POLITECNICO DI TORINO
Repository ISTITUZIONALE

Locating Equivalent Servants over P2P Networks

Original
Locating Equivalent Servants over P2P Networks / Marchetto, G., Ciminiera, L., PAPA MANZILLO, M., Risso, F.G.O., Torrero, L.. - In: IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT. - ISSN 1932-4537. - 8:1(2011), pp. 65-78. [10.1109/TNSM.2011.012111.00013]

Availability:
This version is available at: 11583/2377002 since:

Publisher:
IEEE

Published
DOI:10.1109/TNSM.2011.012111.00013

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

20 June 2026

This is an author's version of the paper

Marchetto G., Ciminiera L., Papa Manzillo M., Risso F., Torrero L.
"*Locating Equivalent Servants over P2P Networks*"

Published in

IEEE Transactions on Network and service Management, vol. 8, n. 1, pp. 65-78

The final published version is accessible from here:

http://dx.doi.org/10.1109/TNSM.2011.012111.00013

©2011 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Locating Equivalent Servants over P2P Networks

Guido Marchetto, Luigi Ciminiera, Marco Papa Manzillo, Fulvio Risso, Livio Torrero, *Members, IEEE*

***Abstract*—While peer-to-peer networks are mainly used to locate unique resources across the Internet, new interesting deployment scenarios are emerging. Particularly, some applications (e.g., VoIP) are proposing the creation of overlays for the localization of services based on equivalent servants (e.g., voice relays). This paper explores the possible overlay architectures that can be adopted to provide such services, showing how an unstructured solution based on a scale-free overlay topology is an effective option to deploy in this context. Consequently, we propose EQUATOR (EQUivalent servAnt locaTOR), an unstructured overlay implementing the above mentioned operating principles, based on an overlay construction algorithm that well approximates an ideal scale-free construction model. We present both analytical and simulation results which support our overlay topology selection and validate the proposed architecture.**

***Index Terms*—Distributed services, equivalent servants, peer-to-peer overlays, scale-free topology.**

I. Introduction

WHILE in the past few years the resource sharing services across the Internet focused on generic storage (e.g., distributed file systems, remote disks), content (e.g., file sharing, video streaming), and CPU cycles, the recent emerging of the cloud computing paradigm might push this vision even further. According to this scenario, the world will be populated by thin and light computing devices acting mainly as frontends, while the computation and the user's data reside elsewhere, in the "cloud". In those services, two groups of entities are defined: "users", that ask for a given service, and "servants" that are actually in charge of providing the service. Servants can be composed of millions of processing platforms either sparse across the Internet, or concentrated in special datacenters. Users do not care about their physical location: they are interested in getting the service, no matter which servant is actually providing it.

At the same time, the current wave of distributed sharing services tends to involve resources available at the edge of the network and hence bases on the peer-to-peer (P2P) paradigm to achieve performance, scalability, and robustness. Among the possible examples, the *Desktop Grid* computing exploits unused resources (storage, computational power, etc.) available on widely located (home) computers, while *NaDa* [1] uses P2P technologies to build "Nano Data Centers" that exploit the DSL gateways placed in our homes. The idea is that users owning enough resources (e.g., a DSL gateway or a home-PC, which are unused for a great portion of time) may enter the cloud and start offering services.

Manuscript received March 2, 2010; revised August 6, 2010 and November 4, 2010. The associate editor coordinating the review of this paper and approving it for publication was R. Stadler.

The authors are with the Dipartimento di Automatica e Informatica, Politecnico di Torino, 10129 Torino, Italy (email: name.surname@polito.it).

In this context, a new set of services is emerging, where every servant is potentially able to satisfy users' requests. In fact, many operations delegated to the cloud (especially by thin clients) often require "limited" resources in terms of bandwidth, storage or CPU cycles, and therefore can be easily handled by any of the many peers participating in the abovementioned service-oriented overlays. We can say that these services are based on multiple, equivalent servants. As a few examples, we can cite the offloading of some computations that are too expensive for mobile devices, the localization of a relay required for anonymizing a communication (e.g., Tor [2]) or establishing a successful VoIP transfer (e.g., Skype [3]), the necessity to keep the state of users in an online game [4], or a Personal Video Recorder that temporarily stores TV streams when the user is offline, not to mention new online-based computational platforms (e.g., Google Chrome OS [5]). In this scenario, applications require the localization of an available servant (i.e., a node that is currently free and hence can offer the service) in the shortest time, rather than a precise resource localization (e.g., a precise document, or a host with a given amount of CPU time available or at least N Megabytes of spare space).

Existing works lack in providing adequate support to these emerging distributed systems. In fact, most of them focus on the development of a system supporting specific requests, ranging from a unique specific file to a set of resources characterized by well-defined parameters. While these systems can also support the localization of equivalent servants, they are not optimized for this purpose because of the different requirements they comply with, more stringent in terms of resource constraints, but simpler in terms of timely response. Hence, for example, they might be unable to locate a serving node in a very short time, such as a relay to be used in an incoming VoIP call. Furthermore, they may insert an unnecessary overhead in the servant lookup, due to the features they provide to support complex queries, which are of little help in the context of services based on equivalent servants.

This paper focuses on services provided by equivalent servants and models and analyzes the performance of structured and unstructured overlays when used to provide such services. We demonstrate that the architecture chosen for the P2P network has a huge impact on the overall performance of the service. In particular, with the support of some analytical and simulation results, we show how an unstructured network based on epidemic dissemination and built over a scale-free overlay topology is an effective solution to deploy in this context. Then, we present EQUATOR (EQUivalent servAnt locaTOR), a P2P-based architecture deployable in real networks for the provision of services based on equivalent servants. EQUATOR aims at guaranteeing high lookup performance, as well as high robustness to failures and churn phases, when

a significant number of peers joins/leaves the network.

After a brief revision of the related work concerning the existing service-oriented overlays (Section II), the paper introduces some possible overlay architectures that can be adopted to support the location of equivalent servants and shows the benefits of scale-free networks in this particular context (Section III). Then, Section IV introduces EQUATOR and describes its operating principles. An extensive simulation study is presented in Section V to evaluate and validate the proposed solution. Finally, Section VI concludes the paper.

II. Related work

During the last few years, *structured* (e.g., Chord [6], Kademlia [7]) and *unstructured* (e.g., Gnutella [8], KaZaA [9]) P2P solutions have started to be adopted as building blocks for the definition of more complete P2P systems able to provide arbitrarily complex distributed services. For example, [10] and [11] present two similar unstructured architectures for the provision of Grid-like services. Other solutions have been proposed in the context of video distribution (e.g., [12], [13]). On the structured side, some example of these architectures have been presented in [14]–[17].

However, all these proposals address a problem that is different from the scenario we have in mind, where users are interested in locating one of the many available servants. Even more important, they do not investigate the effects of the overlay topology on the performance of this type of resource lookup in order to determine the best overlay technology for the given service.

The equivalence of servants is considered in [18]–[20]. In [18], the authors propose a scheme for CPU cycle sharing over an unstructured P2P network. They consider the unbalanced node degree distribution, which may result in real overlay networks, as a possible obstacle to the lookup effectiveness of the system and, consequently, they propose mechanisms to overcome these limitations. In this paper, we show instead how an unbalanced node degree distribution (specifically, a scale-free topology), if properly exploited, ensures high lookup performance. Peer-to-peer SIP (P2PSIP [19]) proposes to use a DHT to support lookups of relay nodes among all the equivalent participating peers, which can be done by randomly selecting a target node and then moving over the DHT to reach this target. Our previous work [20] explores the idea of a service based on equivalent servants, but it limits its application to a distributed connectivity service in a SIP infrastructure.

This paper focuses on services based on equivalent servants and brings several contributions to the existing work on this topic. First, we compare the possible overlay architectures to support our class of services and we show, through extensive analytical and simulation studies, that an unstructured overlay based on a scale-free topology is an interesting solution in this context. Furthermore, we show the corresponding penalty in case a DHT architecture is chosen, as proposed in [19]. Second, we propose a novel overlay construction algorithm which (*i*) is suitable for implementation in real networks, (*ii*) supports a generic service, and (*iii*) approximates an ideal well-known scale-free construction model. Third, we analyze different network scenarios by varying the servant characteristics (e.g., their lifetime), which provides an insight of the possible performance of different services in our context.

III. Overlay architecture options

Since the underlying overlay architecture has a huge impact on the performance of the offered service and on the features that can be guaranteed to the users, this section compares the structured and unstructured approaches with respect to their capability to support services based on equivalent servants. In particular, we focus on the service lookup performance (i.e., the capability of the system to provide a querying user with an available servant) offered by different architectures, presenting some analytical and simulation results which demonstrate that an unstructured overlay based on a scale-free topology is a good choice for handling our service. Then, we elaborate on the other interesting properties of this solution.

A. Structured overlays

We first investigate the possibility to deploy a structured overlay based on a general DHT, as it has been proposed in [19] for the P2PSIP architecture.

Since in our scenario all peers provide the same functionality (i.e., we have only one resource provided by many nodes), the number of copies predominates over the number of distinct services and therefore the ability of DHTs to locate a specific resource is of little help. Therefore, [19] proposes to use the DHT in a more clever way: queries are performed by randomly selecting a target key and then moving in the overlay to reach this target.

Since it does not cause further complexity and possibly improves the system performance, we introduce an additional feature to this querying mechanism: during the lookup process, any node encountered along the path is checked for availability and can be selected as a servant for the querying user. Notice that this operating mode makes the approach independent of the adopted DHT. In fact, only the overlay topology (which is a regular graph in existing DHTs) is of interest in our context. In other words, we adopt the topology of a generic DHT, with a fixed number of neighbors for each node, but we use a different routing mechanism. This solution will be however referred to as DHT in the rest of the paper.

The idea of using a DHT for our scenario of equivalent servants is especially interesting in case a DHT has to be implemented anyway for some other services. For example, P2PSIP already uses a structured overlay to index all possible targets of a multimedia communication, i.e., all the user agents registered in the SIP domain. Using the same DHT to locate, if necessary, a relay node to support the communication (i.e., a servant among the many peers existing in the SIP domain) may be a considerable advantage for that application, which needs to maintain only one overlay structure that can be used for both functions.

B. Unstructured overlays

An efficient unstructured overlay is characterized by high lookup performance and small amount of traffic required to maintain the overlay. Both parameters are influenced by the topology and the operating principles (e.g., how nodes spread information) of the overlay. This section elaborates on these aspects in the context of services based on equivalent servants, proposing to adopt a scale-free topology and motivating this choice.

An interesting lookup solution that avoids the deleterious traffic overhead generated by flooding-based queries is the adoption of a service lookup based on *random walks* [21] encompassing a bounded number of nodes. Within this technique, the service request is forwarded, at each node, to a peer randomly selected among its neighbors. If the encountered node is available or knows an available servant, the procedure terminates. The knowledge of nodes can be improved through proper advertisement messages containing the node itself and/or other participating peers, thus implementing a so called *epidemic dissemination algorithm*.

The effectiveness of random walks depends on the overlay topology adopted in the system. Among other possibilities, a scale-free topology [22] may offer interesting features. In a scale free network, the node degree distribution follows a power-law $P(n) = cn^{-\gamma}$, where $P(n)$ is the probability that a node has n connections and c is a normalization factor. Hence, only few nodes (usually referred to as *hubs*) have a high degree, i.e., are aware of the existence of a large number of participating peers. The idea is that directing random walks toward hubs means looking for the service where there is a great knowledge of servants. This ensures high lookup performance with respect to an overlay based on a balanced degree distribution (e.g., a random graph or a regular topology) where service requests are randomly distributed among peers. This result derives from a well-known property of queuing systems, which says that a unique M/G/k/k queuing system servicing an arrival process with rate λ performs better than k separated M/G/1/1 systems each one servicing an arrival process with rate λ/k. In essence, concentrating the traffic on some nodes that have a deep knowledge of the network (i.e., the hubs, which know a lot of possible servants) provides better performance than accurately distributing the requests among all nodes, as random solutions try to do. This extends the results obtained by Adamic et al. [23] in the context of traditional file lookups in P2P systems, which demonstrated the effectiveness of random walks in scale-free networks due to the greater knowledge of resources available at the hubs.

In order to achieve high lookup performance, hubs should have a deep knowledge about the other participating peers: the greater the number of peers known by a given node, the higher the probability for a user to find an available servant in a short time. Since the epidemic dissemination is based on flooding, the overlay topology has a deep impact not only on peers known by each node, but also on the resulting network efficiency. In fact, the greater the average path length between nodes, the higher the depth of the flooding that is needed for an adequate spread of the information, which may cause an unsustainable load on the network. The scale-free topology also ensures a good efficiency of epidemic dissemination algorithms as exhibits a small average path length. In essence, a large number of advertisement messages reach the hubs even with a small dissemination depth (namely, the number of hops encompassed by advertisement messages before elapsing) and a small out-degree (representing the number of peers to which a node directs advertisement messages).

Another interesting feature of scale-free networks is that they can scale to an arbitrarily large network size without modifying the degree distribution of nodes, which continues to follow the same law. This ensures that new hubs are automatically created when the network size grows, therefore maintaining the above described properties. In essence, scale-free networks potentially combine the advantages of centralized indexing (where a single entity directly handles all possible servants and consequently offers the best performance) and totally distributed solutions (which can scale to an arbitrary large number of participating servants and users).

One of the most popular mechanisms to build a scale-free network was proposed by Barabási and Albert [22] and for this reason is referred to as Barabási-Albert model. Let m denote the out-degree of a node and d denote its in-degree. The Barabási-Albert model requires a set of m_0 nodes to be already in the system at the beginning of the process. Then, each entering node connects to m existing nodes, chosen proportionally to their popularity. This process is known as *preferential attachment*. This network formation algorithm results in a scale free network characterized by a node degree distribution $P(n) = cn^{-3}$ and an average path length which behaves as $\frac{\ln N}{\ln \ln N}$ [22]. The Barabási-Albert model is used as a reference in the rest of the paper. Although in general $P(n) = P(m + d)$, in this case we are interested in the in-degree of a node as it represents its popularity, i.e., it counts the number of nodes that send their advertisements to it. Thereby, without losing in significance, we consider $P(n) = P(d)$ — i.e., the distribution of the in-degree of nodes — in the following.

The Barabási-Albert model is an ideal network formation algorithm that requires a global knowledge of the existing nodes. Clearly, this is not feasible in a real network. Hence, while this section shows the effectiveness of a scale-free solution, Section IV will present an overlay construction algorithm based on a limited network knowledge which approximates the Barabási-Albert model.

C. A lookup performance model

This section compares the above architectures with respect to their capability in locating an available servant. This result is achieved by defining a simple analytical model that derives the average blocking probability (i.e., the probability for a service request to fail because no available servant is found) achieved by each architecture.

1) Model overview: From our point of view, the length of the path that a service request has to follow to reach a target key in a DHT and the depth of a random walk over an unstructured network have a similar meaning: they represent

the amount of hops that a service request can encompass without success before the request has to be considered blocked. Hence, without losing in generality, we denote these two overlay parameters by a common variable, namely D_l, generally defined as the maximum depth of a service lookup. D_l is a fixed value in an unstructured network, while is variable and $O(\log N)$ in a DHT.

For the sake of simplicity, we consider only the case $D_l = 1$ in this model. Within the unstructured approach, we also assume a dissemination depth (i.e., the time-to-live of advertisement messages, denoted as T_d) not greater than 2 hops, as larger values would result in an excessive dissemination traffic overhead in the network. This assumption is confirmed by the guidelines of the Gnutella protocol [8], in which the depth of the dissemination algorithm is set to a maximum of 2 hops.

Let $V = \{v_1, v_2, \ldots, v_N\}$ denote the set of participating peers offering the service, and $\mathcal{S}_i$ denote the set of servants indexed by a given node v_i (including the node itself), i.e., the set of peers that the node v_i can offer to a querying user in the tentative of satisfying her service request. The idea is that, whenever a service request reaches a node v_i of the overlay, such request is satisfied if a servant $s_i \in \mathcal{S}_i$ is available.

Hence, under the assumption $D_l = 1$ and if service requests are supposed to arrive at nodes according to a Poisson process, each node can be modeled as an M/G/k/k queuing system, i.e., a buffer-less system offering k equivalent servers, as also briefly described in Section III-B. End-systems may be part of the overlay if the offered service consumes a small fraction of the available resources, so that local users are not penalized. Hence, we suppose that a node can be a servant only for one user at a time, i.e., for a given node v_i, $k_i = |\mathcal{S}_i|$. This could not be the case in some scenarios, where the offered service consumes a very low percentage of resources. However, it is worth noticing that our model is still valid: if each node can support n service instances, we would have $k_i = n\,|\mathcal{S}_i|$. Similarly, the analysis could be extended to the case where each node can handle a different number of service requests. However, this would complicate the analysis without adding any significant contribution to the comparison among the architectures considered in the paper.

In an M/G/k/k queuing system, the probability that a service request fails (i.e., the blocking probability, which we denote as P_b) can be evaluated by using the well-known Erlang B formula. Let λ_i and μ_i denote the request arrival rate and the service rate at node v_i, respectively. For each node v_i we have

$$P_{b_i} = \frac{{\rho_i}^{k_i}/k_i!}{\sum_{n=0}^{k_i} \rho^n/n!}, \tag{1}$$

where $\rho_i = \lambda_i/\mu_i$ is defined as the *service request load* at node v_i. Clearly, for a given node v_i, P_{b_i} depends on ρ_i and on the amount of servants the node can offer, k_i.

In the next sections we will derive ρ_i and k_i for both the structured and the unstructured scale-free approach. This will be used to calculate the average blocking probability of the system, which allows us to quantitatively compare the two approaches under examination when used to locate equivalent servants. In particular, if ρ_T is the total service request load offered to the overlay, the average blocking probability can be evaluated as

$$P_b = \sum_{i=1}^{N} \frac{\rho_i}{\rho_T} P_{b_i}. \tag{2}$$

2) Structured overlay model: From our point of view, a DHT can be modeled as a regular topology where nodes have a fixed number of neighbors (the out-degree m) given by the size of the tables they use to route queries in the overlay. According to the servant lookup procedure presented in Section III-A, each encountered node along the path toward the target key can satisfy the service request only if the node is available, i.e., $\mathcal{S}_i = \{v_i\}$ and, consequently, $k_i = 1$, $\forall v_i \in V$. We assume that incoming queries can enter the network at nodes selected randomly, as it may happen in real DHTs. Also remembering our main assumption $D_l = 1$, we have $\rho_i = \frac{\rho_T}{N}$, $\forall v_i \in V$, where N is the overlay size, i.e., the number of peers participating to the overlay. The average blocking probability is obtained by substituting k_i and ρ_i in (1) and (2).

3) Unstructured scale-free overlay model: In an unstructured scale-free overlay, the number of servants $|\mathcal{S}_i|$ offered at a node v_i strictly depends on the amount of other peers which advertise themselves to the node, which varies according to the dissemination depth T_d adopted in the network. We consider a scale-free network constructed according to the Barabási-Albert model, which represents the scale-free construction algorithm we adopt as a reference in this paper; we consider $m_0 = m$ for simplicity.

Let A_i denote the amount of messages arriving at node v_i in one advertisement round from peers directly connected to node v_i and from which v_i is reached by a 1-hop-depth dissemination. In this simple case, a node v_i can receive advertisement messages only from its direct neighbors. Clearly, $A_i = d_i$, where d_i is the in-degree of node v_i. In the Barabási-Albert model, the in-degree of a node may vary whenever a new node joins the network. In particular, the probability $P_{n,i}$ that an entering node v_n connects to an existing node v_i, thus modifying its degree, is given by

$$P_{n,i} = \frac{d_i(n) + m}{2n}, \quad n > i, \tag{3}$$

where $d_i(n)$ is the in-degree of node v_i when node v_n joins the network. This time dependence can be calculated by applying the *continuum theory*, introduced in [22] for this purpose. The outcome of this theory is that, at a given time t, $d_i(t) = m\sqrt{(t/i)} - m$. Thereby, we can argue that, for a network size N (i.e., at "time N"), the amount of messages arriving at node v_i in one advertisement round when $T_d = 1$ is

$$A_i = m\sqrt{\frac{N}{i}} - m. \tag{4}$$

Analogously, let A'_i denote the number of messages arriving at node v_i in one advertisement round from peers connected to the direct neighbors of node v_i and from which v_i is reached by a 2-hop-depth dissemination. We define these nodes as "second-hop neighbors" of v_i. In this case, the calculation of the number of advertisement messages A'_i is more difficult

as it is no longer deterministically related to the in-degree d_i of the node. For this reason, we focus our analysis on the average number of received advertisement messages, which is more tractable and does not preclude the validity of the model. In particular, considering that the average in-degree of the neighbors of node v_i can be evaluated as $\sum_{n=i+1}^{N} c\, d_n P_{n,i}$ and that, from the continuum theory, $P_{n,i} = (m/2)(n\,i)^{-0.5}$, we can derive the average number of advertisement messages generated by the second-hop neighbors of v_i as follows:

$$< A'_i > \;=\; d_i \sum_{n=i+1}^{N} c\, d_n P_{n,i} \approx d_i \int_{i+1}^{N} c\, d_n P_{n,i}\, \mathrm{d}n \approx d_i\, c\, m^2 \sqrt{\frac{N}{i}} \left[\frac{1}{2} \ln\left(\frac{N}{i}\right) + \sqrt{\frac{i}{N}} - 1 \right], \quad (5)$$

adopted as an estimation of A'_i in the following. The normalization factor c can be evaluated by imposing $\sum_{n=i+1}^{N} c\, P_{n,i} = 1$.

The set $\mathcal{S}_i$ of the servants indexed at a node v_i is composed of the node itself and the peers it discovers through the epidemic dissemination mechanisms. The m peers a node v_i is connected to (i.e., its out-degree) are assumed to index other servants and contacted in the case the service cannot be satisfied at node v_i. It is worth noticing that, while for a node v_i, $|\mathcal{S}_i| = A_i + 1$ if $T_d = 1$ (i.e., the number of servants indexed at the node is equal to the number of advertisement messages received in each advertisement round, plus the node itself) a similar consideration is in general not correct if $T_d = 2$, i.e., $|\mathcal{S}_i| \neq A_i + A'_i + 1$. This is due to the fact that, when $m \geq 1$, the second-hop neighbors discovered may be not unique and we may count the same node twice. This may happen when two first-hop neighbors have an additional direct connection between them (therefore they both appear as second hop neighbors as well) or when the same second-hop neighbor is reached through two different first-hop neighbors.

Concerning the first type of duplicated node, we can obtain an approximate evaluation of the average number of links among the direct neighbors of a node v_i as follows:

$$L_i \;\approx\; \frac{1}{2} \int_{i+1}^{N} \mathrm{d}l \int_{i+1}^{N} \mathrm{d}n P_{l,i}\, P_{n,i}\, P_{l,n} \approx \frac{m^3}{16i} \left[\ln\left(\frac{N}{i}\right) \right]^2,$$

derived from the definition of "average clustering coefficient" introduced in [24] by considering that in the Barabási-Albert model a node v_n can be connected to node v_i only if $n > i$. Analogously, we can evaluate the average number of duplications involving a second-hop neighbor and two direct neighbors of a node v_i as follows:

$$\begin{aligned} L'_i \;&\approx\; \int_{i+1}^{N} \mathrm{d}l \int_{i+1}^{N} \mathrm{d}n P_{l,i}\, P_{(l+1),i}\, P_{n,l}\, P_{n,(l+1)} \\ &\approx\; \frac{m^4}{16 i^2} \left[\ln\left(\frac{N}{i}\right) + \frac{i}{N} - 1 \right]. \end{aligned}$$

These parameters indicate, for each node, the average number of duplications resulting in the neighborhood of the node, from which an estimation of the number of non-unique discovered nodes could be derived. In particular, we have

$$k_i = a_i A_i + b_i (A'_i - L_i - L'_i) + 1,$$

where $a_i,\, b_i \in \{0, 1\}$, $a_i = 1$ iff $T_d \geq 1$, $b_i = 1$ iff $T_d \geq 2$.

As mentioned before, we believe that a scale-free topology is especially advantageous if we direct service requests to hubs. Hence, we assume

$$\rho_i = \frac{k_i}{\sum_{n=1}^{N} k_n} \rho_T, \quad (6)$$

which corresponds to distributing incoming service requests among nodes proportionally to their popularity.

The above derived values for k_i and ρ_i can be used to calculate the average blocking probability achieved in an unstructured scale-free overlay by applying (1)[1] and (2). However, it is worth noticing that, for a given node v_i, if $v_i \in \mathcal{S}_i$ and $v_i \in \mathcal{S}_j$, v_i is a direct neighbor or a second-hop neighbor of v_j. This makes servants shared among several nodes and then introduces correlations which are not considered in the M/G/k/k system and in the Erlang B formula, which is not valid in such situations. We can model these correlations by introducing an additional load at all nodes, therefore taking into account the service requests directed to the nodes that share some servants. In particular, given an average service request load ρ_i on a node v_i, the average contribution to the load on a node $v_j \in \mathcal{S}_i$ due to v_i is $\rho_j^i = \frac{\rho_i}{k_i}$. From (6), we can derive that this contribution is constant and equal to $\frac{\rho_T}{\sum_{n=1}^{N} k_n}, \forall v_j \in V$. This considered, we can argue that the additional load to consider at a node v_i is $\rho_{i_{add}} = (m^2+m)\rho_i$, where ρ_i is derived from (6). This approach for considering correlations deriving from the sharing of servants among nodes is approximated and, as shown in the next section, holds only for small values of m. Specific analytical work would be required to exactly model this phenomenon, which is however left for future work.

D. Lookup performance comparison

The above presented analytical model is used for comparing the structured and the unstructured scale-free approaches concerning the average blocking probability they achieve. A network size of $N = 5000$ nodes is considered for this comparison. Furthermore, we assume an exponential service time distribution with rate $\mu_n = \mu$, $\forall n : 1 \leq n \leq N$. In absence of any more detailed information about the possible service time distribution in this particular distributed service scenario, we consider this assumption a good approximation of the actual behavior of possible users, which could be involved in relatively short multimedia communications with higher probability, but also in longer sessions of video-streaming and on-line gaming. In essence, we customize our queuing system to an M/M/k/k. Notice how this does not influence our

[1] Being k_i an average value, it may be non-integer. Hence, we modify (1) as follows: $P_{b_i} = \frac{1}{\rho_i \int_{Q}^{\infty} e^{-\rho_i \gamma} (1+\gamma)^{k_i}\, \mathrm{d}\gamma}$. This is known as Generalized Erlang B formula and can be used in presence of a non-integer number of servers.

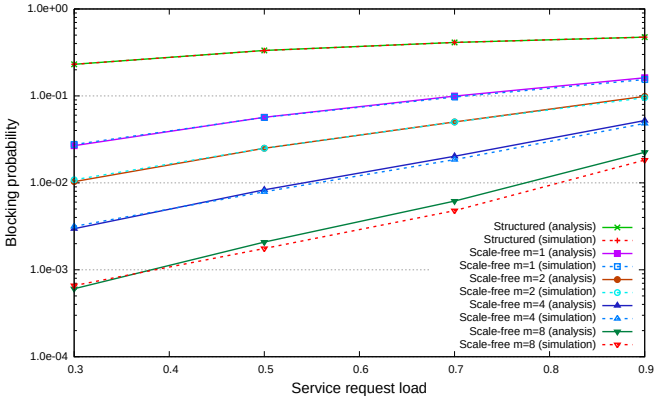


Fig. 1. Average 1-hop blocking probability: comparison between structured and unstructured scale-free overlays.

performance evaluation as the Erlang B formula is insensitive to the service time distribution.

Fig. 1 compares the structured approach and the unstructured scale-free solution adopting an epidemic dissemination depth $T_d = 2$. In particular, it plots the average 1-hop blocking probability (i.e., the average blocking probability obtained when $D_l = 1$) achieved by these two types of network. Some values of the out-degree m are considered for the unstructured approach[2]. The figure shows how the unstructured scale-free approach based on epidemic dissemination and random walks significantly outperforms the modified DHT-based lookup over structured overlays introduced in Section III-A.

To validate our model, we developed a custom, event-driven simulator implementing both the structured and the unstructured scale-free approach. The former exploits a regular topology where service request load is randomly distributed among peers. Concerning the unstructured scale-free approach, the network topology is constructed according to the Barabási-Albert model and the service request load is distributed among peers proportionally to their in-degree, as specified by the model itself. Fig. 1 also compares our analytical model with the results obtained by simulation. We can observe how our approximated approach to address the correlations emerging from the presence of servants indexed at more than one peer, consisting in the introduction of additional load at nodes, is valid only for small values of m. For example, at a service request load $\rho_T = 0.7$, the analytical model provides an average blocking probability which is 20% higher than the real value derived by simulation when $m = 8$. However, it is worth noticing how small increments in the value of m result in sensible improvements of the lookup performance. This is of great importance in our unstructured context, as the traffic overhead generated by flooding of advertisement messages is directly proportional to the out-degree m of the participating peers. Hence, we can concentrate on small values of m, which guarantee small traffic overhead together with excellent lookup performance. For this reason, we consider $m = 2$ in the following.

To further extend our overlay comparison, we consider larger values of D_l. This was not included in our model because of the complex correlations that rise when service requests may experience more than one hop. In fact, requests may arrive at a node after being refused at previous hops, making an analytical modeling difficult. Consequently, we derive this result only by simulation. Moreover, we include additional comparisons which may be of interest for our work. In fact, so far we considered the utilization of the DHT as proposed in [19]. However, a possible more effective approach may be to include epidemic dissemination in the structured overlay, so that nodes may increase the number of servants they can offer to querying users. Since in our context we are interested in the number of node edges (proportional to the number of servants discovered), rather than in the specific peers to which they point, such an approach is expected to have similar performance to an unstructured overlay implementing a *random graph* (also considered in this comparison for the sake of completeness). In fact, a random graph is by definition a quasi-regular topology where node degree assumes values close to the average degree m with high probability [22]. Notice that this analogy does not hold in traditional file-sharing systems, where efficient lookups over structured overlays are guaranteed only if peers establish connections according to well-defined rules.

Fig. 2 compares all these approaches concerning the average blocking probability achieved at different values of D_l in the two different service request load conditions $\rho_T = 0.6$ and $\rho_T = 0.9$. Besides confirming that the unstructured random solution and the structured approach enriched with epidemic dissemination perform similarly, the figure shows how the unstructured scale-free overlay outperforms other solutions. In particular, Fig. 2(b) shows that the lookup performance of a random walk does not degrade too much with the increasing of the traffic intensity, thus being able to effectively support also services requiring real-time lookups.

In summary, the outcome of these analytical and simulation results is that, besides avoiding the complexity due to the maintenance of a structured network, an unstructured overlay based on epidemic dissemination and resulting in a scale-free topology is also preferable for the offered lookup performance. In particular, we observed how such a system can guarantee good lookup performance even in presence of small values of m and T_d ($T_d \leq 2$ in these examples), which are instrumental to limit the overhead in the network due to the flooding of advertisement messages. As described in Section III-B, these properties derive from the large availability of servants at the hubs and the small diameter of scale-free networks.

E. Further properties of the unstructured scale-free overlay

Scale-free networks offer some other properties which could be interesting for the deployment of service-oriented overlays. First of all, we show how the average blocking probability can be further lowered by reducing the number of nodes from which users can start random walks in order to locate a servant. In particular, since one of the key ideas under the adoption of a scale-free topology is to preferably direct queries toward hubs, we can force users to direct their service requests

[2]Notice that the structured overlay is insensitive to the out-degree value when referring to the lookup performance.

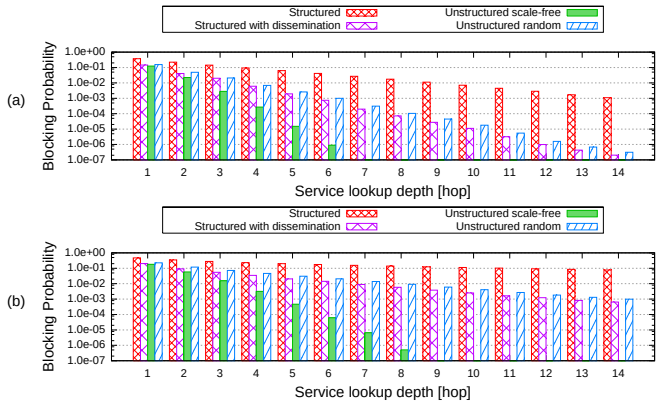


Fig. 2. Average blocking probability as a function of the service lookup depth D_l: comparison among structured, structured with epidemic dissemination, unstructured scale-free, and unstructured random overlays; (a) $\rho_T = 0.6$; (b) $\rho_T = 0.9$.

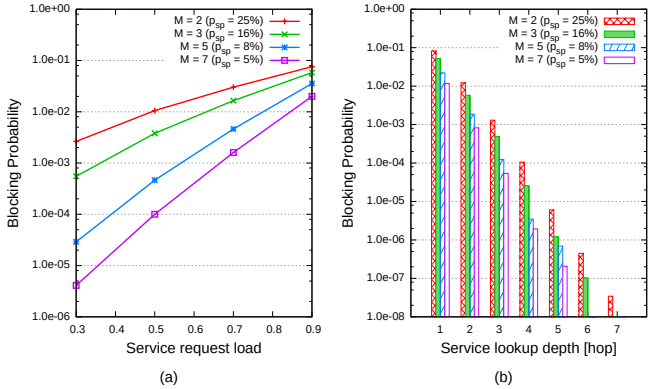


Fig. 3. Lookup performance of a hierarchical scale-free overlay: (a) Average 1-hop blocking probability; (b) Average blocking probability as a function of the service lookup depth D_l ($\rho_T = 0.6$).

to nodes whose in-degree is greater than a given value M. In essence, service requests can be directed to a percentage $p_{sp} = (m^2/(M+m)^2) * 100$ of nodes, as can be derived from the continuum theory. Fig. 3(a) and Fig. 3(b) plot the average 1-hop blocking probability and the average blocking probability as a function of the random walk depth D_l, respectively, achieved by the unstructured scale-free network for some values of M when $m = 2$. A service request load $\rho_T = 0.6$ is considered in Fig. 3(b). We can observe how small values of M, resulting in percentages p_{sp} not lower than 5%, significantly improve the lookup performance of the scale-free overlay. These are admissible values if we consider that very few nodes handle lookup requests in existing hierarchical file-sharing systems (see for example [25], which reports on a measurement study of the KaZaA overlay). Moreover, as it will be clearer in the following, handling a service requests means processing a short packet and replying with some peer descriptors (i.e., a few hundreds of bytes). This produces even less effort than the task usually assigned to peers in hierarchical file-sharing overlays, where they have to handle and maintain all the shared resources.

The other interesting property of these networks is their higher resiliency to random node deletions with respect to overlays based on random graphs [26]. This can be explained observing that most of the paths between nodes pass through hubs: if a peripheral node is deleted, it is unlikely to affect communication between other peripheral nodes. Intuitively, the scale-free with $m \geq 2$ also tolerates the (low frequent) deletion of hubs, since most of the nodes are connected to m hubs for construction, thus preserving service continuity. This makes a scale-free overlay resilient to node failures, which generally occur randomly in a network. However, this solution may appear vulnerable to attacks specifically conducted to destroy a great portion of hubs. This is correct for static scale-free networks, but here we deal with a dynamic overlay where connections among nodes periodically change, as it will be presented in the next section. This network dynamicity leads our scale-free solution to tolerate also these negative events, as it will be shown in Section V-F.

IV. EQUATOR

The previous section demonstrated the effectiveness of an unstructured network based on a scale-free topology. However, both the Barabási-Albert model, adopted for the scale-free construction, and the lookup mechanisms deriving from this approach make some assumptions that cannot be satisfied in the real world; particularly, we envision four problems in the model that require some real-world adaptations. In fact, the Barabási-Albert model requires *(i)* a global knowledge of nodes and *(ii)* their popularity in order to perform the preferential attachment; *(iii)* hubs are supposed to index an arbitrarily large number of servants, which are used to satisfy incoming service requests; finally, *(iv)* nodes are considered static, i.e., the model does not consider nodes joining and leaving the network.

This section presents EQUATOR, an unstructured overlay based on the Barabási-Albert model (and hence on a scale-free topology), which adopts a set of construction and operating rules that are suitable for a real network. Furthermore, an epidemic dissemination algorithm is used to spread the network knowledge among the participating peers. A portion of a possible EQUATOR overlay is shown in Fig. 4 (some details will be clarified in the following), with some peers being part of the scale-free topology and some normal users accessing the offered service.

A. EQUATOR Bootstrap Service

In real P2P networks, entering nodes cannot have any knowledge about the existing overlay and therefore a Bootstrap Service is required in order to give such nodes the opportunity to join the network. In particular, the Barabási-Albert model requires a set of m_0 peers to be available at the initial step of the overlay setup. A simple solution (adopted in many existing overlays such as KaZaA [9]) consists in setting up some static peers and pre-configuring their addresses on each client.

In EQUATOR, we prefer a more flexible approach that relies on multiple bootstrap servers reachable through appropriate DNS records (e.g., `SRV` entries), thus guaranteeing redundancy and load balancing. Bootstrap servers globally store information about m_0 participating peers; when a peer joins

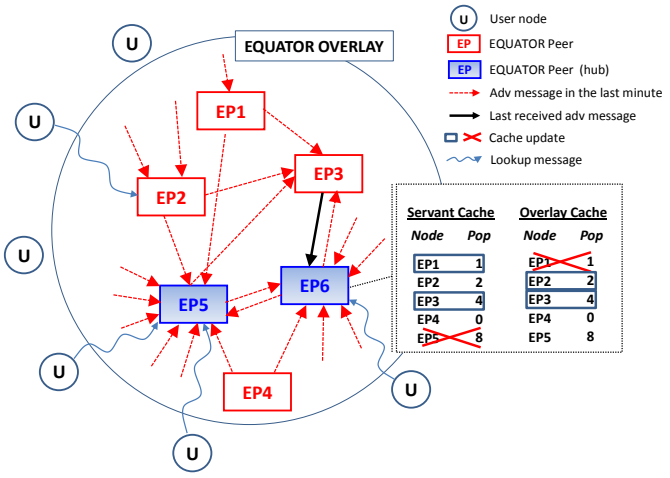


Fig. 4. EQUATOR architecture.

the overlay, it adds itself to the list in case the number of entries in the bootstrap servers is $n < m_0$. Since entries in the bootstrap servers expire after a predefined lifetime, each peer periodically re-contacts the bootstrap servers and potentially adds itself to the list.

B. Node popularity

In a network based on epidemic dissemination, nodes send advertisement messages to other nodes in order to maintain the overlay. Although the details of this advertisement process will be presented in Section IV-C, we need to define first a feasible method for computing the popularity of nodes, which is one of the crucial points of the Barabási-Albert model because it is at the foundation of the preferential attachment policy and hence of the scale-free construction mechanism.

In a scale-free topology the popularity is equivalent to the in-degree of the node. Since it is unfeasible for an EQUATOR node to be aware of its in-degree, EQUATOR adopts as popularity metric the number of advertisement messages a node receives, which is proportional to its in-degree. In particular, a node can estimate its popularity by maintaining statistics about the average number of received messages per minute. The popularity of a node is used both in the overlay construction (to connect to the most popular nodes) and in the overlay maintenance (to keep connections to hubs) and is propagated in the advertisement messages, as detailed in Section IV-C.

C. Overlay knowledge and advertisement

Each node in the overlay maintains two different node caches: a *servant cache* and an *overlay cache*. The former contains the set $\mathcal{S}_i$ of servants indexed by a peer v_i and it is populated by nodes that are lightly loaded with high probability, i.e., nodes (often leaves) that are most appropriate for satisfying an incoming service request. The latter contains a subset of the participating peers representing the entire overlay, among which the node selects the m peers to connect to. Hence, it includes nodes of different popularity in order to better represent the overlay. We denote by τ_{sc} the size of the servant cache and by τ_{oc} the size of the overlay cache.

At each advertisement round (which we suppose to occur every t_{adv} minutes), an EQUATOR node sends an advertisement message (i.e., it "connects") to m peers in its overlay cache, chosen with a probability proportional to their popularity and hence according to the preferential attachment mechanism. These messages contain a list of tuple `<node, popularity, ttl>`: n_{sc} entries are selected as the less popular peers present in the servant cache, while n_{oc} entries are randomly selected from the overlay cache. This is done to give nodes the opportunity to learn both servants that are available with high probability (i.e., the leaves) and a set of nodes of different popularity to improve their local representation of the overlay. In fact, nodes that receive the message insert the n_{sc} entries in the servant cache and the n_{oc} entries in the overlay cache. When caches are full, the n_{sc} entries replace the most popular peers of the servant cache, while the entries replaced by the new n_{oc} nodes in the overlay cache are chosen randomly. Notice that the removal of oldest entries (as proposed in CYCLON [27]) is not a good policy in EQUATOR as it is necessary to maintain the above popularity distributions in the caches. However, entries expire after `ttl` seconds in order to purge old nodes from the cache (if not refreshed) and avoid zombies.

When the dissemination depth $T_d > 1$, nodes along the dissemination path also insert themselves in the advertisement messages before forwarding the message to the next hop. Since these nodes are highly popular peers with high probability (for scale-free construction), this slightly biases the overlay cache with highly popular nodes, with the aim of favoring hubs to be contacted and hence promoting preferential attachment.

An example of the cache update process is shown in Fig. 4, when the EQUATOR peer EP6 receives an advertisement message from EP3 (the solid arrow in the figure). In the figure, a peer announces two peers it knows, one picked from the servant cache and one picked from the overlay cache (in addition to the node itself). We also suppose a cache size $\tau_{sc} = \tau_{oc} = 4$ peers and only one entry of each cache to be empty when the advertisement message arrives. The most popular peer of the servant cache and a randomly selected peer from the overlay cache are removed to accommodate the newly discovered peers.

D. Cache refresh

In EQUATOR, the knowledge of the network at any time t is limited to a few nodes, i.e., the ones that are in the two caches. Apparently, this is a radical departure from the scale-free model in which nodes have the knowledge of the entire network. However, the advertisement policies implemented in EQUATOR allows a frequent update of the two caches, therefore changing the known peers over time. In fact, each node periodically advertises itself and some peers contained in its two caches, so that peers receiving advertisement messages can update their knowledge of the network by filling up, and possibly refreshing, their caches.

Refresh is the key technique that allows the deployment of small caches, which limits overheads due to both cache management and advertisement and lookup traffic (all nodes

in the servant cache have to be contacted during the lookup procedure, as described in Section IV-E). Furthermore, it reduces the possibility to have an old servant, which may be dead or currently unavailable (actually servicing a request) in the servant cache. In fact, a frequent cache refresh ensures the set of indexed servants changes frequently, resulting in a sort of round robin among them. Since the cache refresh rate at a node is proportional to the number of advertisement messages received and, consequently, to its popularity, this effect is maximized at the hubs, which have the opportunity to virtually offer a large number of servants, notwithstanding the limited size of the servant cache.

Frequent entry refresh is also important for the overlay cache to allow the overlay to be dynamic and hence more robust. When a new peer joins, its overlay cache only contains the bootstrap nodes retrieved from the EQUATOR Bootstrap Service. Thanks to the refresh, nodes can insert new peers in their overlay cache and update the popularity information of the peers they already knows. This increasing knowledge of the network allows nodes to incrementally contribute to the construction of the scale-free topology as they can apply a more and more accurate preferential attachment. Hence, the overlay results in a scale-free topology, although variable over time. Furthermore, a frequent refresh ensures nodes are aware of live peers and hence well connected to the rest of the overlay.

E. Service lookup procedures from normal users

While the overlay contains all the peers that are available to offer some of their resources (i.e., are potential servants), many hosts may join the system as normal users in order to simply exploit the overlay services and without taking an active part in the overlay.

Users are most interested in service lookup functionalities and therefore have an advantage at connecting to peers that know many servants. In fact, in our model service requests are distributed among the participating peers proportionally to their popularity, i.e., requests are preferably directed to hubs. Consequently, preferential attachment is beneficial also for users and therefore we need to implement an approximation of this algorithm also with respect to these nodes.

The service lookup procedure we defined for normal users works as follows. Each user maintains a node cache, referred to as *lookup cache*. Whenever a user logins in EQUATOR, her EQUATOR instance connects to the Bootstrap Service and retrieves the initial m_0 nodes. The user node selects one of them randomly and downloads its overlay cache. This procedure is repeated periodically in order to guarantee both the user node to have up-to-date knowledge of existing peers and service lookups to be well distributed among the peers. In fact, simply populating the lookup cache with nodes retrieved from the Bootstrap Service would possibly result in concentrating the lookup traffic among a few peers, with possible congestions.

Whenever the user needs to start a service lookup, she picks one node from the lookup cache, selected with a probability proportional to its popularity, and sends the service request to it. If available, the contacted peer itself satisfies the service request, otherwise replies with a message containing its servant and overlay caches. Peers in the servant cache are contacted in parallel, asking them for the service, while peers in the overlay cache are used to refresh the lookup cache, so that a second round of search can be possibly performed (if no positive response is received). Notice that the algorithm implements a random walk encompassing D_l nodes, in accordance with our model.

F. Complementary issues

This section focuses on some complementary issues that are common to many service-oriented overlays, including EQUATOR. Since these problems are rather general, the solutions that can be implemented in EQUATOR are usually the same already proposed in other systems. However, EQUATOR-specific optimizations may be available in some cases. Those are briefly analyzed and constitute possible future work on the EQUATOR architecture.

First, the problem of limiting the damage caused by malicious nodes is definitely a challenge as in EQUATOR, like in many other distributed systems, peers are not under the control of a central authority. Some solutions (e.g., [19], [28], [29]) have been proposed and can be seamlessly applied in EQUATOR. For example, in [19] public key certificates are distributed among users to allow them to verify the origin and the integrity of messages, hence limiting the operation of malicious peers as they can be easily traceable. Similarly, certificates can be used in EQUATOR to authenticate advertisement messages, so that they can be considered trusted. We will further discuss the robustness of EQUATOR in the next section.

Furthermore, a service-oriented overlay often requires to enable a *proximity-aware* service selection. In EQUATOR, this can be done by using well-known techniques (e.g., Vivaldi [30]), as well as by modifying the proposed construction algorithm in order to build a locality-aware scale-free overlay (e.g., by customizing the approach proposed in [31]).

A third possible issue is the definition of mechanisms for encouraging users to enter the EQUATOR overlay and share some of their resources. Existing solutions (e.g., [32]–[34]) can be adapted to operate in the EQUATOR scenario. However, a precise definition of mechanisms and protocols to provide incentives in the EQUATOR overlay is left for future work.

V. EQUATOR SIMULATION RESULTS

This section presents some simulation results on the EQUATOR architecture. We first validate our overlay construction algorithm, which we show to result in a scale-free topology. We also show how EQUATOR is comparable to the ideal Barabási-Albert network in terms of lookup performance. We then elaborate on the system parameters, also focusing on the lookup and advertisement overhead at nodes. Finally, we investigate the behavior of our solution in different scenarios triggered by different kinds of peers.

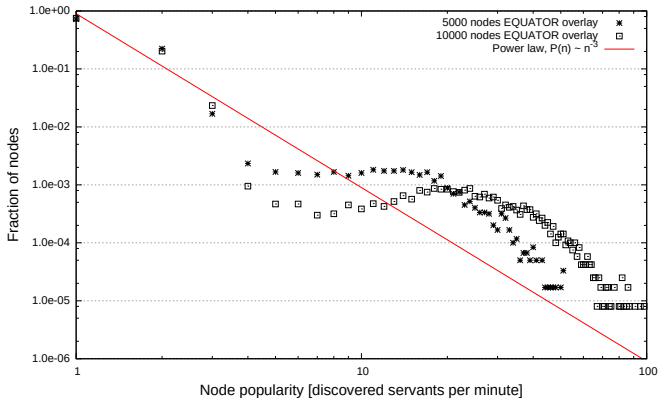


Fig. 5. Node popularity distribution.

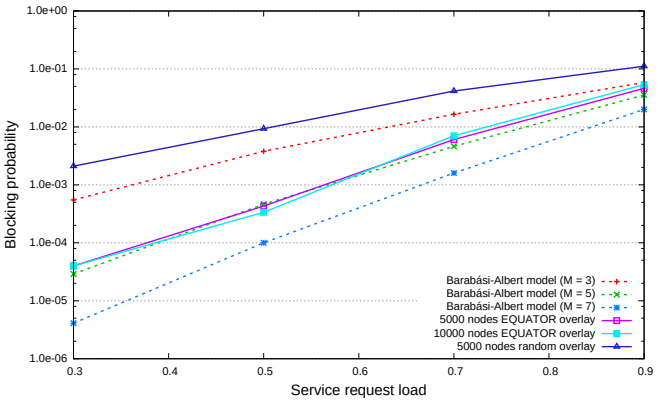


Fig. 6. 1-hop average blocking probability.

A. Simulation background

To perform our simulations, we developed a custom, event-driven simulator implementing the EQUATOR algorithms presented in the previous section. The simulator considers two types of nodes: participating peers and user nodes. The former are part of the EQUATOR overlay, while the latter represent the customers that need to exploit the offered service. Participating peer arrivals are modeled using a Poisson process, while we consider several distributions for peer lifetimes in order to investigate the behavior of EQUATOR in different scenarios. User node arrivals are modeled using a Poisson process, while user node lifetimes are assumed to be exponentially distributed. Once entered the network, user nodes run the lookup cache population algorithm presented in Section IV-E.

We model service requests with a further Poisson process. Whenever a service request is scheduled, it is randomly associated with one of the user nodes currently present in the network, which immediately starts a lookup procedure. To be compliant with the assumptions introduced in Section III-C, the service duration is exponentially distributed. We consider several service request rates, ranging from 50 to 150 requests/min. These values result in a service request load $\rho_T = 0.3 \div 0.9$.

A single Bootstrap Server is adopted for simplicity. Incoming nodes, either they are participating peers or users, contact this server and retrieve the m_0 registered peers. Different values for the overlay size N are considered, obtained by adopting a proper average peer arrival rate which, coupled with the average peer lifetime, results in an overlay of about N peers in the steady-state. Concerning the other system parameters, we set $\tau_{sc} = \tau_{oc} = 20$ nodes and $t_{adv} = 30$ min, which Section V-D will show to be proper values for the EQUATOR overlay. Moreover, we set $n_{sc} = n_{oc} = 3$ nodes, $m_0 = 20$ nodes. Finally, we set $m = 2$, $T_d = 2$, and we assume that each peer can handle only one session at a time, as explained in Section III.

B. Overlay construction

Our first simulations aim at validating our overlay construction algorithm. We assume node lifetimes to be exponentially distributed for simplicity, with an average node lifetime equal to 500 min. Different node dynamicity levels will be analyzed in the following.

Fig. 5 plots the popularity distribution of nodes, measured as the average number of different servants per minute (including the node itself) that a node can offer to querying users. Two overlay sizes $N = 5000$ and $N = 10000$ are considered to verify the scalability properties of the network[3]. The solid line represents a power law distribution $P(n) \sim n^{-3}$, i.e., the node popularity distribution in a Barabási-Albert network. The figure shows how the EQUATOR overlay assumes a scale-free topology which well approximates the Barabási-Albert network for both values of N. A certain discrepancy exists between EQUATOR and the theoretical curve for high popularity values. However, it is worth noticing how these differences are amplified by the log-log scale of the graph. Since values are related to very small portions of the entire overlay population, differences are actually of little significance. Furthermore, they are mainly due to the difficulty in collecting adequate statistics because of the low number of nodes involved.

Besides the degree distribution, it is necessary to study the clustering coefficient of the EQUATOR network in order to complete the validation of our overlay construction algorithm. In EQUATOR, the overlay is dynamic and hence links between nodes change frequently. Consequently, we evaluate this parameter as the average value among the clustering coefficients periodically observed in the network. We consider that, at a given instant of time, a node is connected to another if it sent an advertisement message to that node during the last advertisement round. Table I reports on the average clustering coefficient evaluated for different overlay sizes and compares it with the theoretical value [24] of the Barabási-Albert network. We can observe how EQUATOR reasonably approximates the Barabási-Albert model also concerning this parameter, which is slightly higher than the theoretical value, but significantly lower than the clustering coefficient of highly clustered scale-free networks, e.g., the World Wide Web, whose clustering coefficient is about 0.1 [35].

These results validate the overlay construction algorithm deployed in EQUATOR, as also confirmed by the results

[3]These values of N guarantee the significance of the obtained results and meet our memory and CPU constraints.

TABLE I
AVERAGE CLUSTERING COEFFICIENT

Network size	EQUATOR	BA model
5000	0.0084	0.0036
10000	0.0072	0.0021

TABLE II
SAMPLED CUMULATIVE DISTRIBUTION FUNCTION OF THE LOOKUP/ADVERTISEMENT MESSAGES RECEIVED BY NODES

Lookup messages (percentage of msg)	**Portion of nodes**	**Adv messages** (number of msg/min)	**Portion of nodes**
$\leq 0.01\%$	0.60	≤ 0.001	0.69
$\leq 0.1\%$	0.62	≤ 0.01	0.76
$\leq 1\%$	0.93	≤ 0.1	0.94
$\leq 10\%$	0.99	≤ 1	0.98
$\leq 100\%$	1	≤ 7	1

presented in the following.

C. Lookup performance

To validate the effectiveness of the EQUATOR overlay when providing lookup services, we consider the 1-hop average blocking probability (i.e., the probability that a user does not find an available servant when $D_l = 1$). Coherently with the assumptions of Section III-C, we consider a lookup hop to be exhausted when that node (that receives a service request) and all the servants it knows have been asked for the service.

We use as a reference the lookup performance obtained over a Barabási-Albert network where lookup procedures start only at nodes whose in-degree is greater than a given value M. We consider values for M ranging from 3 (corresponding to a percentage of nodes involved in the lookup procedures $p_{sp} = 16\%$) to 7 (corresponding to $p_{sp} = 5\%$) a good trade-off between lookup performance and lookup load distribution among nodes, as discussed in Section III-E. Fig. 6 shows how EQUATOR and this ideal network achieve comparable results. In particular, EQUATOR behaves similar to a Barabási-Albert overlay where $M = 5$ (corresponding to $p_{sp} = 8\%$).

Given the limited size of caches in EQUATOR, this result is obtained thanks to the policies adopted in advertising peers and in handling such caches. These tend to favor the selection of popular nodes, thus approximating the behavior of a Barabási-Albert network where M assumes values reasonably greater than 1. This is confirmed by the cumulative distribution of the average percentage of lookup messages per minute received by nodes when $D_l = 1$, presented in Table II for the case $N = 5000$. Although about 40% of participating peers are target of lookups from users, about 7% of nodes handle 99% of service requests, i.e., $p_{sp} \approx 7\%$, with a consequent high lookup performance.

For the sake of completeness, Fig. 6 also considers the lookup performance of EQUATOR when nodes select their neighbors randomly among peers in the overlay cache and users start lookup procedures from a node selected randomly among peers they know. These mechanisms emulate the behavior of existing hierarchical overlays (e.g., KaZaA), where super nodes are sparsely and randomly connected and ordinary nodes (the users in our case) do not implement any degree-driven selection of the super nodes to contact during searches [25]. The figure shows the better performance of EQUATOR with respect of this randomized overlay, thus confirming the effectiveness of our scale-free approach.

D. Effect of cache size and advertisement rate

In order to justify our design choice concerning the cache size and the advertisement rate adopted to derive the above results, in this section we elaborate on the effect of these parameters on the system performance. In particular, Fig. 7 plots the blocking probability achieved in EQUATOR as a function of these parameter values. To obtain the three curves, each parameter is varied separately, while the others are kept constant and equal to the abovementioned values ($\tau_{sc} = \tau_{oc} = 20$ nodes and $t_{adv} = 30$ min). A service request load $\rho_T = 0.9$ is considered to analyze a critical scenario. Furthermore, we set $N = 5000$.

In Fig. 7(a), we can observe how values of a few tens for τ_{sc} and τ_{oc} are sufficient to ensure a low blocking probability, which does not decrease significantly with a further increase of these values. In essence, a proper cache refresh, coupled with a limited cache size, allows EQUATOR to emulate an ideal system where each node has an arbitrary number of neighbors and a global knowledge of the network. To complete this analysis, Fig. 7(b) shows how an advertisement interval t_{adv} of a few tens of minutes is sufficient to ensure a good cache refresh. Lower values of t_{adv} are not necessary and do not provide a significant performance increase. This is due to the scale-free nature of the EQUATOR overlay: the shape and the short average path length it exhibits ensure a good refresh rate of the hub caches, thus leading to high lookup performance.

A higher advertisement rate may be necessary in order to use EQUATOR in different contexts, e.g., to locate specific resources. However, this is not the purpose of the system, which has been designed for locating equivalent servants. Adamic et al. [23] demonstrated the effectiveness of unstructured scale-free overlays when adopted to locate specific resources. However, this use of the scale-free topology requires different overlay maintenance, resource discovery, and lookup techniques that better support the offered service.

E. Message overheads

The above presented results prove the effectiveness of EQUATOR. In particular, they show how the scale-free topology ensures overlay efficiency with a limited advertisement rate ($t_{adv} = 30$ min), a small dissemination-depth ($T_d = 2$), and a limited cache size ($\tau_{sc} = \tau_{oc} = 20$ nodes). This results in a reduced per-node-overhead, as confirmed by Table II, which also includes the cumulative distribution of the average number of advertisement messages per minute processed at nodes when $N = 5000$. We can observe how 98% of nodes process less than 1 advertisement messages per minute and remaining 2% process always less than 7 messages per minute.

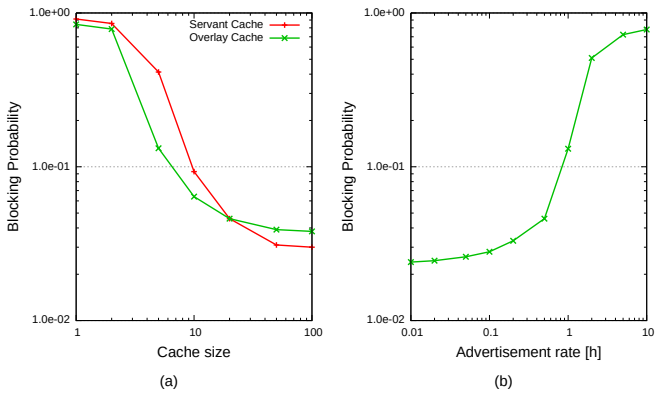


Fig. 7. Effect of parameters on the system performance: (a) cache size; (b) advertisement rate.

Concerning the lookup overhead, studied at the reasonable service request rate of 100 requests/min (i.e., $\rho_T = 0.6$) and for a network size $N = 5000$, we observed a maximum average service request rate at a single node of 5 messages/min. Furthermore, we observed a pick rate of about 20 messages/min, registered in about 1% of the total number of simulated minutes. This pick is mainly due to both the dynamics of request arrivals, which are modeled with a Poisson process. A hypothetical centralized solution would register an average request load on the central server of 100 messages/min (i.e., all requests would be directed to the server). This value is 20 times greater than the maximum average value observed in EQUATOR. Furthermore, also in this case we would register picks due to the characteristics of the request arrival process.

When the network size grows, the network maintains its scale-free topology. Consequently, the number of nodes with an adequate popularity, which are likely to be contacted during lookup procedures, increases. Hence, although on equal load conditions the number of requests at the hubs increases, this value will not increase linearly with the size. For example, we also simulated a 50000 node overlay, where we did not see the maximum average request rate per node growing linearly from 5 to 50 messages/min. We registered instead a maximum value of 30 messages/min.

F. Failure probability

So far we considered the average blocking probability as a performance metric of EQUATOR, and compared it with the results obtained over a Barabási-Albert network. However, in a dynamic scenario such as EQUATOR, users can perceive service degradation also when an available servant is found, but then suddenly leaves the network before the service ends. This problem is common to all service-oriented overlays and can be mitigated in several ways, depending on the specific service deployed. Possible solutions are the utilization of backup nodes [20], the adoption of intelligent node selection and service migration policies [36], or the creation of application checkpoints [37]. The development of novel solutions in this context is outside the scope of the paper; however, we investigate for completeness how the EQUATOR architecture performs when different node lifetime distributions are used.

TABLE III
FAILURE PROBABILITY

Number of backup nodes	Highly dynamic overlay	Moderately dynamic overlay	Quasi-static overlay
0	0.1762	0.0167	0.0012
1	0.1031	0.0066	0.0006
2	0.0543	0.0028	0.0003
3	0.0296	0.0012	0.0001

Among all possible countermeasures against unexpected node departures, we analyze the utilization of backup nodes, located during the exploitation of the service on the first servant. We assume for simplicity that a peer can be a backup node for an arbitrarily number of users.

We consider three different network scenarios, characterized by different participating peer behaviors: a *highly dynamic overlay*, exemplified as a P2P-based Voice-over-IP network, a *moderately dynamic overlay*, exemplified as a P2P-based file-sharing network, and a *quasi-static overlay*, where participating peers are quasi-static nodes such as set-top-boxes, DSL gateways, data-centers, or various kinds of servers. Concerning the first scenario, the node lifetime distribution is obtained empirically after analyzing Skype traffic coming from/to the network of the University campus [38]. Node lifetimes are instead modeled as a Weibull distribution (shape $= 0.2$, scale $= 1200$) in the moderately dynamic overlay, as resulting in [39] for a file-sharing network. The third scenario is obtained by considering node lifetime exponentially distributed with an average node lifetime of 2 months (significantly longer than the average service duration). Table III reports on the overall failure probability (defined as the probability for the service to be disrupted, due to either a lookup failure or the servant node departure during the service exploitation) achieved in EQUATOR when $\rho_T = 0.6$. An overlay size $N = 5000$ is considered for these tests. Notice that the more dynamic the overlay, the higher is the failure probability, although backup nodes improve the overall performance. These results confirm how quasi-static nodes (such as the DSL gateways of NaDa or geographically distributed data-centers) are interesting potential peers that can be used to build service-oriented overlays, and in particular EQUATOR.

In these tests we set $D_l = 4$, which allowed us to isolate the contribution of leaving servants from the overall failure probability, because the probability for a lookup to fail can be considered negligible (in fact, we did not observe lookup failures during simulations). Notice how this further confirms the effectiveness of overlay construction algorithm of EQUATOR as the system performs similarly to the Barabási-Albert network when $D_l > 1$ (see Fig. 3).

It is also interesting to investigate how node sudden and massive failures affect the overall failure probability. We defined a failure event in the EQUATOR simulator that periodically replaces a given percentage p_f of peers (selected randomly) with new ones. Fig. 8 shows the evolution of the overall failure probability over time in the quasi-static scenario when $p_f = 0$ (i.e., no cancellation occurs), $p_f = 10\%$, and

$p_f = 20\%$. A service request load $\rho_T = 0.9$ is considered and no countermeasures for node departures are adopted to analyze the worst-case scenario. Notice how replacing 10% of peers (Fig. 8(b)) has almost no effect on the EQUATOR performance, which, excluding a brief transitory which follows the replacement events, is comparable to that obtained during normal operation. The failure of 20% of the participating peers (Fig. 8(c)), although unlikely to occur, is considered for completeness. Such an event affects the system performance, which degrades with respect to the previous cases. However, we can observe how the overlay takes a reasonable time to almost completely recover from the failure and starts again to provide high lookup performance.

An even more catastrophic event for a scale-free topology is the removal of hubs, possibly due to attacks. Hence, we repeated the last experiment by replacing the 20% most popular peers (i.e., the peers that received the greatest number of advertisement messages). This is deleterious for a Barabási-Albert network, which has been proved to collapse when about 3% most connected nodes are removed [40]. Fig. 8(d) reports on the obtained results concerning the evolution of the failure probability in such a scenario. As expected, the failure probability rapidly increases when the replacement occurs because the overlay topology is damaged and, consequently, lookups fail. However, also in this case the network automatically recovers in a reasonable amount of time. This is a major advantage of EQUATOR with respect to static scale-free networks and is due to both the policy adopted to populate the overlay cache and the dynamicity of links among nodes. The presence of lowly popular peers (which are not targets of the attack) in the overlay cache allows nodes to continue the advertisement and hence avoids the complete destruction of the network. This is in line with the theoretical results presented in [41], which demonstrates that the insertion of additional links among lowly connected nodes significantly increases the robustness of scale-free networks to hub deletions. The network dynamicity ensures nodes reconstruct the topology as highest popular peers are likely to be contacted during each advertisement round, thus further gaining in popularity and hence becoming the new hubs.

These results confirm the effectiveness of EQUATOR, which couples a high lookup performance with an adequate resilience to failures and intentional attacks, even when massive node deletions occur.

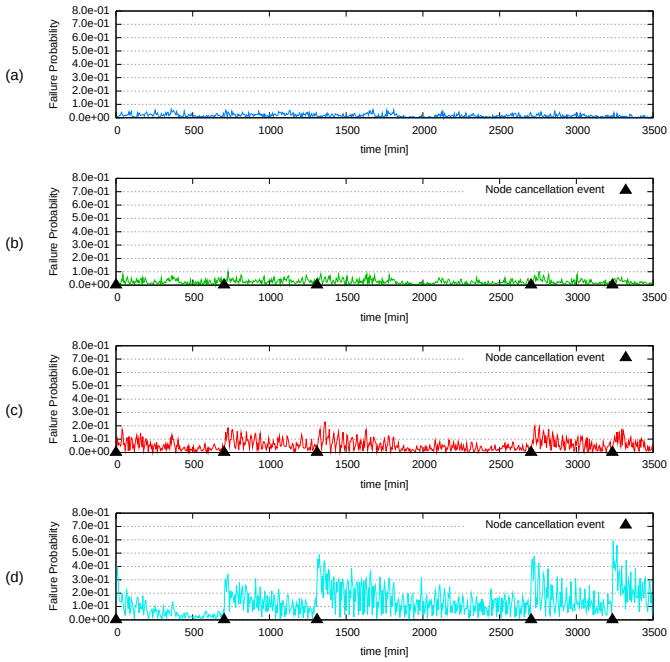


Fig. 8. Time evolution of the failure probability in a quasi-static overlay scenario for different node cancellation patterns: a) $p_f = 0$; b) $p_f = 10\%$; c) $p_f = 20\%$; d) $p_f = 20\%$ (most popular peers).

VI. Conclusion

This paper focuses on service-oriented overlays where users are interested to locate any of the many available overlay peers in the shortest time, i.e., the offered service is based on equivalent servants. Existing solutions, either structured or unstructured, can support these services but are not optimized for this purpose, which however is growing in importance due to the spread of many applications which need these specific features (e.g., a proxy node to anonymize a communication). This paper compares structured and unstructured overlays, demonstrating through analytical and simulation results how an unstructured solution relying on a scale-free topology is an effective option to deploy for offering services based on equivalent servants. On the basis of this result, we proposed the EQUivalent servAnt locaTOR (EQUATOR) architecture, which overcomes the issues related to the deployment of a scale-free topology for service location in a real network, mainly due to the static nature of the ideal scale-free construction algorithm and the lack of a global knowledge of the participating peers. Simulation results confirmed the effectiveness of EQUATOR, showing how it offers good lookup performance in conjunction with low message overhead and high resiliency to node churn and failures. Some possible future works are introduced in Section IV-F and are related to some complementary issues ranging from the proximity-aware selection of servants to the introduction of proper incentives to encourage nodes to join the EQUATOR overlay and offer their resources.

References

[1] V. Valancius, N. Laoutaris, L. Massoulie, C. Diot, and P. Rodriguez, "Greening the internet with nano data centers," in *Proc. ACM CoNEXT*, 2009.

[2] "Tor: anonymity online." [Online]. Available: http://www.torproject.org

[3] "Skype: Free internet telephony that just works." [Online]. Available: http://www.skype.com

[4] P. Bettner and M. Terrano, "1500 archers on a 28.8: Network programming in age of empires and beyond," in *Proc. Game Develop. Conf.*, 2001.

[5] S. Pichai and L. Upson, "Introducing the google chrome os," 2009. [Online]. Available: http://googleblog.blogspot.com/2009/07/introducing-google-chrome-os.html

[6] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 17–32, 2003.

[7] P. Maymounkov and D. Mazières, "Kademlia: A Peer-to-Peer information system based on the XOR metric," in *Peer-to-Peer Systems*, 2002, pp. 53–65.

[8] T. Klinberg and R. Manfredi, "Gnutella 0.6," Jun. 2002. [Online]. Available: http://groups.yahoo.com/group/the_gdf

[9] "Kazaa media desktop," 2001. [Online]. Available: http://www.kazaa.com/

[10] *A Peer-to-Peer Approach to Resource Location in Grid Environments*, vol. 0. Los Alamitos, CA, USA: IEEE Computer Society, 2002.

[11] D. Puppin, S. Moncelli, R. Baraglia, N. Tonellotto, and F. Silvestri, "A grid information service based on peer-to-peer," in *Euro-Par*, 2005, pp. 454–464.

[12] D. Tran, K. Hua, and T. Do, "A peer-to-peer architecture for media streaming," *IEEE J. Sel. Areas in Comm.*, vol. 22, no. 1, pp. 121–133, Jan. 2004.

[13] T. Do, K. Hua, and M. Tantaoui, "P2vod: providing fault tolerant video-on-demand streaming in peer-to-peer environment," in *Proc. IEEE Int. Conf. Comm.*, vol. 3, June 2004, pp. 1467–1472.

[14] *MAAN: A Multi-Attribute Addressable Network for Grid Information Services*, vol. 0, 2003.

[15] A. R. Bharambe, M. Agrawal, and S. Seshan, "Mercury: supporting scalable multi-attribute range queries," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 353–366, 2004.

[16] Y. Zhu and Y. Hu, "Ferry: A p2p-based architecture for content-based publish/subscribe services," *IEEE Trans. Par. Distrib. Systems*, vol. 18, no. 5, pp. 672–685, May 2007.

[17] J. Albrecht, D. Oppenheimer, A. Vahdat, and D. A. Patterson, "Design and implementation trade-offs for wide-area resource discovery," *ACM Trans. Int. Technol.*, vol. 8, no. 4, pp. 1–44, 2008.

[18] A. Awan, R. A. Ferreira, S. Jagannathan, and A. Grama, "Unstructured peer-to-peer networks for sharing processor cycles," *Parallel Comput.*, vol. 32, no. 2, pp. 115–135, 2006.

[19] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne, "Resource location and discovery (reload) base protocol," Internet Engineering Task Force, Internet Draft draft-ietf-p2psip-base-06, Nov. 2009, (Work in progress).

[20] L. Ciminiera, G. Marchetto, F. Risso, and L. Torrero, "Distributed connectivity service for a sip infrastructure," *IEEE Network*, vol. 22, no. 5, pp. 33–40, 2008.

[21] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," *SIGMETRICS Perform. Eval. Rev.*, vol. 30, no. 1, pp. 258–259, 2002.

[22] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Rev. Mod. Phys.*, vol. 74, pp. 47–97, Jan. 2002.

[23] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman, "Search in power-law networks," *Phys. Rev. E*, vol. 64, no. 4, pp. 046 135+, Sep 2001.

[24] K. Klemm and V. M. Eguíluz, "Growing scale-free networks with small world behavior," *Phys. Rev. E*, vol. 65, 2002, 057102.

[25] J. Liang, R. Kumar, and K. W. Ross, "The kazaa overlay: A measurement study," *Comp. Netw.*, vol. 49, no. 6, 2005.

[26] R. Cohen, K. Erez, D. B. Avraham, and S. Havlin, "Resilience of the internet to random breakdowns," *Phys. Rev. Lett.*, no. 21, pp. 4626–4628, Nov.

[27] S. Voulgaris, D. Gavidia, and M. van Steen, "Cyclon: Inexpensive membership management for unstructured p2p overlays," *J. Netw. Syst. Manag.*, vol. 13, no. 2, 2005.

[28] W. Yeager and J. Williams, "Secure peer-to-peer networking: The jxta example," *IEEE IT Profess.*, vol. 4, no. 2, pp. 53–57, 2002.

[29] E. Tamani and P. Evripidou, "Applying trust mechanisms in an agent-based p2p network of service providers and requestors," in *Proc. IEEE Int. Symp. Cluster Comp. and the Grid*, 2006, p. 13.

[30] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: a decentralized network coordinate system," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 15–26, 2004.

[31] G. P. Jesi, A. Montresor, and O. Babaoglu, "Proximity-aware superpeer overlay topologies," *IEEE Trans. Netw. Serv. Manag.*, vol. 4, no. 2, pp. 74–83, Sep. 2007.

[32] F. Hong, Y. Feng, M. Li, and Z. Guo, "Constructing incentive oriented overlay on mobile peer-to-peer networks," in *Proc. IEEE Int. Conf. Par. Process.*, 2007, p. 52.

[33] G. Tan and S. Jarvis, "A payment-based incentive and service differentiation scheme for peer-to-peer streaming broadcast," *IEEE Trans. Par. Distr. Syst.*, vol. 19, no. 7, pp. 940–953, July 2008.

[34] A. T. S. Ip, J. C. S. Lui, and J. Liu, "A revenue-rewarding scheme of providing incentive for cooperative proxy caching for media streaming systems," *ACM Trans. Multim. Comput. Commun. Appl.*, vol. 4, no. 1, pp. 1–32, 2008.

[35] L. A. Adamic, "The small world web," in *Proc. Springer-Verlag Europ. Conf. Res. and Adv. Techn. for Dig. Libr.* London, UK: Springer-Verlag, 1999, pp. 443–452.

[36] *Node selection for a fault-tolerant streaming service on a peer-to-peer network*, vol. 2. Los Alamitos, CA, USA: IEEE Computer Society, 2003.

[37] R. de Camargo, F. Kon, and R. Cerqueira, "Strategies for checkpoint storage on opportunistic grids," *IEEE Distrib. Syst. Online*, vol. 7, no. 9, pp. 1–1, Sept. 2006.

[38] D. Bonfiglio, M. Mellia, M. Meo, and D. Rossi, "Detailed analysis of skype traffic," *IEEE Trans. Multim.*, vol. 11, no. 1, pp. 117–127, 2009.

[39] V. Aggarwal, O. Akonjang, A. Feldmann, R. Tashev, and S. Mohr, "Reflecting P2P user behaviour models in a simulation environment," in *Proc. Euromicro Conf. Par., Distr. and Netw.-Based Process.*, 2008, pp. 516–523.

[40] R. Cohen, K. Erez, D. Ben-Avraham, and S. Havlin, "Breakdown of the internet under intentional attack," *Phys. Rev. Lett.*, vol. 86, pp. 3682–3685, Apr. 2001.

[41] J. Zhao and K. Xu, "Enhancing the robustness of scale-free networks," *J. Phys. A: Mathem. and Theor.*, vol. 42, no. 19, p. 195003, May.

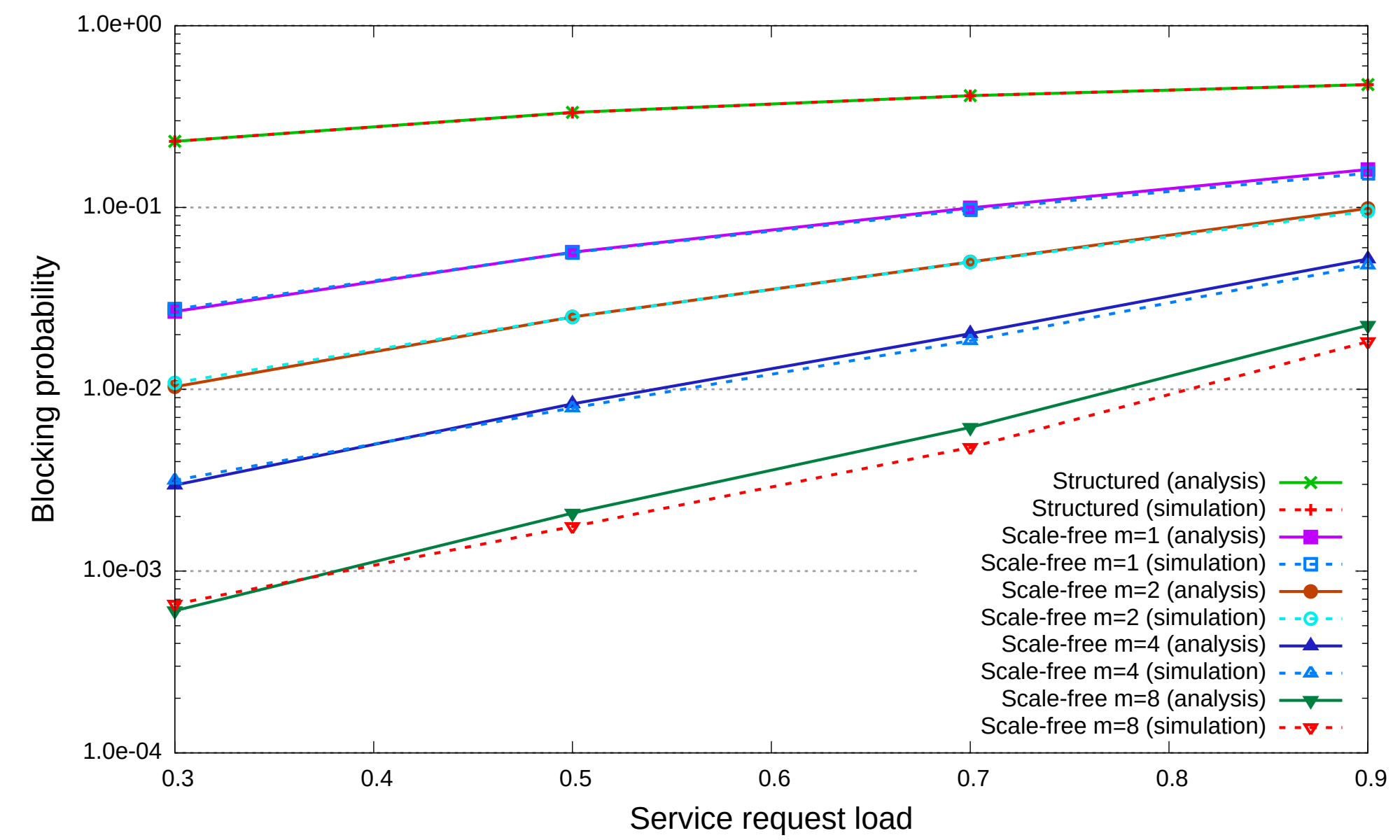
Guido Marchetto is a post-doctoral fellow at the Department of Control and Computer Engineering of Politecnico di Torino. He got his Ph.D. in Computer Engineering in April 2008 and his laurea degree in Telecommunications Engineering in April 2004, both from Politecnico di Torino. His research topics are peer-to-peer technologies, distributed services, and Voice over IP protocols. His interests include network protocols and network architectures.

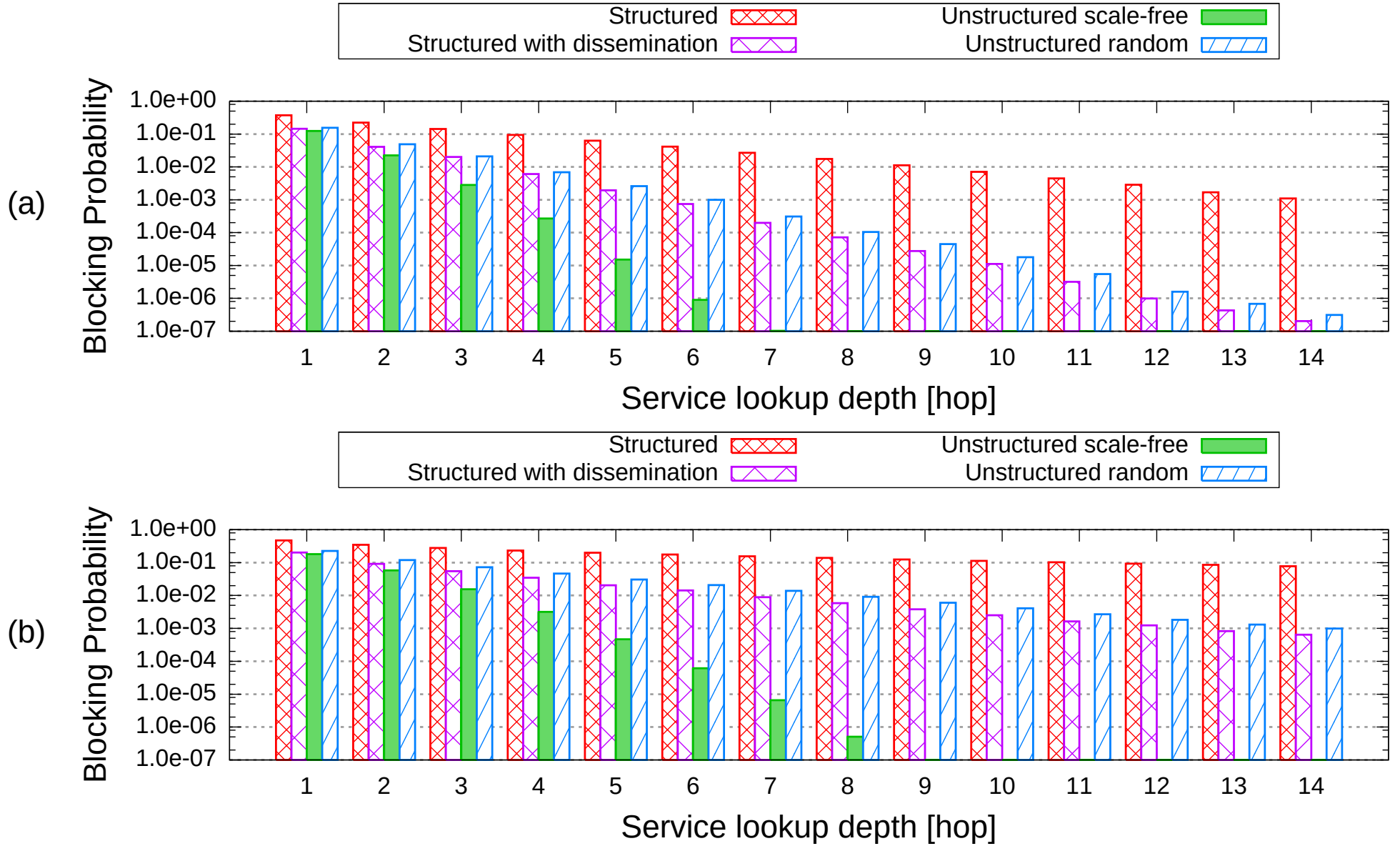
Luigi Ciminiera is professor of Computer Engineering at the Dipartimento di Automatica e Informatica of Politecnico di Torino, Italy. His research interests include grids and peer-to-peer networks, distributed software systems, and computer arithmetic. He is a coauthor of two international books and more than 100 contributions published in technical journals and conference proceedings. He is a member of the IEEE.

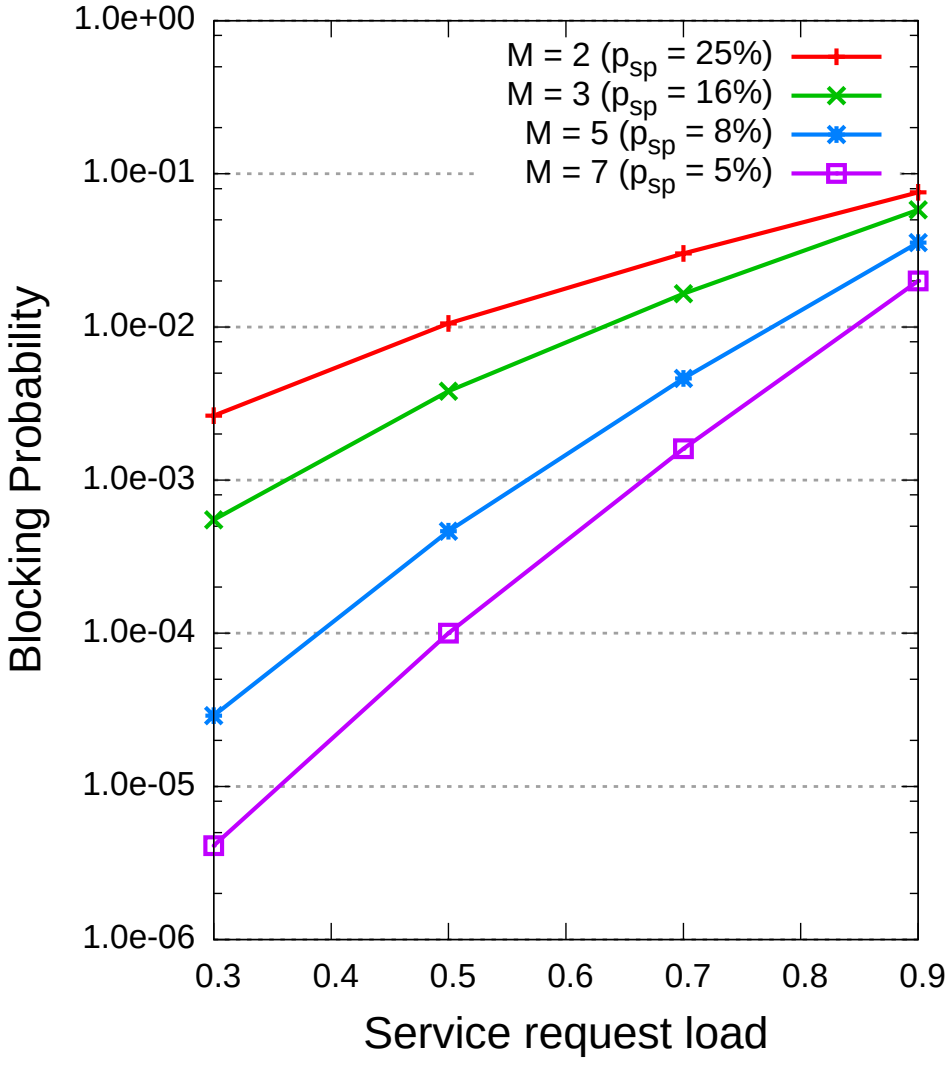
Marco Papa is a Ph.D. student in Computer and System Engineering at the Department of Control and Computer Engineering of Politecnico di Torino (Technical University of Turin), Italy. He holds a B.S. Degree and M.S. Degree both in Computer Engineering. His research interests include quality of service, privacy and peer-to-peer technologies.

Fulvio Risso is Assistant Professor at the Department of Control and Computer Engineering of Politecnico di Torino. He is author of several papers on quality of service, packet processing, network monitoring, and IPv6. Present research activity focuses on efficient packet processing, network analysis, network monitoring, and peer-to-peer overlays.

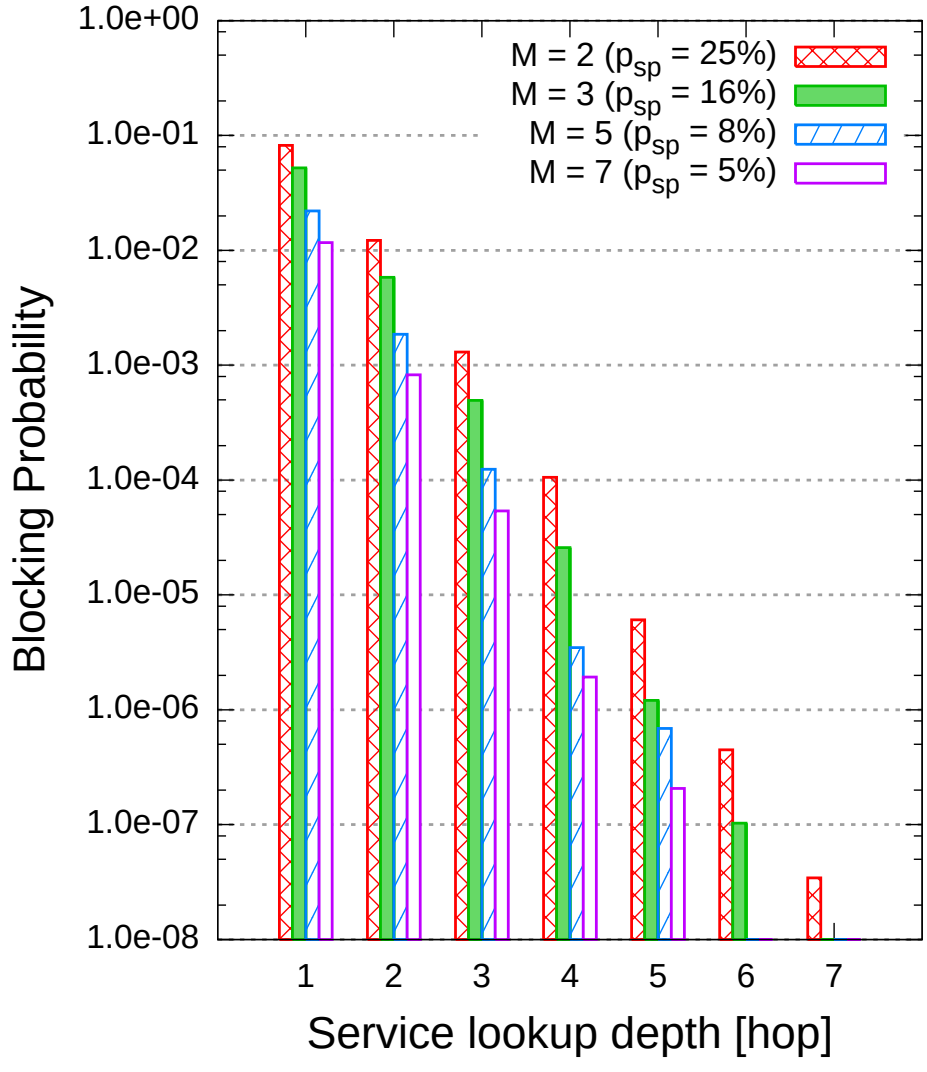
Livio Torrero is a post-doctoral fellow at the Department of Control and Computer Engineering of Politecnico di Torino. He got his Ph.D. in Computer Engineering in April 2009 and his laurea degree in Computer Engineering from Politecnico di Torino in November 2004. His research topics include the Voice over IP protocols, the IPv6 protocol, peer-to-peer technologies and their NAT/firewall related issues.


1.0e+00
1.0e-01
1.0e-02
1.0e-03
1.0e-04
Blocking probability
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Service request load
Structured (analysis)
Structured (simulation)
Scale-free m=1 (analysis)
Scale-free m=1 (simulation)
Scale-free m=2 (analysis)
Scale-free m=2 (simulation)
Scale-free m=4 (analysis)
Scale-free m=4 (simulation)
Scale-free m=8 (analysis)
Scale-free m=8 (simulation)

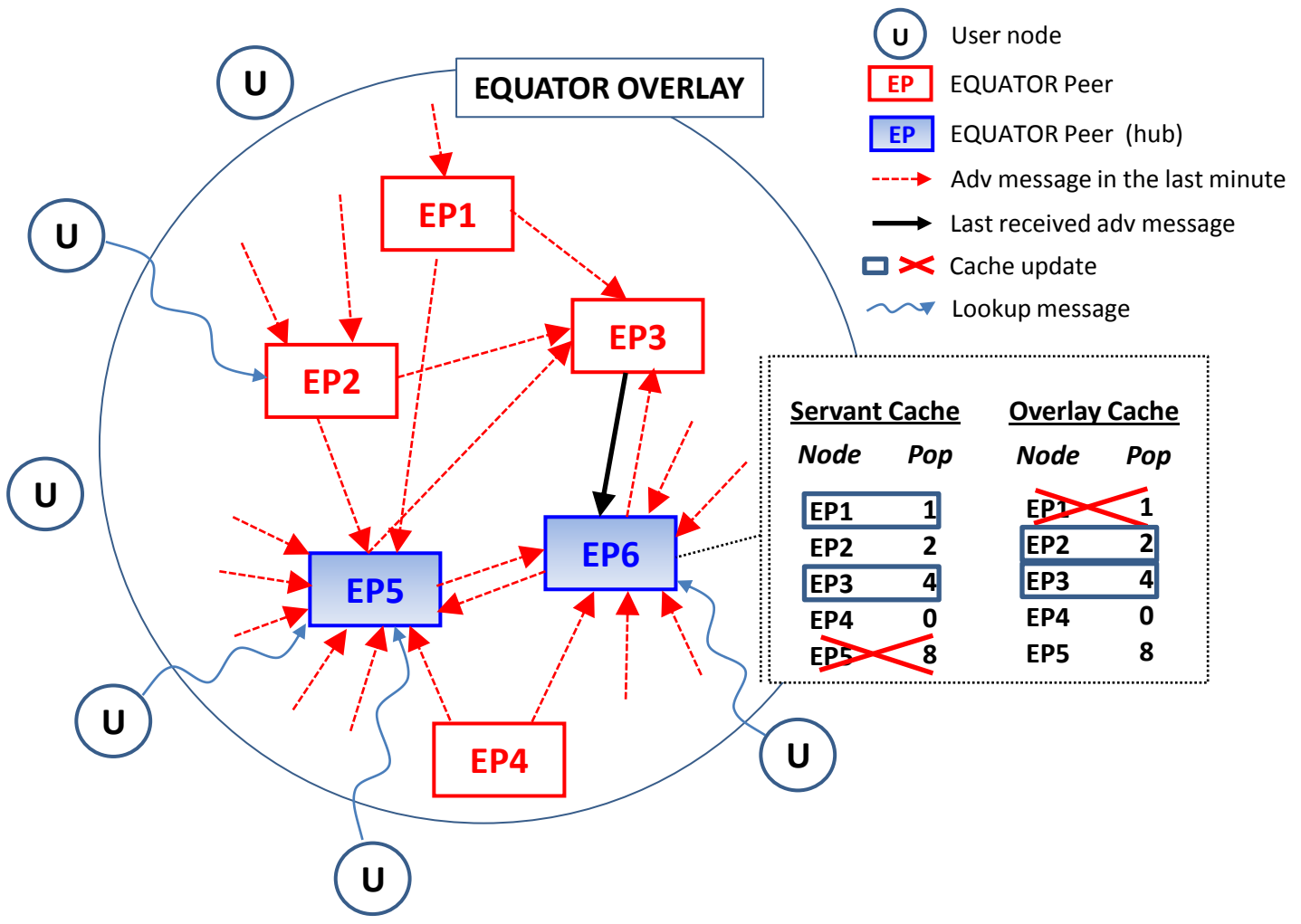

(a)
Structured
Structured with dissemination
Unstructured scale-free
Unstructured random
Blocking Probability
1.0e+00
1.0e-01
1.0e-02
1.0e-03
1.0e-04
1.0e-05
1.0e-06
1.0e-07
1 2 3 4 5 6 7 8 9 10 11 12 13 14
Service lookup depth [hop]
(b)
Structured
Structured with dissemination
Unstructured scale-free
Unstructured random
Blocking Probability
1.0e+00
1.0e-01
1.0e-02
1.0e-03
1.0e-04
1.0e-05
1.0e-06
1.0e-07
1 2 3 4 5 6 7 8 9 10 11 12 13 14
Service lookup depth [hop]

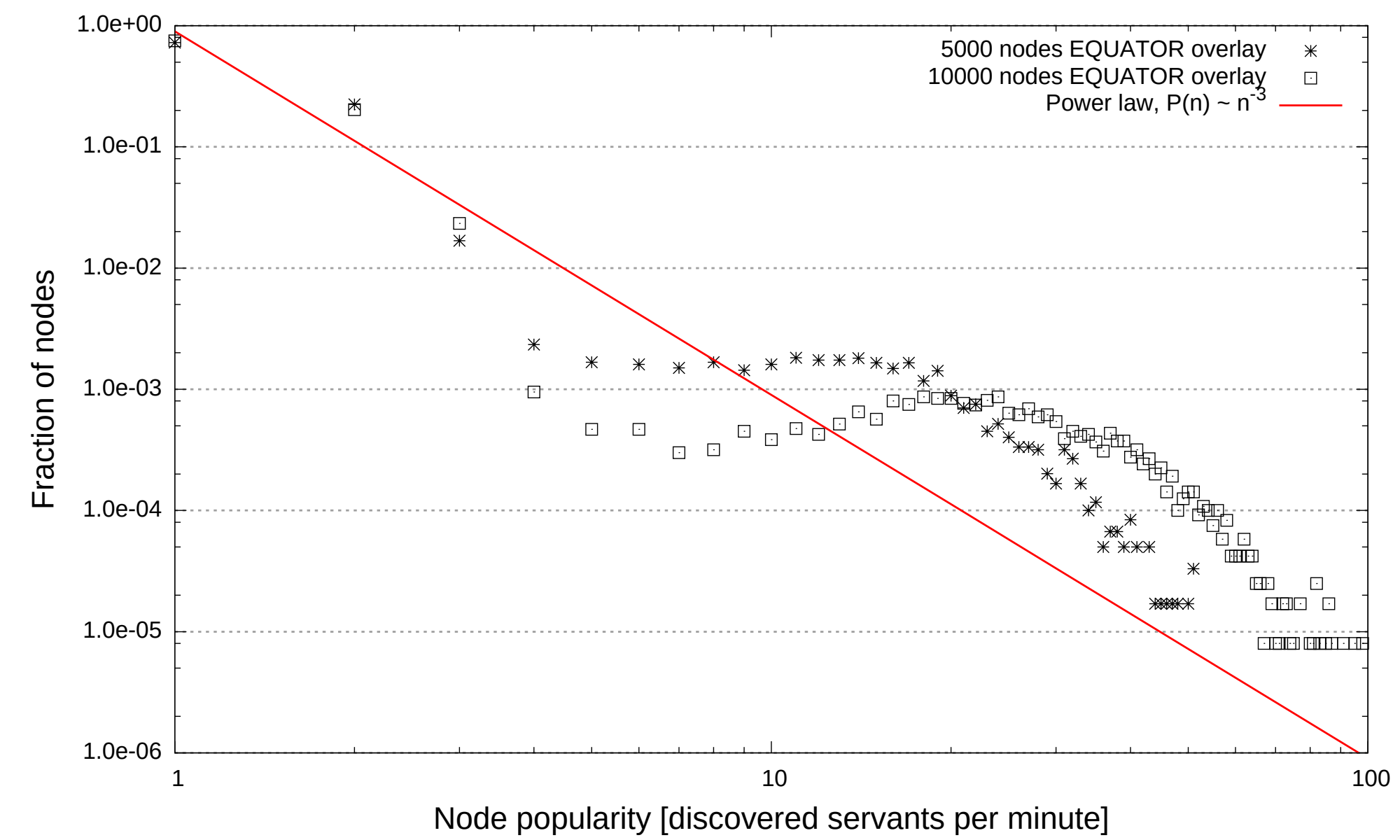

1.0e+00
1.0e-01
1.0e-02
1.0e-03
1.0e-04
1.0e-05
1.0e-06
Blocking Probability
M = 2 (p_sp = 25%)
M = 3 (p_sp = 16%)
M = 5 (p_sp = 8%)
M = 7 (p_sp = 5%)
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Service request load

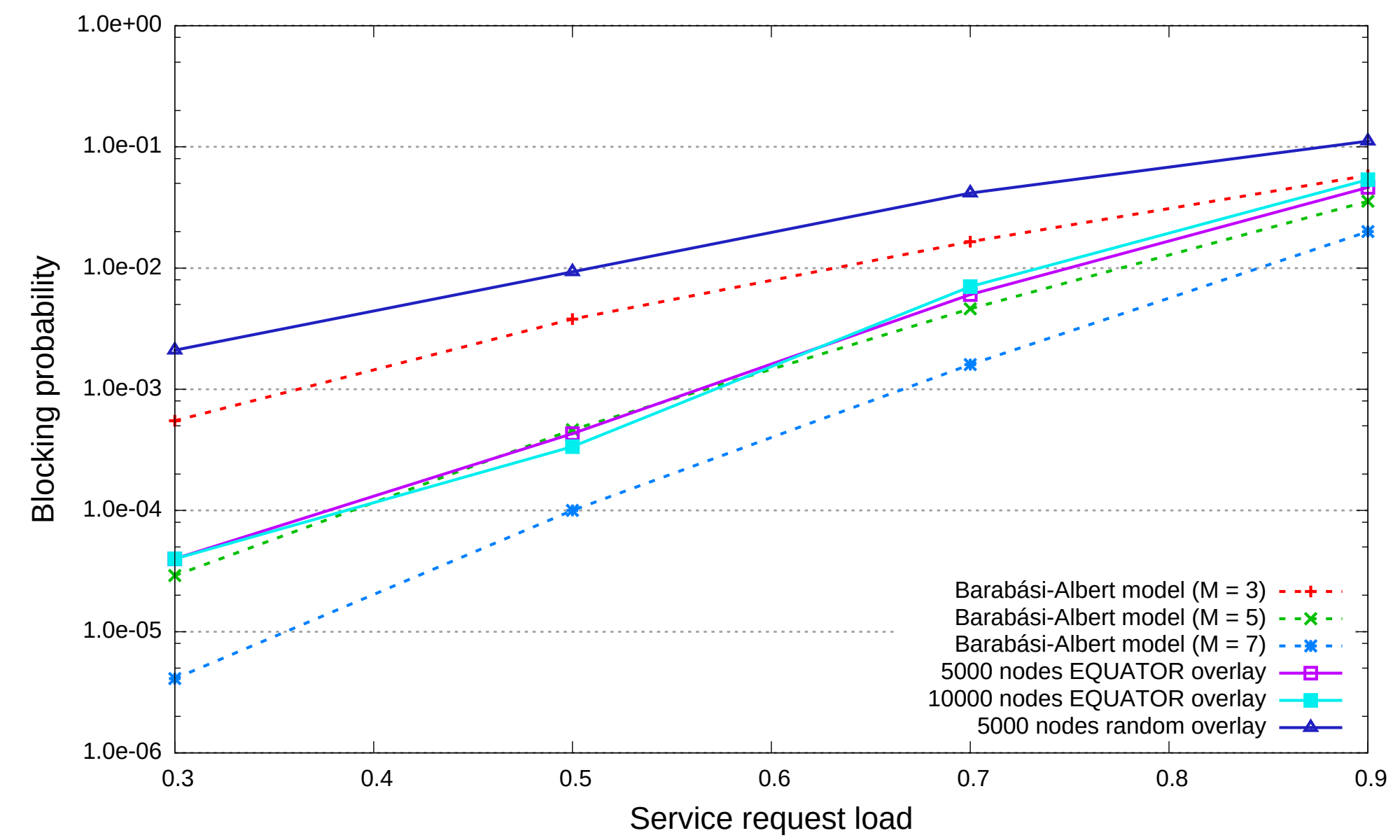
(a)

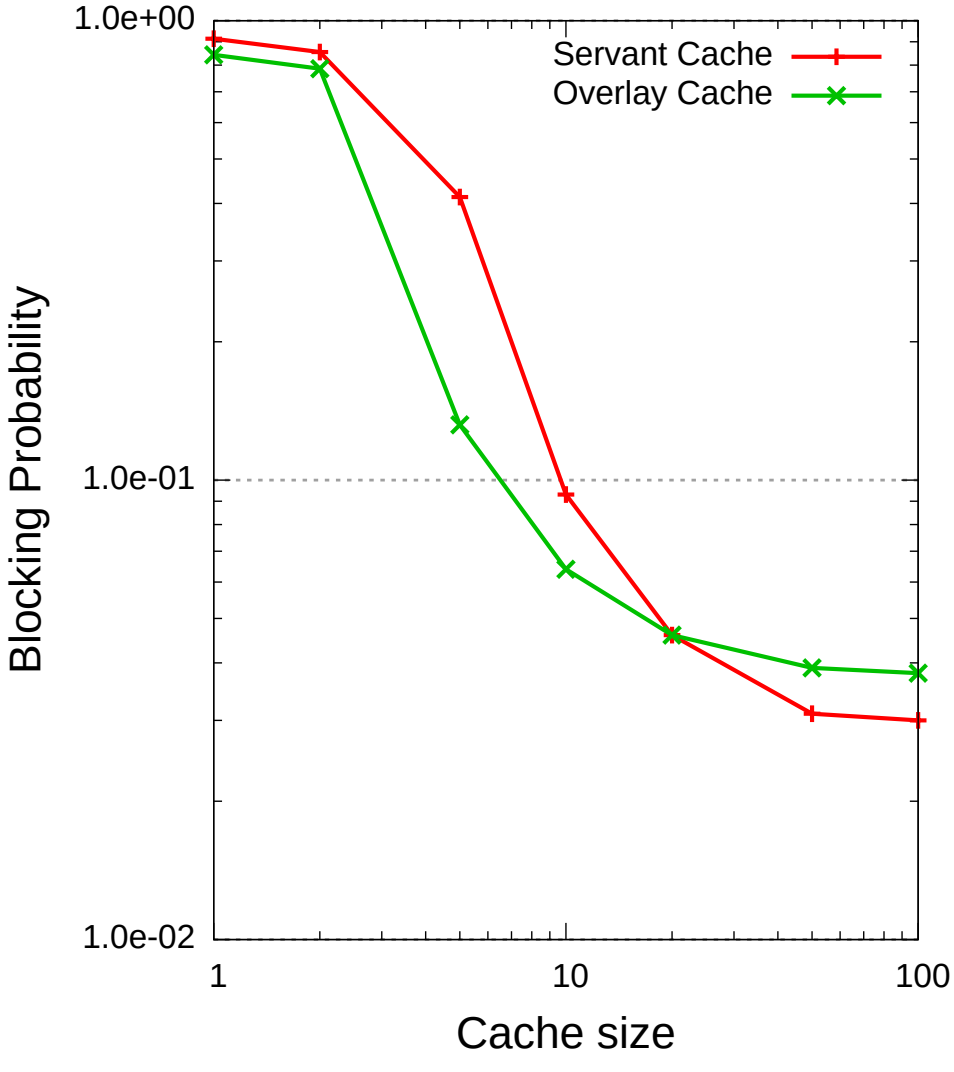

1.0e+00
1.0e-01
1.0e-02
1.0e-03
1.0e-04
1.0e-05
1.0e-06
1.0e-07
1.0e-08
Blocking Probability
M = 2 (p_sp = 25%)
M = 3 (p_sp = 16%)
M = 5 (p_sp = 8%)
M = 7 (p_sp = 5%)
1
2
3
4
5
6
7
Service lookup depth [hop]

(b)

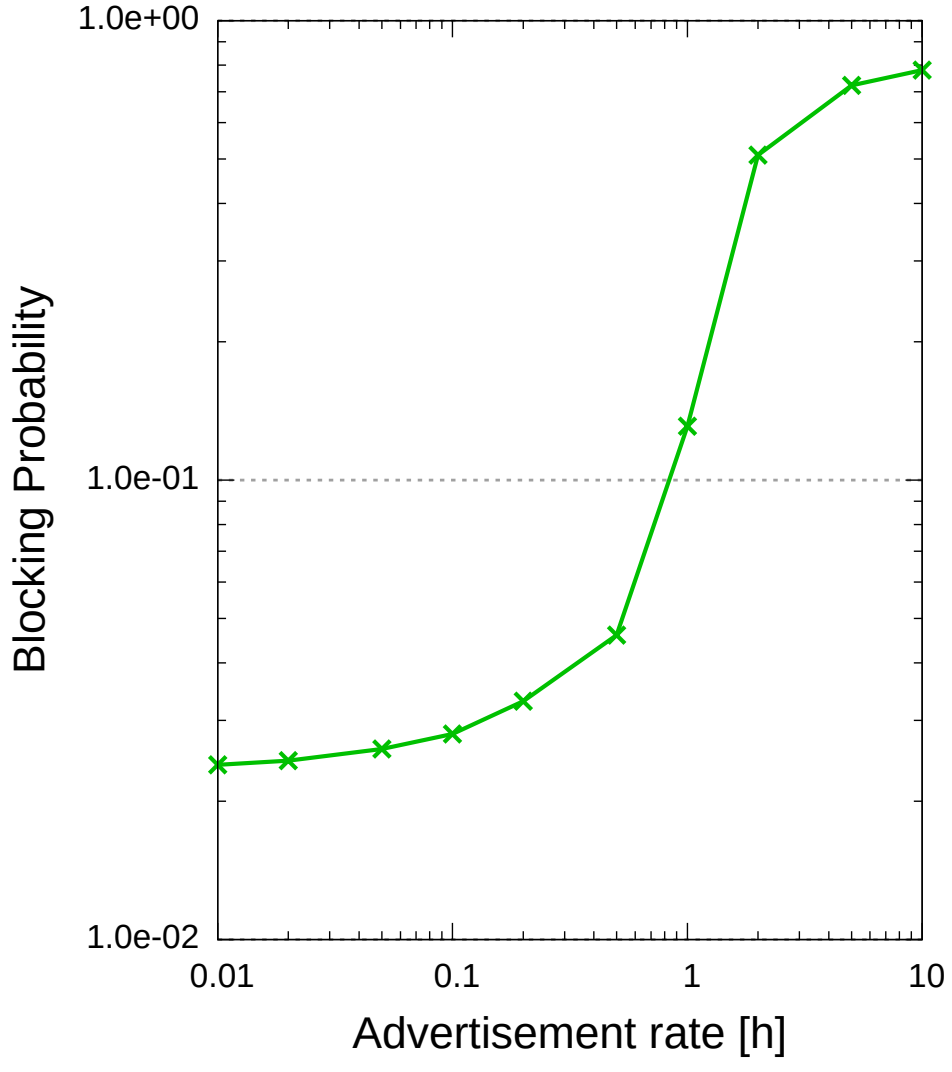

U
User node
EP
EQUATOR Peer
EP
EQUATOR Peer (hub)
Adv message in the last minute
Last received adv message
Cache update
Lookup message
EQUATOR OVERLAY
EP1
EP2
EP3
EP4
EP5
EP6
U
U
U
U
U
U
Servant Cache
Node Pop
EP1 1
EP2 2
EP3 4
EP4 0
EP5 8
Overlay Cache
Node Pop
EP1 1
EP2 2
EP3 4
EP4 0
EP5 8


5000 nodes EQUATOR overlay
10000 nodes EQUATOR overlay
Power law, P(n) ~ n^-3
Fraction of nodes
1.0e+00
1.0e-01
1.0e-02
1.0e-03
1.0e-04
1.0e-05
1.0e-06
1
10
100
Node popularity [discovered servants per minute]

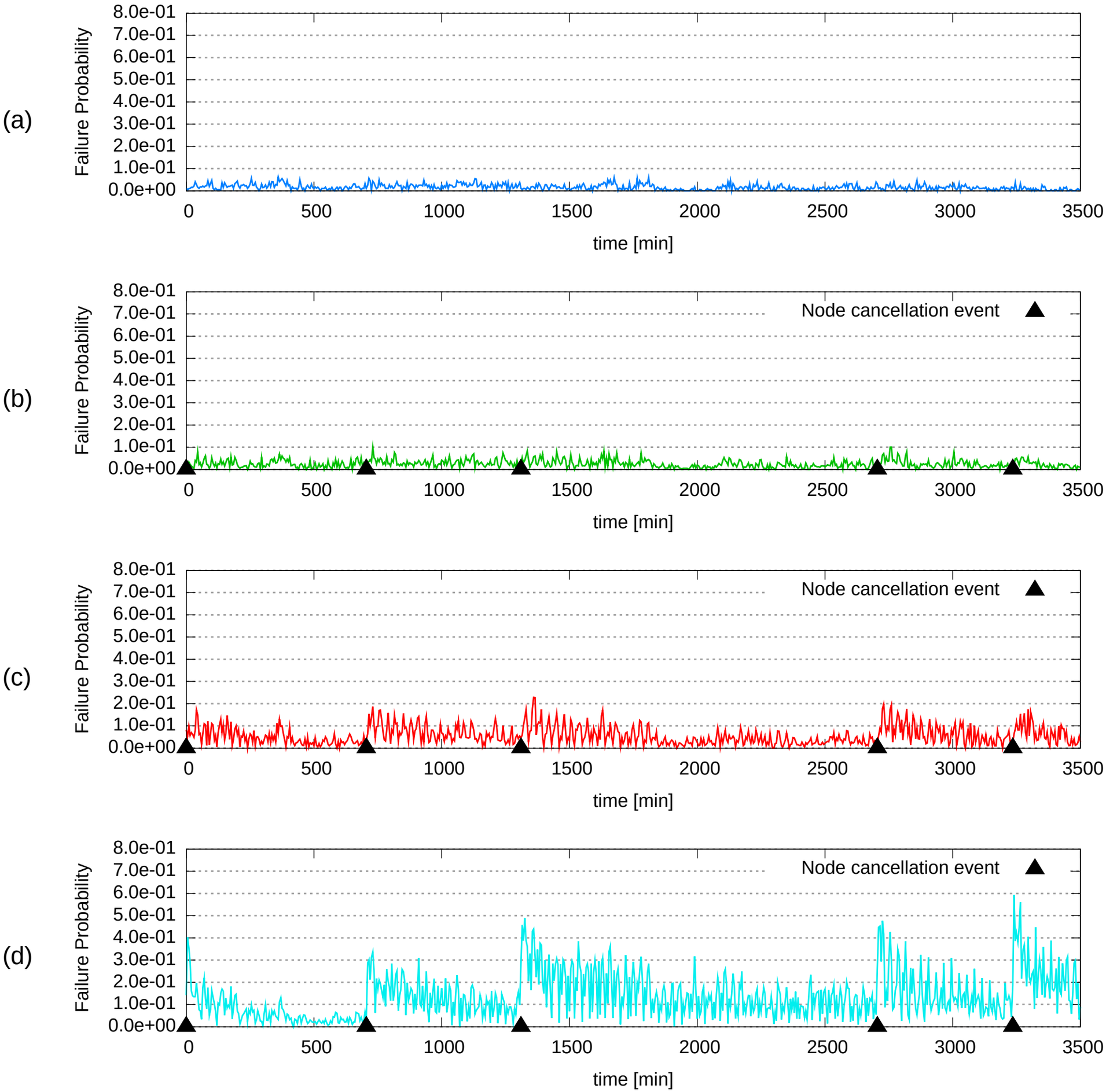

1.0e+00
1.0e-01
1.0e-02
1.0e-03
1.0e-04
1.0e-05
1.0e-06
Blocking probability
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Service request load
Barabási-Albert model (M = 3)
Barabási-Albert model (M = 5)
Barabási-Albert model (M = 7)
5000 nodes EQUATOR overlay
10000 nodes EQUATOR overlay
5000 nodes random overlay


Servant Cache
Overlay Cache
1.0e+00
1.0e-01
1.0e-02
Blocking Probability
1
10
100
Cache size

(a)


1.0e+00
1.0e-01
1.0e-02
Blocking Probability
0.01
0.1
1
10
Advertisement rate [h]

(b)


(a)
(b)
(c)
(d)
Failure Probability
8.0e-01
7.0e-01
6.0e-01
5.0e-01
4.0e-01
3.0e-01
2.0e-01
1.0e-01
0.0e+00
0
500
1000
1500
2000
2500
3000
3500
time [min]
Node cancellation event