay topology of unstructured P2P applications that allow to mitigate the mismatch between overlay and network topologies.

3. System description

In a hierarchical unstructured P2P network a two-tier overlay structure divides peers into two groups: lower and higher level peers. Examples of P2P applications adopting this architectural paradigm are Gnutella, FastTrack/Kazaa, DirectConnect and for non-file sharing P2P applications Skype. We choose Gnutella as our reference application due to its widespread use although the approach can be easily customized to describe other hierarchical applications.

In this section we summarize the features of the Gnutella 0.6 protocol that are most relevant to our work. We refer the reader to [20] for a comprehensive description of the protocol. Gnutella uses a two-tier overlay structure that divides peers into two groups: *ultra-peers* and *leaf* peers [19,20]. Fig. 1 depicts the overlay structure of the Gnutella network: the ultra-peers form a *top-level* overlay while the leaf peers are connected to the top-level overlay through one or multiple connections. The peers that do not implement the ultra-peer features (i.e., they use a old version of the Gnutella protocol) are called *legacy* peers. The legacy peers can only reside in the top-level and do not accept any connection to the leaves.[1] The ultra-peer two-tier structure has been found effective to improve the scalability of the system. A leaf keeps only a small number of connections open, and that is to ultra-peers. An ultra-peer acts as a proxy to the Gnutella network for the leaves connected to it. This has the effect of making the Gnutella network scalable by reducing the number of peers on the network involved in message handling and routing, as well as by reducing the actual traffic among them.

When a leaf issues a request for a resource it sends query messages to the ultra-peers connected to it. These ultra-peers forward the query only to their leaves which are likely to have a copy of the resource (this technique is called Query Routing Protocol). Ultra-peers also forward the query to their neighbors in the top-level overlay; the process of forwarding queries to all neighbors is called *flooding*. To prevent the top-level overlay to be overloaded with messages, queries have a Time to Live (TTL) attribute that sets the maximum number of hops that can be taken from the ultra-peers that first received the query from a leaf. Following different paths in the overlay network, multiple copies of a query issued by a leaf for a resource could be received by an ultra-peer; in this case, an ultra-peer processes the query

[1] Note that we depict the overlay network topology as a *directed* graph because, although the top-level peers are connected among them by using TCP connections, we discovered in the snapshots we obtained from the authors of [8] and in snapshots we obtained from a crawler we developed that the neighboring relations are not always symmetric. We will briefly discuss the impact of this issue on our modeling framework.

once and discards successive copies. Furthermore, ultra-peers select a subset of their neighbors to forward a query; if the search returns a too small number of hits other neighbors are selected to receive the query. This technique is called Dynamic Querying and helps to reduce the amount of traffic caused by searches of common resources.

In Section 4 we develop a model to represent the two-tier Gnutella overlay topology. We model the algorithm for locating a resource that we described but we do not model the Dynamic Querying feature and make the simplifying assumption that ultra-peers forward a query to *all* their neighbors in the top-level. This allows us to simplify the model derivation and provides upper bounds on the probability of locating a resource and the number of messages sent during the search. Nevertheless, extending our model to include the Dynamic Querying algorithm is simple and we will briefly discuss this issue in Section 7.

4. The model

In this section we derive the model to represent hierarchical P2P networks as described in Section 3. Section 4.1 defines the input model parameters that describe the two-tier topology of the hierarchical P2P application as well as the distribution of a target resource throughout the application network. Section 4.2 derives the generating function for the random variable that represents the number of ultra-peers that can be reached by a query forwarded by a randomly chosen ultra-peer. Section 4.3 exploits the reachability results to obtain the hit probabilities for leaves in a given class and Section 4.4 terminates Section 4 by providing the characterization of the random variable describing the overall number of messages sent during a search as well as the number of messages exchanged between peers belonging to the same class.

In the rest of the section discrete random variables will be denoted in upper-case, probabilities will be denoted in lower-case while generating functions of probability distributions will be denoted in calligraphic style. For the sake of brevity and clarity in the rest of the paper we will use the notation "the g.f. for X" as a shorthand for the formal and correct statement "the generating function of the probability distribution for discrete random variable X".

We assume the reader is familiar with the properties of generating functions for probability distributions of random variables and with their application to the analysis of random graphs. We refer to [9] for a comprehensive survey on this topic. Furthermore, to help the reader we summarized the notation we define to specify our modeling framework in Table 1.

4.1. Model parameters

We consider a set of $\{1 \dots n_c\}$ classes and we assume that each leaf and each ultra-peer belong to a given class $c \in \{1 \dots n_c\}$. The input parameters are the probability distributions that define the discrete random variables representing the application network topology and the distribution of a resource among peers. To this end, we assume that a resource is spread throughout the leaves following a class-dependent probability. In particular, we define ρ_c as the probability that a randomly chosen peer (either leaf or ultra-peer) in class c has a copy of the resource.

We represent the connections between leaves and ultra-peers as a random bipartite graph whose definition requires the specification of n_c^2 (one for each pair of classes) discrete random variables and their probability distributions. In particular we define:

- $U_{i,j}$ the discrete random variable that describes the number of ultra-peers in class j that are connected to a randomly chosen leaf in class i. The probability distribution of $U_{i,j}$ is denoted as $\{u_{i,j,k}\},\ 1 \le i, j \le n_c,\ k \ge 0$ where $u_{i,j,k}$ denotes the probability that $U_{i,j} = k$. The g.f. for $U_{i,j}$ is defined as $\mathcal{U}_{i,j}(z) = \sum_{k=0}^{\infty} u_{i,j,k} \cdot z^k$.
- $L_{i,j}$ the discrete random variable that describes the number of leaves in class j that are connected to a randomly chosen ultra-peer in class i. The probability distribution of $L_{i,j}$ is denoted as $\{l_{i,j,k}\},\ 1 \le i, j \le n_c,\ k \ge 0$ where $l_{i,j,k}$ denotes the probability that $L_{i,j} = k$. The g.f. for $L_{i,j}$ is defined as $\mathcal{L}_{i,j}(z) = \sum_{k=0}^{\infty} l_{i,j,k} \cdot z^k$.

Ultra-peers in the application network form an overlay by establishing connections to other ultra-peers to forward queries on the behalf of their shielded leaves. We represent the connections between ultra-peers in the overlay network as a random directed graph. To describe the main assumption we made and to illustrate the consequences on the rest of the model derivation we consider the case where ultra-peers are not partitioned into classes.

Let O and I be the random variables representing the number of incoming and outgoing edges of a randomly chosen ultra-peer, respectively, and let $\{p_{k_i,k_o}\}$ be the joint probability distribution describing the number of incoming

Table 1
Model notation

Notation	Description
n_c	Number of classes
$U_{i,j}$	Number of ultra-peers in class j connected to a leaf in class i
$L_{i,j}$	Number of leaves in class j connected to an ultra-peer in class i
$L_{k,i}^{\text{excl}}$	Number of other leaves in class i connected to an ultra-peer in class k
$O_{i,j}$	Number of outgoing edges from an ultra-peer in class i to ultra-peers in class j
$R_{i,j,d}$	Number of ultra-peers in class j reachable in d hops from an ultra-peer in class i
$NL_{i,j,d}$	Number of leaves d hops away in class j that can be queried by a leaf in class i
$NU_{i,j,d}$	Number of ultra-peers d hops away in class j that can be queried by a leaf in class i
$HL_{i,j,d}$	Number of leaves (with resource) d hops away in class j that can be queried by a leaf in class i
$HU_{i,j,d}$	Number of ultra-peers (with resource) d hops away in class j that can be queried by a leaf in class i
$M_{k,k'}^{U\to U}$	Number of messages forwarded by an ultra-peer in class k to ultra-peers in class k'
$M_{k,j}^{U\to L}$	Number of messages forwarded by an ultra-peer in class k to its leaves in class j
$M_{j,k}^{L\to U}$	Number of messages sent by a leaf in class j to ultra-peers in class k
$TM_{i,d}^{U\to U}$	Number of ultra-peer to ultra-peer messages sent by ultra-peers d hops away from a leaf in class i
$T_i^{U\to U}$	Total number of ultra-peers to ultra-peers messages for a leaf in class i
$TM_{i,d}^{U\to L}$	Number of ultra-peers to leaves messages sent by ultra-peers d hops away from a leaf in class i
$T_i^{U\to L}$	Total number of ultra-peers to leaves messages for a leaf in class i
$T_i^{L\to U}$	Total number of leaves to ultra-peers messages for a leaf in class i
$ITM_{i,d}^{U\to U}$	Number of ultra-peer to ultra-peer messages inside class i sent by ultra-peers d hops away from a leaf in class i
$IT_i^{U\to U}$	Total number of ultra-peers to ultra-peers messages inside class i for a leaf in class i
$ITM_{i,d}^{U\to L}$	Number of ultra-peers to leaves messages inside class i sent by ultra-peers d hops away from a leaf in class i
$IT_i^{U\to L}$	Total number of ultra-peers to leaves messages inside class i for a leaf in class i
$IT_i^{L\to U}$	Total number of leaves to ultra-peers messages inside class i for a leaf in class i

(i.e., k_i) and outgoing (i.e., k_o) edges of a randomly chosen ultra-peer and let $\mathcal{G}(x, y) = \sum_{k_i,k_o} p_{k_i,k_o} x^{k_i} y^{k_o}$ be the generating function for the joint probability distribution. We assume that the two marginal distributions $\{p_{k_i}\}$ where $p_{k_i} = \sum_{k_o=0}^{\infty} p_{k_i,k_o}$ and $\{p_{k_o}\}$ where $p_{k_o} = \sum_{k_i=0}^{\infty} p_{k_i,k_o}$ are independent (the validity of this assumption is discussed in Section 5). It follows that we assume that $\mathcal{G}(x, y) = \mathcal{G}_i(x)\mathcal{G}_o(y)$ where $\mathcal{G}_o(y) = \sum_{k_o=0}^{\infty} p_{k_o} y^{k_o}$ and $\mathcal{G}_i(x) = \sum_{k_i=0}^{\infty} p_{k_i} x^{k_i}$.

An immediate consequence of the independence assumption is that if we are also interested in the probability distribution of the number of outgoing edges of the ultra-peer reached by following a randomly chosen edge from the general expression we obtain

$$\frac{\partial_x \mathcal{G}(x, y)|_{x=1}}{\hat{k}} = \frac{\partial_x \mathcal{G}_i(x)\mathcal{G}_o(y)|_{x=1}}{\hat{k}} = \mathcal{G}_o(y) \cdot \frac{\partial_x \mathcal{G}_i(x)|_{x=1}}{\hat{k}} = \mathcal{G}_o(y),$$

since $\partial_x \mathcal{G}_i(x)|_{x=1} = \hat{k}$. It follows that the independence assumption we made allows us to express the g.f. for the neighborhood d hops away from a randomly chosen ultra-peer as $\mathcal{G}_o^{(d)}(y)$ where $\mathcal{G}_o^{(d)}(y)$ is a shorthand to denote the composition of $\mathcal{G}_o(y)$ with itself d times.

Back to our class partition, we define:

- $O_{i,j}$ the discrete random variable that describes the number of outgoing edges that emanate from an ultra-peer in class i to ultra-peers in class j. The probability distribution of $O_{i,j}$ is denoted as $\{o_{i,j,k}\},\ 1 \le i, j \le n_c,\ k \ge 0$ where $o_{i,j,k}$ denotes the probability that $O_{i,j} = k$. The g.f. for $O_{i,j}$ is defined as $\mathcal{O}_{i,j}(z) = \sum_{k=0}^{\infty} o_{i,j,k} \cdot z^k$.

4.2. Modeling reachability in the overlay network

For the rest of the model development we need to characterize the random variable $R_{i,j,d}$ that represents the number of ultra-peers in class j that can be reached in d hops from a randomly chosen ultra-peer in class i. From the implications of the independence assumptions we discussed it turns out that the g.f. for $R_{i,j,d}$ is given by

$$\mathcal{R}_{i,j,d}(z) = \begin{cases} \mathcal{O}_{i,j}(z) & \text{if } d = 1 \\ \prod_{m=1}^{n_c} \mathcal{O}_{i,m}(\mathcal{O}_{m,j}(z)) & \text{if } d = 2 \\ \prod_{m=1}^{n_c} \prod_{m'=1}^{n_c} \mathcal{O}_{i,m}(\mathcal{O}_{m,m'}(\mathcal{O}_{m',j}(z))) & \text{if } d = 3. \end{cases} \tag{1}$$

The expression can be generalized for arbitrary values of d and has been derived by observing that an ultra-peer in class j can be reached in two hops from an ultra-peer in class i by first visiting an ultra-peer in class m and then reaching the ultra-peer in class j. This path can be taken using all possible intermediate classes m; if we assume that for any i and j random variables $O_{i,j}$ are mutually independent then we can exploit the well-known results for the generating functions of the sum of independent random variables that is given by the product of the generating functions of the summed random variables.

4.3. Modeling the hit probabilities

In this section we derive the hit probabilities among leaves for a query issued by a leaf in class i (Section 4.3.1) as well as the hit probability among ultra-peers for the same query (Section 4.3.2).

4.3.1. Hit probability among leaves

We consider a randomly chosen leaf in class i. We aim to characterize the discrete random variable $NL_{i,j,d}$ that represents the number of leaves d hops away in class j that can be queried to know whether they hold a copy of the requested resource. All the leaves that can be probed because they share the same ultra-peers of the randomly chosen one are assumed to be 0 hops away from it, those connected to ultra-peers that can be reached if the query forwarded for 1 hop are assumed to be 1 hop away and so on.

To derive the g.f. of $NL_{i,j,0}$ we observe that the number of leaves of class j connected to an ultra-peer in class k is described by $L_{k,j}$ and that the number of ultra-peers in class k to which a leaf of class i is connected is represented by $U_{i,k}$. Moreover, we need to characterize the discrete random variable $L_{k,i}^{\text{excl}}$ that represents the number of other leaves in class i connected to an ultra-peer in class k, i.e., *excluding* the randomly chosen leaf in class i. The g.f. for $L_{k,i}^{\text{excl}}$ is given by $\mathcal{L}_{k,i}^{\text{excl}}(z) = \frac{1}{\bar{l}_{k,i}} \frac{\mathrm{d}\mathcal{L}_{k,i}(z)}{\mathrm{d}z}$ where $\bar{l}_{k,i} = \frac{\mathrm{d}\mathcal{L}_{k,i}(z)}{\mathrm{d}z}|_{z=1}$ (see [9] for details). We can now derive the g.f. for $NL_{i,j,0}$ as

$$\mathcal{NL}_{i,j,0}(z) = \begin{cases} \prod_{k=1}^{n_c} \mathcal{U}_{i,k}(\mathcal{L}_{k,j}(z)) & \text{if } i \neq j \\ \prod_{k=1}^{n_c} \mathcal{U}_{i,k}(\mathcal{L}_{k,j}^{\text{excl}}(z)) & \text{if } i = j. \end{cases} \tag{2}$$

The g.f. for $NL_{i,j,d}$ can be obtained by properly composing the generating functions for the neighborhood in the overlay network with generating functions for the number of ultra-peers connected to a leaf in class i. In particular we obtain $\mathcal{NL}_{i,j,d}(z) = \prod_{k=1}^{n_c} \prod_{k'=1}^{n_c} \mathcal{U}_{i,k'}(\mathcal{R}_{k',k,d}(\mathcal{L}_{k,j}(z)))$ for $d > 0$.

To obtain the expression for the hit probabilities we introduce random variables $HL_{i,j,d}$ to describe the number of leaves in class j that are d hops away from a randomly chosen leaf in class i and that store a copy of the requested resource (we denote these leaves as *hits*). The hits are a subset of the leaves described by $NL_{i,j,d}$ that we obtain by

modeling their number by a binomial distribution with parameter ρ_j. It turns out that the g.f. for $HL_{i,j,d}$ is equal to $\mathcal{HL}_{i,j,d}(z) = \mathcal{NL}_{i,j,d}(1 + (z-1)\rho_j)$ (see [9] for details).

We can define the first performance index related to the probability of finding at least one leaf that stores a copy of the requested resource. For a leaf in class i we define

$$phl_{i,j,d} = 1 - \mathcal{HL}_{i,j,d}(0) \tag{3}$$

as the probability of finding at least one hit among leaves in class j that are d hops away.

Exploiting the independence assumption we can derive the g.f. for the random variable describing the number of hits regardless of the distance from the leaf in class i originating the query as $\mathcal{HL}_{i,j}(z) = \prod_{d=0}^{\mathrm{TTL}} \mathcal{HL}_{i,j,d}(z)$ where the time to live (TTL) is the maximum number of hops that a query can take when forwarded by ultra-peers in the overlay network. We then define

$$phl_{i,j} = 1 - \mathcal{HL}_{i,j}(0) \tag{4}$$

as the probability of finding at least one hit among leaves in class j.

As a last step we derive the g.f. for the random variable describing the number of hits regardless of the class information that $\mathcal{HL}_i(z) = \prod_{j=1}^{n_c} \mathcal{HL}_{i,j}(z)$ from which it follows that the probability of finding at least one hit among leaves for a query issued by a leaf in class i is defined as

$$phl_i = 1 - \mathcal{HL}_i(0). \tag{5}$$

4.3.2. Hit probability among ultra-peers

We consider a randomly chosen leaf in class i. We characterize the discrete random variable $NU_{i,j,d}$ that represents the number of ultra-peers d hops away in class j whose state can be probed by a randomly chosen leaf in class i to select those that store a copy of the requested resource. To derive the g.f. of $NU_{i,j,0}$ we observe that the number of ultra-peers in class j to which a leaf of class i is connected is represented by $U_{i,k}$. Therefore, we readily obtain $\mathcal{NU}_{i,j,0}(z) = \mathcal{U}_{i,k}(z)$ while in the case of $d > 0$ we get $\mathcal{NU}_{i,j,d}(z) = \prod_{k=1}^{n_c} \mathcal{U}_{i,k}(\mathcal{R}_{k,j,d}(z))$.

We introduce the random variable $HU_{i,j,d}$ to describe the number of ultra-peers in class j that are d hops away from a randomly chosen leaf in class i and that store a copy of the requested resource. These hits are a subset of the ultra-peers described by $NU_{i,j,d}$ that we obtain by modeling their number by a binomial distribution with parameter ρ_j. It turns out that the g.f. for $HU_{i,j,d}$ is equal to $\mathcal{HU}_{i,j,d}(z) = \mathcal{NU}_{i,j,d}(1 + (z-1)\rho_j)$ (see [9] for details) and the probability of finding at least one hit among ultra-peers in class j that are d hops away from a leaf in class i is given by

$$phu_{i,j,d} = 1 - \mathcal{HU}_{i,j,d}(0). \tag{6}$$

The g.f. for the random variable describing the number of hits regardless of the distance from the leaf in class i originating the query is given by $\mathcal{HU}_{i,j}(z) = \prod_{d=0}^{\mathrm{TTL}} \mathcal{HU}_{i,j,d}(z)$, while the g.f. for the random variable describing the number of hits regardless of the class information is obtained as $\mathcal{HU}_i(z) = \prod_{j=1}^{n_c} \mathcal{HU}_{i,j}(z)$.

Similarly, we define the probability of finding at least one hit among ultra-peers in class j as

$$phu_{i,j} = 1 - \mathcal{HU}_{i,j}(0) \tag{7}$$

and the probability of finding at least one hit among ultra-peers as

$$phu_i = 1 - \mathcal{HU}_i(0). \tag{8}$$

4.4. Modeling the query traffic

The equations we derived can be further exploited to characterize the number of messages sent during the entire query process triggered by a randomly chosen leaf in class i. In this section we first obtain expressions for the total number of messages (Section 4.4.1) and then we derive expressions for the number of messages exchanged within the same class of the leaf that issued the query (Section 4.4.2).

4.4.1. Total number of messages

The total number of messages is obtained by summing the contribution of the messages sent from ultra-peers to ultra-peers, from ultra-peers to leaves, and from leaves to ultra-peers. These three contributing terms require the definition of three random variables:

(1) we define the random variable $M_{k,k'}^{U\to U}$ to represent the number of messages forwarded by an ultra-peer in class k to ultra-peers in class k' when handling an incoming query. We assume that ultra-peers employ a simple flooding algorithm where an incoming query is forwarded to all neighbors in the overlay network excluding the ultra-peer from which the incoming query was received. In this case the g.f. for $M_{k,k'}^{U\to U}$ is given by $\mathcal{M}_{k,k'}^{U\to U}(z) = \mathcal{O}_{k,k'}(z)$;
(2) in order to save bandwidth and to reduce the overhead associated with query handling, ultra-peers employ an algorithm where an incoming query is forwarded only to the leaves they are connected to and that store a copy of the resource (the schema is called Query Routing Protocol [19]). This feature can be included in our model description by introducing the random variable $M_{k,j}^{U\to L}$ to represent the number of messages forwarded by an ultra-peer in class k to its leaves in class j that store a copy of the requested resource. The g.f. for $M_{k,j}^{U\to L}$ is simply given by $\mathcal{M}_{k,j}^{U\to L}(z) = \mathcal{L}_{k,j}(1 + (z-1)\rho_j)$;
(3) the number of messages sent by a leaf to the ultra-peers to which it is connected is described by the random variable $M_{j,k}^{L\to U}$ whose g.f. is simply $\mathcal{M}_{j,k}^{L\to U}(z) = \mathcal{U}_{j,k}(z)$;

Starting from these basic random variables we can derive the g.f. for the random variable $TM_{i,d}^{U\to U}$ that describes the number of ultra-peer to ultra-peer messages sent by ultra-peers d hops away from a leaf in class i as

$$\mathcal{TM}_{i,d}^{U\to U}(z) = \begin{cases} \prod_{k=1}^{n_c}\prod_{k'=1}^{n_c} \mathcal{U}_{i,k}(\mathcal{M}_{k,k'}^{U\to U}(z)) & \text{if } d = 0 \\ \prod_{k=1}^{n_c}\prod_{k'=1}^{n_c}\prod_{k''=1}^{n_c} \mathcal{U}_{i,k}(\mathcal{R}_{k,k',d}(\mathcal{M}_{k',k''}^{U\to U}(z))) & \text{if } d > 0. \end{cases} \tag{9}$$

Therefore, the g.f. of the random variable for the total number of ultra-peer to ultra-peer messages $T_i^{U\to U}$ is given by $\mathcal{T}_i^{U\to U}(z) = \prod_{d=0}^{TTL-1} \mathcal{TM}_{i,d}^{U\to U}(z)$.

Similarly, the g.f. for the random variable $TM_{i,d}^{U\to L}$ that describes the number of ultra-peer to leaves messages sent by ultra-peers d hops away from a leaf in class i as

$$\mathcal{TM}_{i,d}^{U\to L}(z) = \begin{cases} \prod_{k=1}^{n_c}\prod_{j=1}^{n_c} \mathcal{U}_{i,k}(\mathcal{M}_{k,j}^{U\to L}(z)) & \text{if } d = 0 \\ \prod_{k=1}^{n_c}\prod_{k'=1}^{n_c}\prod_{j=1}^{n_c} \mathcal{U}_{i,k}(\mathcal{R}_{k,k',d}(\mathcal{M}_{k',j}^{U\to L}(z))) & \text{if } d > 0. \end{cases} \tag{10}$$

Also in this case, the g.f. of the random variable for the total number of ultra-peer to leaves messages $T_i^{U\to L}$ is given by $\mathcal{T}_i^{U\to L}(z) = \prod_{d=0}^{\mathrm{TTL}} \mathcal{TM}_{i,d}^{U\to L}(z)$. Finally, the g.f. of the random variable for the total number of leaves to ultra-peers messages $T_i^{L\to U}$ is given by

$$\mathcal{T}_i^{L\to U}(z) = \prod_{k=1}^{n_c} \mathcal{U}_{i,k}(z). \tag{11}$$

It is now possible to define the random variable T_i that represents the total number of messages sent following a query issued by a randomly chosen leaf in class i. Since $T_i = T_i^{U\to L} + T_i^{U\to U} + T_i^{L\to U}$ by the independence assumption we obtain its g.f. as $\mathcal{T}_i(z) = (\mathcal{T}_i^{U\to L}(z)) \cdot (\mathcal{T}_i^{U\to U}(z)) \cdot (\mathcal{T}_i^{L\to U}(z))$ from which we define the average number of total messages sent to locate a resource for a leaf in class i as

$$\overline{m}_i = \frac{\mathrm{d}\mathcal{T}_i(z)}{\mathrm{d}z}|_{z=1}. \tag{12}$$

4.4.2. Total number of messages within the same class

A similar derivation can be carried out to characterize the random variable describing the number of messages sent within class i when a randomly chosen leaf in class i starts a search process.

Following our derivation in Section 4.4.1, we can define the random variables $ITM_{i,d}^{U\to U}$, $ITM_{i,d}^{U\to L}$, $IT_i^{U\to U}$, $IT_i^{U\to L}$, and $IT_i^{L\to U}$ that consider only messages whose source and destination are both in class i.

We derive the g.f. of $ITM_{i,d}^{U\to U}$ as

$$\mathcal{ITM}_{i,d}^{U\to U}(z) = \begin{cases} \mathcal{U}_{i,i}(\mathcal{M}_{i,i}^{U\to U}(z)) & \text{if } d = 0 \\ \prod_{k=1}^{n_c} \mathcal{U}_{i,k}(\mathcal{R}_{k,i,d}(\mathcal{M}_{i,i}^{U\to U}(z))) & \text{if } d > 0 \end{cases}$$

and we obtain the g.f. of $IT_i^{U\to U}$ as $\mathcal{IT}_i^{U\to U}(z) = \prod_{d=0}^{TTL-1} \mathcal{ITM}_{i,d}^{U\to U}(z)$.

To account for the messages sent from ultra-peers to leaves we consider the g.f. of $ITM_{i,d}^{U\to L}$ as

$$\mathcal{ITM}_{i,d}^{U\to L}(z) = \begin{cases} \mathcal{U}_{i,i}(\mathcal{M}_{i,i}^{U\to L}(z)) & \text{if } d = 0 \\ \prod_{k=1}^{n_c} \mathcal{U}_{i,k}(\mathcal{R}_{k,i,d}(\mathcal{M}_{i,i}^{U\to L}(z))) & \text{if } d > 0 \end{cases}$$

and we obtain the g.f. of $IT_i^{U\to L}$ as $\mathcal{IT}_i^{U\to L}(z) = \prod_{d=0}^{\mathrm{TTL}} \mathcal{ITM}_{i,d}^{U\to L}(z)$.

Finally, the g.f. of the random variable for the total number of leaves to ultra-peers messages $IT_i^{L\to U}$ is given by $\mathcal{IT}_i^{L\to U}(z) = \mathcal{U}_{i,i}(z)$.

The random variable IT_i that represents the total number of messages sent following a query issued by a randomly chosen leaf in class i that are exchanged between peers (either leaves or ultra-peers) that are in class i is defined as $IT_i = IT_i^{U\to L} + IT_i^{U\to U} + IT_i^{L\to U}$ and by the independence assumption we obtain its g.f. as $\mathcal{IT}_i(z) = (\mathcal{IT}_i^{U\to L}(z)) \cdot (\mathcal{IT}_i^{U\to U}(z)) \cdot (\mathcal{IT}_i^{L\to U}(z))$ from which we define the average number of total messages sent between peers that are both in class i as

$$\overline{im_i} = \frac{\mathrm{d}\mathcal{IT}_i(z)}{\mathrm{d}z}|_{z=1}. \tag{13}$$

5. Discussion on the modeling assumptions

The model we developed is based on several assumptions:

- *Directedness of the graph representing the top-level overlay network*. To check how reasonable is the assumption that the edges connecting peers in the overlay network form a directed graph we computed the *reciprocity* [21] of a graph defined as $r = \frac{L^{\leftrightarrow}}{L}$ where $L^{\leftrightarrow}$ denotes the number of edges pointing in both directions and L denotes the total number of links. The value of r represents the average probability that an edge has a reciprocal in the opposite direction. We computed r for the snapshots we used to validate our model: the values ranged from 0.908 to 0.928 showing that the top-level overlay network can be represented as a directed graph. We also developed an undirected version of our model which is more complex but that yields results that are very close to the simpler directed graph model we presented in Section 4.
- *Independence of the incoming and outgoing degree distributions*. To test for the independence of the degree distributions we estimated from the snapshots the joint degree distribution $\{p_{k_i,k_o}\}$ and computed the marginal distributions $\{p_{k_i}\}$ and $\{p_{k_o}\}$. We computed the standard Pearson coefficient of correlation as $p = \frac{\sum_{k_i,k_o} p_{k_i,k_o} - p(k_i)\cdot p(k_o)}{\sigma_i \cdot \sigma_o}$. Values of p close to 0 indicate a lack of linear dependence between the two random variables described by distributions $\{p_{k_i}\}$ and $\{p_{k_o}\}$. We computed p and observed that this value ranged from 0.072 to 0.165 showing the absence of strong linear dependence.
- *Independence of the outgoing degree and the number of leaves distributions*. We estimated from the snapshots the joint degree distribution $\{p_{l_i,k_o}\}$ that describes the number of leaves connected to an ultra-peer and the number of connections to other ultra-peers. We computed the standard Pearson coefficient p and observed that this value ranged from 0.017 to 0.043 showing also in this case the absence of strong linear dependence.

- *Average cardinality of intersection of leaves sets*. In deriving several generating functions, e.g., Eq. (2), we made the implicit assumption that the intersection of the sets of leaves managed by ultra-peers connected to a randomly chosen leaf is the empty set. Therefore, given a leaf k connected to n ultra-peers whose set of leaves we denote as LS_i $(i = 1 \dots n)$ we computed the overlap index for leaf k as

$$oi(k) = \frac{\left| \bigcup_{\substack{i,j=1 \\ i \neq j}}^{n} (LS_i \bigcap LS_j) \right| - 1}{\left| \bigcup_{i=1}^{n} LS_i \right| - 1}$$

  we then computed the average overlap index $\overline{oi}$ by averaging the values of $oi(k)$ over all leaves. We observed that $\overline{oi}$ ranged from 0.002 to 0.004 proving that the assumption we made is verified by the snapshots we used to validate our model.

6. Results

In this section we describe the simulation methodology we used to validate our model and we present validation results for several system scenarios. We also exploit the model to assess the impact of locality awareness of topology creation algorithms on the user perceived hit probability and on the number of messages at the network level. To this end we classify peers depending on the Internet Autonomous System (AS) they belong with the simplifying assumption that the cost of a message between two peers in the same AS is negligible with respect to the cost of a message between two peers in different ASs. Other classifications are possible: in particular, a classification based on the Internet Service Provider (ISP) of peers would be more realistic with respect to the cost of messages between peers. We chose the AS-based classification due to the availability of software mapping IP addresses to AS numbers.

6.1. Simulation methodology

We thoroughly evaluated the accuracy of Eqs. (4), (5), (7), (8), (12) and (13) by comparing their values with the results of simulations on real overlay topologies obtained by crawling the Gnutella file sharing application. We obtained snapshots of the Gnutella topology from the authors of [8] who developed a distributed crawler to gather information on the ultra-peers and leaves connected to each contacted ultra-peer. We also developed a similar crawler to obtain more snapshots. In all these snapshots, the identity of peers is expressed as a pair of IP address and port number used by application participants. We enriched this description by computing the Autonomous System Number (ASN) associated with each IP address of the snapshots. To this end, we used the ASN extension for the Webalizer software [22].

The simulator we developed employs standard statistical procedures to estimate 95% confidence intervals for all the performance indexes defined in Section 4. We consider $N_{\text{snapshots}}$ overlay topologies (in our experiments $N_{\text{snapshots}} = 20$): each topology is used to obtain one realization of each of the performance metrics. The hth realization is obtained in the following way:

- in the initialization phase each of the N_l leaves and N_u ultra-peers is assigned the resource with probability ρ_c if it belongs to class c.
- each leaf issues a request for the resource by sending a query to each of the ultra-peers it is connected. Each ultra-peer is then the root of a breadth-first visit whose depth is equal to TTL. During the visit an ultra-peer forwards the query to its neighbors in the overlay and to its leaves that store a copy of the object. Ultra-peers discard query already received for the same object issued by the same leaf and in this case do not forward the query to their neighbors.
- for each query the total number of messages as well as the number of messages exchanged between peers in the same class are recorded and estimated, i.e., Eqs. (12) and (13). Furthermore, the average fraction of leaves in class i that find at least one copy of the resource among leaves and ultra-peers in class j is recorded to obtain estimates of phl_i, phu_i, $phl_{i,j}$, and $phu_{i,j}$, i.e., Eqs. (4), (5), (7) and (8).

6.2. Model validation

In this section we provide validation results for the model we developed. We fixed TTL = 2 for all the experiments and obtained the input model parameter that define the application topology, i.e., $\{u_{i,j,k}\}$, $\{l_{i,j,k}\}$, and $\{o_{i,j,k}\}$, by estimating each distribution from each snapshot and then computing the average distribution. The value of TTL we use is realistic since several Gnutella implementations set the initial value for query messages well below the maximum suggested value of 7. This is because the number of connections maintained by one ultra-peer to others can be as high as 30 therefore, low values for TTL help to avoid overloading the top-level overlay network.

We computed the empirical distribution $\{u_{i,j,k}\}$ by averaging the distribution of the single snapshots, i.e., $u_{i,j,k} = \frac{1}{N_{\text{snapshots}}} \sum_{h=1}^{N_{\text{snapshots}}} u_{i,j,k}(h)$ where $u_{i,j,k}(h)$ is the fraction of leaves in class i that have k connections to ultra-peers in class j in the hth snapshot. We then used the empirical distributions as input to the model we develop and we computed the values of Eqs. (4), (5), (7), (8), (12) and (13).

Each scenario we consider is further characterized by the class partition, i.e., the identity of ASs we consider, and the distribution of the requested resource among peers, i.e., the set of ρ_c probabilities one for each AS we consider in the partition. Exploring all the parameter space would be unfeasible therefore we adopted the following approach: starting from the snapshot enriched with the ASN attribute for IP addresses we ranked each AS by the number of leaves contained in each snapshot. We obtained the top five ASs (in decimal notation these are AS 1668, 19548, 20115, 22773, and 7132) and considered all partitions that can be obtained with five classes. We then randomly assigned a ρ_c probability to each AS we consider in the partition ranging from 10^{-2} to 10^{-7}.

Tables 2–6 summarize the validation results we present. The tables refer to one case of 5 classes partition, two cases of 3 classes partition, and two cases of a 1 class partition. In each table the AS called INTERNET refers to all the leaves that do not belong to any of the class used to partition the set of peers. The tables report the value of each of the performance indexes we defined in Section 4 as well as the confidence interval provided by simulation. As it can be easily noted the model predictions are in excellent agreement with predictions obtained by simulations: in a very large fraction of cases the model provides predictions that fall within the confidence intervals. In the minority of cases where this does not happen the model provides slight over-estimations. Due to the lack of space we selected a subset of the results we obtained; all the model results we omit are in the same kind of excellent agreement with the simulation outcomes.

We can safely conclude that our model is accurate in predicting the behavior of the performance indexes we defined in a broad range of different system scenario. Furthermore, the model required a few seconds of CPU time to be numerically solved by Maple while simulations required several hours to terminate.

6.3. Model exploitation

In this section we use the model we developed to investigate the effect of locality awareness in algorithms to create and maintain the overlay topology of a hierarchical P2P application. In locality-aware algorithms an effort is made to create logical connections among peers that minimize the cost of message transmission over the physical network by trying to connect pair of application participants that are “close” to each other, e.g., they are part of the same ISP or the same AS.

Locality awareness is thus an important issue at the system level to optimize performance of the application that results in faster response to user requests for both locating and transferring a resource. Exploiting locality is also beneficial for ISPs: P2P friendly solutions are currently being sought, and many of them have already been proposed which try to minimize the transit costs incurred by providers by exploiting traffic locality. In this case, locality awareness in the creation and maintenance of the application network topology leads to economical benefits for ISPs that provide access to customers who are mostly interested in using P2P-based applications [16].

Nevertheless, pushing the exploitation of locality to the extreme could have unexpected negative consequences on the possibility of a resource to spread fast and widely enough to become accessible to most users with high probability. To investigate this issue we use our model to analyze the scenario where a resource is initially made available by one user that is connected to the Internet with an IP address belonging to a given AS (the seed AS). In this case, all peers in ASs other than the seed that issue requests for locating this resource will only hit in the seed AS. This scenario represents the case of a newly created resource that is made available for the very first time by one peer in the system.

Table 2
Validation results where the IP addresses of peers are partitioned into 5 classes represented by AS numbers

phl_{AS_i,AS_j}	AS 1668	AS 19548	AS 20115	AS 22773	AS 7132	INTERNET
	0.936	0.335	0.037	0.006	0.000	0.0014
AS 1668	[0.921, 0.940]	[0.305, 0.404]	[0.028, 0.080]	[0.002, 0.021]	[0.000, 0.000]	[0.001, 0.030]
	0.957	0.381	0.044	0.007	0.001	0.0017
AS 19548	[0.930, 0.962]	[0.335, 0.466]	[0.032, 0.091]	[0.004, 0.027]	[0.000, 0.000]	[0.001, 0.035]
	0.953	0.372	0.043	0.007	0.001	0.0016
AS 20115	[0.928, 0.961]	[0.325, 0.450]	[0.032, 0.090]	[0.003, 0.024]	[0.000, 0.000]	[0.001, 0.036]
	0.946	0.356	0.040	0.006	0.001	0.0015
AS 22773	[0.925, 0.948]	[0.321, 0.425]	[0.031, 0.080]	[0.003, 0.025]	[0.000, 0.000]	[0.001, 0.032]
	0.956	0.378	0.043	0.007	0.001	0.0016
AS 7132	[0.932, 0.959]	[0.331, 0.466]	[0.032, 0.090]	[0.003, 0.027]	[0.000, 0.000]	[0.001, 0.033]
	0.949	0.364	0.042	0.007	0.001	0.0016
INTERNET	[0.921, 0.943]	[0.316, 0.428]	[0.029, 0.086]	[0.003, 0.023]	[0.000, 0.000]	[0.00!, 0.033]
ρ	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
	677.8	800.8	785.8	735.7	794.5	779.8
$\overline{m}_i$	[622.8, 748.7]	[742.0, 838.8]	[723.0, 824.3]	[659.4, 801.3]	[727.5, 844.7]	[694.1, 823.5]
	0.0	1.4	1.2	0.7	3.8	549.5
$\overline{im}_i$	[0.0, 0.0]	[1.1, 1.9]	[1.1, 1.5]	[0.3, 1.8]	[2.4, 5.5]	[482.2, 577.9]
	0.959	0.975	0.972	0.967	0.974	0.969
phl_i	[0.936, 0.949]	[0.943, 0.970]	[0.941, 0.968]	[0.939, 0.956]	[0.944, 0.967]	[0.935, 0.952]
	0.026	0.030	0.030	0.028	0.030	0.029
phu_i	[0.007, 0.026]	[0.007, 0.031]	[0.008, 0.032]	[0.007, 0.031]	[0.008, 0.031]	[0.006, 0.030]

Peers outside the considered ASs are classified in the INTERNET class.

We consider the partition with five ASs that we used to validate our model in Section 6.2 and values for ρ_c equal to 0 for all ASs but AS 19548 that is the seed in this scenario with $\rho_c = 10^{-2}$. To evaluate the impact of locality awareness in creating and maintaining the application network topology we modified the snapshots in the following way: we rewired the connections between an ultra-peer and its leaves in order to obtain increasing fraction of leaves that belong to the same AS to which the ultra-peer belongs. We left unchanged the total number of leaves for each ultra-peer and the number of ultra-peers connected to a leaf. After this modification we estimated the distributions that define the application topology between ultra-peers and leaves, i.e., $\{u_{i,j,k}\}$, $\{l_{i,j,k}\}$, in the same way as we did in Section 6.2 to validate our model. Furthermore, we considered an ideal locality-aware algorithm for the overlay network among ultra-peers whose goal is to let each ultra-peer establish at least m connections to other ultra-peers in the same AS. To model the effect of such algorithm we modified the distributions $\{o_{i,i,k}\}$ in the following way for each class i in the partition:

$$o'_{i,i,k} = \begin{cases} 0 & \text{if } k < m \\ o_{i,i,m} + \sum_{h=0}^{m-1} o_{i,i,h} & \text{if } k = m \\ o_{i,i,k} & \text{if } k > m \end{cases}$$

Table 3
Validation results where the IP addresses of peers are partitioned into 3 classes represented by AS numbers

phl_{AS_i,AS_j}	AS 7132	AS 20115	AS 1668	INTERNET
AS 7132	0.630 [0.565, 0.6622]	0.043 [0.032, 0.090]	0.005 [0.0004, 0.022]	0.017 [0.004, 0.061]
AS 20115	0.621 [0.551, 0.657]	0.043 [0.032, 0.090]	0.004 [0.0001, 0.021]	0.017 [0.003, 0.063]
AS 1668	0.576 [0.508, 0.619]	0.037 [0.028, 0.080]	0.004 [0.0002, 0.019]	0.015 [0.002, 0.052]
INTERNET	0.610 [0.535, 0.636]	0.042 [0.029, 0.086]	0.004 [0.0003, 0.020]	0.017 [0.003, 0.057]
ρ	10^{-3}	10^{-4}	10^{-5}	10^{-6}
$\overline{m}_i$	790.2 [722.8, 840.4]	781.5 [719.5, 819.9]	674.1 [618.6, 744.8]	774.4 [689.9, 818.5]
$\overline{im}_i$	3.9 [2.5, 5.6]	1.2 [1.0, 1.4]	0.0 [0.0, 0.0]	631.8 [570.7, 653.2]
phl_i	0.654 [0.616, 0.679]	0.646 [0.604, 0.675]	0.600 [0.567, 0.629]	0.634 [0.589, 0.649]
phu_i	0.053 [0.017, 0.045]	0.052 [0.016, 0.048]	0.045 [0.014, 0.042]	0.051 [0.016, 0.045]

Peers outside the considered ASs are classified in the INTERNET class.

where $o'_{i,i,k}$ is the modified probability that a randomly chosen ultra-peer in class i has k connections to ultra-peers in class i. To keep the total number of connections to ultra-peers unchanged we also modified the probability distribution considering the remaining classes, therefore we also redefined distributions $\{o_{i,j,k}\}$ with $i \neq j$ in the following way:

$$o'_{i,j,k} = \begin{cases} o_{i,j,k} - (\overline{o'}_{i,i} - \overline{o}_{i,i}) \dfrac{\overline{o}_{i,j}}{\sum_{c=1,c\neq i}^{n_c} \overline{o}_{i,c}} \dfrac{o_{i,j,k}}{\overline{o}_{i,j}} & \text{if } k > 0 \\ 1 - \sum_{h=1}^{\infty} o'_{i,j,h} & \text{if } k = 0. \end{cases}$$

Fig. 2 shows how beneficial, from the system point of view, is the exploitation of locality in topology creation and maintenance algorithms. The graphs show the fraction of query messages that are exchanged between peers in the same AS (denoted as iqf). When the fraction of intra-AS leaves and the value of m increase the fraction of "low cost" transmissions increases dramatically approaching the value of 1.

On the other hand, Fig. 3 depicts the value of phl_{AS} for increasing values of the fraction of intra-AS leaves for each ultra-peer and for increasing values of m. We observe that when the locality is mostly exploited at the level of connections between ultra-peers and leaves (cases $m = 1$ and $m = 5$) it results in a slight improvement in the hit probabilities until the fraction of intra-AS leaves for each ultra-peer reaches a threshold value that is AS-dependent. From values greater than the threshold increasing the fraction of intra-AS leaves for each ultra-peer deteriorates the performance of the search algorithm. In the extreme case where the fraction of intra-AS leaves for each ultra-peer is equal to 1 the topology of the network application collapses into a set of isolated communities that cannot share any resource with other communities. In this case, no diffusion of the resource is possible outside the seed AS and all hit probabilities reduce to 0 for all ASs but the seed.

A different picture appears for $m = 10$ and $m = 14$: in this case the effect of exploiting locality is a stronger segregation of peers inside each AS. It is difficult for leaves outside the seed AS to locate the resource since a large

Table 4
Validation results where the IP addresses of peers are partitioned into 3 classes represented by AS numbers

phl_{AS_i,AS_j}	AS 19548	AS 22773	AS 7132	INTERNET
	0.380	0.068	0.011	0.017
AS 19548	[0.335, 0.466]	[0.043, 0.173]	[0.004, 0.012]	[0.003, 0.061]
	0.355	0.062	0.010	0.016
AS 22773	[0.321, 0.425]	[0.044, 0.154]	[0.004, 0.012]	[0.004, 0.057]
	0.376	0.067	0.011	0.017
AS 7132	[0.331, 0.466]	[0.046, 0.169]	[0.004, 0.013]	[0.004, 0.061]
	0.361	0.064	0.010	0.016
INTERNET	[0.316, 0.428]	[0.041, 0.160]	[0.003, 0.011]	[0.003, 0.057]
ρ	10^{-3}	10^{-4}	10^{-5}	10^{-6}
	796.0	731.4	789.8	772.3
$\overline{m}_i$	[737.1, 833.1]	[654.7, 796.6]	[722.8, 839.7]	[688.0, 816.3]
	1.4	0.7	3.8	590.2
$\overline{im}_i$	[1.1, 1.9]	[0.3, 1.8]	[2.4, 5.5]	[519.7, 622.8]
	0.439	0.412	0.435	0.419
phl_i	[0.399, 0,547]	[0.381, 0.505]	[0.394, 0.548]	[0.379, 0.506]
	0.031	0.028	0.030	0.029
phu_i	[0.013, 0.045]	[0.012, 0.044]	[0.013, 0.046]	[0.013, 0.042]

Peers outside the considered ASs are classified in the INTERNET class.

Table 5
Validation results where the IP addresses of peers are partitioned into 1 class represented by AS numbers

phl_{AS_i,AS_j}	AS 20115	INTERNET
	0.043	0.019
AS 20115	[0.032, 0.090]	[0.003, 0.063]
	0.041	0.018
INTERNET	[0.029, 0.086]	[0.003, 0.057]
ρ	10^{-4}	10^{-6}
	781.1	771.9
$\overline{m}_i$	[718.4, 818.6]	[688.7, 816.6]
	1.2	722.2
$\overline{im}_i$	[1.0, 1.5]	[643.5, 762.1]
	0.061	0.059
phl_i	[0.038, 0.145]	[0.036, 0.135]
	0.0033	0.0032
phu_i	[0.0003, 0.0108]	[0.0002, 0.0100]

Peers outside the considered AS are classified in the INTERNET class.

number of ultra-peers connected to a leaf belongs to the same AS and a large fraction of the neighbors of these ultra-peers reside in the same AS, too. The final effect is that leaves of the seed AS experience a hit probability that approaches the value 1 while leaves outside the seed AS struggle to find a path to the resource.

Table 6
Validation results where the IP addresses of peers are partitioned into 1 class represented by AS numbers

phl_{AS_i,AS_j}	AS 22773	INTERNET
	0.062	0.017
AS 22773	[0.044, 0.154]	[0.004, 0.057]
	0.064	0.018
INTERNET	[0.041, 0.161]	[0.003, 0.058]
ρ	10^{-4}	10^{-6}
	730.9	773.3
$\overline{m}_i$	[654.1, 796.2]	[690.6, 817.7]
	0.7	733.5
$\overline{im}_i$	[0.3, 1.8]	[640.1, 788.0]
	0.079	0.081
phl_i	[0.076, 0.178]	[0.077, 0.180]
	0.0026	0.0027
phu_i	[0.0001, 0.0166]	[0.0001, 0.017]

Peers outside the considered AS are classified in the INTERNET class.

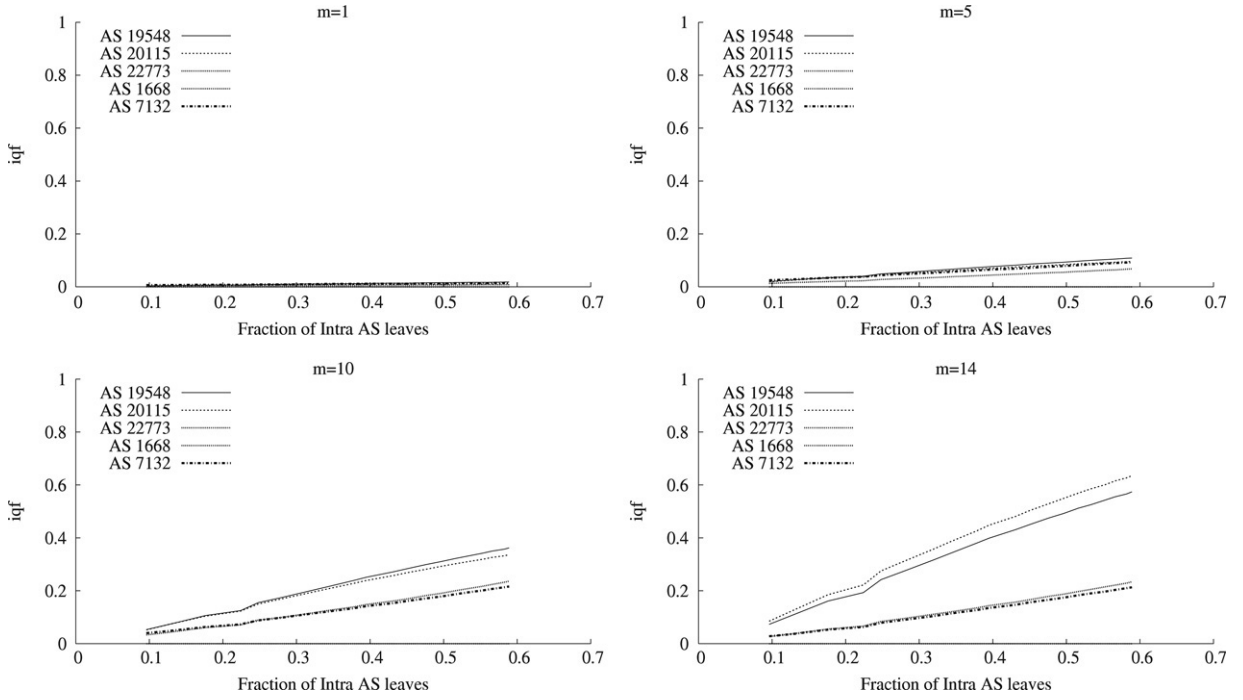

Fig. 2. Fraction of internal query traffic for increasing values for the fraction of intra-AS leaves for each ultra-peer. Each graph refers to increasing values of m: $m = 1$ (top left), $m = 5$ (top right), $m = 10$ (bottom left), $m = 14$ (bottom right).

7. Conclusions and future development

In this paper we developed a random graphs-based performance evaluation techniques to understand design trade-offs for hierarchical unstructured P2P networks. The complex application network topology is represented using two random graph models: the connections between lower and higher level peers are modeled as a bipartite random graph while the overlay network used by ultra-peers to forward queries is modeled as a directed generalized random graph. These two basic models are enriched by considering peers of either level as partitioned into classes. Starting from the probability distributions that define the application network topology we have derived the generating functions of the

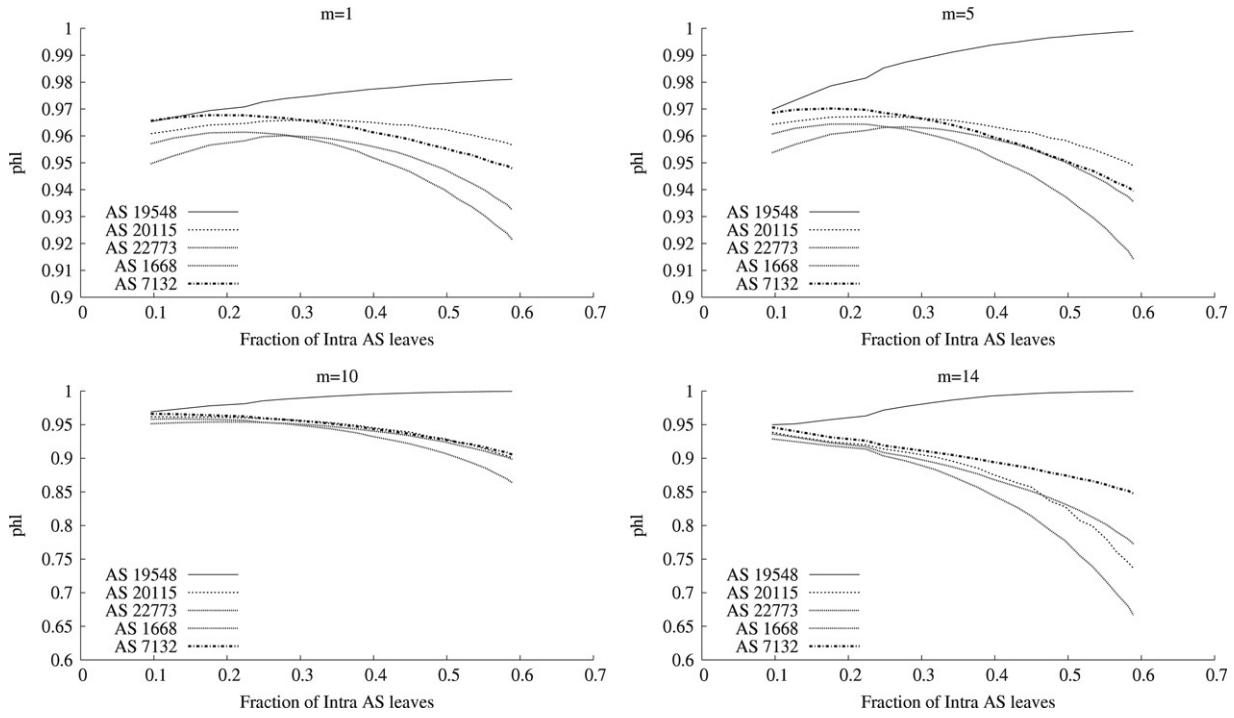

Fig. 3. Hit probability for leaves in each AS for increasing values for the fraction of intra-AS leaves for each ultra-peer. Each graph refers to increasing values of m: $m = 1$ (top left), $m = 5$ (top right), $m = 10$ (bottom left), $m = 14$ (bottom right).

probability distribution of several interesting random variables that we consider as performance indexes. In particular, we derived the characterization of hit probabilities for lower level peers in each class as well as hit probabilities in a particular target class. We also characterized the random variable describing the number of messages sent during the entire query management as well as the number of messages that are exchanged between peers of the same class.

The model predictions have been validated against simulation results computed on snapshots of the Gnutella network that we obtained from the previous work by other authors and from our own crawler. We proved that the model results are in excellent agreement with the outcome of simulations on several complex and heterogeneous scenarios we considered. Successively, we used the model to investigate the effect of locality awareness in algorithms for the creation and maintenance of the application network topology. We observed that pushing the exploitation of locality to the extreme deteriorates the performance of search algorithms and therefore should be fine tuned in real implementations.

The current work can be extended in several ways: a more sophisticated flooding or random walk-based algorithm could be sought for forwarding queries in the overlay network. Our model can be easily extended to include these variations. Another extension is the exploitation of the model to support the analysis of peering or transit agreement between ISPs when the main source of traffic over their access links is generated by P2P applications. In this case, we should derive the g.f. for the random variable describing the number of messages that cross the boundaries of class i. This can be easily done by arranging Eqs. (9)–(11) to consider peers where either the source or the destination class (but not both) is equal to i. As a last possibility, Dynamic Querying could be incorporated into the model derivation by generalizing the equations that represent reachability in the top-level overlay, i.e., Eq. (1), by introducing a probability p_{fwd} that is used to select a subset of neighbors to which a query should be forwarded. By increasing p_{fwd} it is possible to represent the effect of adapting the number of ultra-peers that receive a query to forward as a function of the number of hits obtained by searches with lower values of p_{fwd}.

Acknowledgements

We are extremely grateful to the anonymous referees for their helpful and constructive comments. We thank the authors of [8] for providing us with a few snapshots of Gnutella.

References

[1] The true picture of peer-to-peer file sharing, Tech. Rep., Cachelogic Research. URL: http://www.cachelogic.com/research/, July 2004.
[2] G. Development Forum, The Gnutella v0.6 protocol. URL: http://groups.yahoo.com/group/thegdf/files/, 2002.
[3] F. Peer-to-Peer Technology company. URL: http://www.fasttrack.nu/, 2002.
[4] K. Media Desktop. URL: http://www.kazaa.com/, 2002.
[5] DirectConnect Protocol Specification. URL: http://sourceforge.net/docman/?group_id=36589, 2001.
[6] A.S. Baset, H. Schulzrinne, An analysis of the skype peer-to-peer internet telephony protocol, in: Proc. IEEE Infocom 2006, Barcelona, Spain, 23–29 April 2006.
[7] M. Ripeanu, I. Foster, Mapping gnutella network, in: Proc. of the First International Workshop on Peer-to-Peer Systems, IPTPS 2002, in: LNCS, vol. 2429, Springer, Cambridge, MA, USA, 2002.
[8] D. Stutzbach, R. Rejaie, S. Sen, Characterizing unstructured overlay topologies in modern P2P file-sharing systems, in: Proc. of the ACM SIGCOMM Internet Measurement Conference, Berkeley, CA, USA, 2005.
[9] M.E.J. Newman, S.H. Strogatz, D.J. Watts, TRandom graphs with arbitrary degree distributions and their applications, Physical Review E 64 (2001).
[10] L.A. Adamic, R.M. Lukose, A.R. Puniyani, B.A. Huberman, Search in power-law networks, Physical Review E 64 (2001).
[11] K. Kant, An analytic model for peer to peer file sharing networks, in: Proceedings International Communications Conference, Anchorage, AL, USA, May 2003.
[12] R. Gaeta, G. Balbo, S. Bruell, M. Gribaudo, M. Sereno, A simple analytical framework to analyze search strategies in large-scale peer-to-peer networks, in: Proceedings of the Conference Performance 2005, Performance Evaluation 62 (1–4) (2005) 1–16.
[13] N. Sarshar, V. Roychowdury, P. Boykin, Percolation search algorithm, making unstructured P2P networks scalable, in: Proc. of the Fourth IEEE P2P 2004, Zurich, Switzerland, IEEE Computer Society Press, 2004.
[14] C. Gkantsidis, M. Mihail, A. Saberi, Hybrid seach schemes for unstructured peer-to-peer networks, in: Proc. IEEE Infocom 2005, Miami, FL, USA, IEEE Comp. Soc. Press, 2005.
[15] Cachelogic Research, Peer-to-Peer in 2005, URL: http://www.cachelogic.com/research/.
[16] Meeting the Challenge of Today's Evasive P2P Traffic, Sandvine co., An Industry White Paper, Waterloo, Canada, 2004.
[17] Y. Liu, X. Liu, L. Xiao, L.M. Ni, X. Zhang, Location-aware topology matching in P2P systems, in: Proceedings of INFOCOM 2004, Hong Kong, March 2004.
[18] Y. Liu, L. Xiao, X. Liu, L.M. Ni, X. Zhang, Location awareness in unstructured peer-to-peer systems, IEEE Transactions on Parallel and Distributed Systems 16 (2005) 163–174.
[19] A. Fisk, Gnutella Dynamic Query Protocol v0.1, Tech. Rep., Gnutella Developer's Forum, 2003.
[20] T. Klingberg, R. Manfredi, Gnutella 0.6, Tech. Rep., URL: http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html.
[21] D. Garlaschelli, M. Loffredo, Patterns of link reciprocity in directed networks, Physical Review Letters 93 (26) (2004).
[22] Autonomous System Number ASN extension for Webalizer. URL: http://www.init7.net/webalizer_asn/, 2005.

Rossano Gaeta was born in Torino, Italy in 1968. He received his Laurea and Ph.D degrees in computer science from the University of Torino in 1992 and 1997, respectively. He is currently associate professor at the Computer Science Department of the University of Torino. His research interests include the use of formal models for dependability analysis of time critical systems and the performance evaluation of peer-to-peer computing systems and large scale sensor networks.

Matteo Sereno was born in Nocera Inferiore, Italy. He received his Laurea degree in Computer Science from the University of Salerno, Italy, in 1987, and his Ph.D. degree in Computer Science from the University of Torino, Italy, in 1992. He is currently full professor at the Computer Science Department of the University of Torino. His current research interests are in the area of performance evaluation of computer systems, communication networks, peer-to-peer systems, sensor networks, queueing networks and stochastic Petri net models.