Gossiping for resource discovering: An analysis based on complex network theory

Stefano Ferretti
Department of Computer Science, University of Bologna, Mura Anteo Zamboni 7, 40127 Bologna, Italy

ABSTRACT
This paper analyzes the adoption of unstructured P2P overlay networks to build resource discovery services. We consider a simple distributed communication protocol, which is based on gossip and on the local knowledge each node has about resources held by its neighbors. In particular, upon reception (or generation) of a novel query, a node relays the message to those neighbors that have resources whose profile matches the query. Moreover, the node gossips the query to other remaining neighbors, so that the query can be disseminated through the overlay. A mathematical analysis is provided to estimate the number of nodes receiving the query (and consequently, the portion of query hits), based on the network topology, resource availability and gossip probability. Results show that the use of unstructured networks, coupled with simple dissemination protocols, represent a viable approach to build P2P resource discovery systems.

1. Introduction
The amount of services and resources available in current computer networks has become so high that their discovery results as one of the more difficult issues to cope with. This remains true whatever the type of the considered resource might be, e.g. some data content rather than a computing facility. Different ways are possible to distribute (and locate) resources on a network. The employed system architecture, its topology (i.e. the graph resulting from the interaction map among nodes of the network) and the consequent node organization impose constraints on the topology of searching methods to discover these resources and influence their performance. In particular, resource discovery mechanisms can be built on top of either structured or unstructured architectures.

Structured networks are those where links are created based on the contents held by nodes. Examples of structured architectures are the traditional client/server, tree-based and hierarchical content-dependent structures [1], Peer-to-Peer (P2P) systems built using Distributed Hash Tables (DHTs) [2–5].

Conversely, in an unstructured P2P overlay network, links among nodes are established arbitrarily, i.e. they do not depend on the contents being disseminated through the overlay [6]. These solutions are particularly simple to build. Thus, unstructured overlays may be useful in very dynamic contexts. Peers locally manage their connections based on some general desired topology. Such a selected topology may vary depending on the characteristics the system should have. For instance, a uniform graph where nodes have all the same degree (i.e. number of connected nodes) might be useful to balance the communication load at peers. Conversely, a scale-free topology might be preferred to create a robust overlay, with a reduced network diameter (this is obtained at the cost of having some nodes subject to higher workloads) [7]. A main question here is how queries can be disseminated to locate resources effectively.

In this paper, we show that a simple gossip protocol (augmented with some additional local information about the network) can be employed to build effective resource discovery mechanisms on top of unstructured P2P overlay networks. The proposed analytical model is based on complex network theory. In particular, according to the protocol, peers maintain local knowledge about their neighbors, i.e. each node is aware of its resources and those of nodes which are directly linked to it (this requires very little management costs). The discovery process passes through the links that compose the overlay. Hence, a node generating a query can ask its neighbors only. In turn, each time a node receives a query, it relays the related message to those neighbors holding resource items matching the query (if there is any). In addition, it gossips the message to other neighbors (which do not own any matching resource), so that the query can be disseminated through the overlay.

The proposed model is able to estimate the amount of nodes receiving the query and the amount of query hits, i.e. how many resource items can be found using this simple resource discovery protocol. The approach is quite general; the network topology...
can be set by defining the node degree distribution probability. Depending on the network topology, the resource availability and the gossip probability, it is possible to understand if the query reaches a limited amount of nodes, or if it is spread through the whole network, i.e. it might reach an infinite amount of nodes. In substance, this study demonstrates that the use of unstructured networks employing gossiping dissemination strategies, based on local decision processes, guarantees that queries can percolate through the network. This outcome promotes the development of viable resource discovery mechanisms over unstructured P2P architectures.

In order to validate the effectiveness of the mathematical model, we have compared numerical outcomes with those obtained via simulation. A discrete event simulator has been built, which is able to mimic the distributed communication protocol on top of a randomly generated unstructured overlay whose topology can be specified using a given node degree probability distribution. We simulated a wide number of overlay networks, varying the network topology and degree distribution parameters, the network size, the resource availability. We also varied the parameters characterizing the communication protocol, i.e. the gossip probability. Results obtained via simulation are comparable with those coming from the analytical model.

As a final remark, it is worth pointing out that we are not suggesting here to replace completely structured and reliable distributed schemes, which are usually employed to build resource discovery approaches, with unstructured overlays using gossip. Rather, our claim is that this solution represents an interesting alternative when dealing with large scale and highly dynamic systems. In this case, in fact, the costs for managing and maintaining a structured (or centralized) distributed system is quite high. The proposed approach is particularly effective when discovery mechanisms are needed which allow to locate a set of (one or several) resource items deployed on a network, rather than all resources (in this case a structured approach might provide better results). In fact, the protocol allows to span the unstructured network without flooding it. Hence, through the described protocol a requesting node obtains a set of available resources deployed on the unstructured system. Then, depending on their characteristics, it can decide which one to ask/utilize.

The remainder of this paper is organized as follows. Section 2 discusses some background and the related work. Section 3 presents the system model. Section 4 states the local protocol executed at each node. Section 5 presents the mathematical model. Section 6 outlines results coming from a numerical analysis and simulation. Finally, Section 7 provides some concluding remarks.

2. Background and related work

Resource discovery in distributed systems can be implemented using different techniques [8]. Usually, the proposed solutions rely either on centralized architectures (e.g. [9–11]) or P2P structured-overlays (e.g. [12–15]) to store and disseminate contents. In essence, these are architectural solutions where links among nodes are created based on the contents held by nodes. These solutions are widely employed in several real use cases, but they might have significant drawbacks. For instance, according to the client/server architecture, all nodes connect to the single node that has the whole knowledge about resources present in the system, i.e. the server. This solution is really simple and can be quite effective to look for a resource, but the server, which acts as the index system, constitutes the bottleneck and the single point of failure. Other classic structured searching approaches, such as DNS, rely on tree based or hierarchical structures [1]. While more scalable, these solutions are typically subject to high costs for the management of the overlay structure.

Another usual structured approach consists on the use of Distributed Hash Tables (DHTs) in P2P systems [2–5]. In this case, the idea is to characterize resources through a specific key-word. Such information is distributed across the network, so that each node is responsible for a given key. A hashing algorithm is used to identify the node who stores the searched resource. Hence, each query based on some particular key passes through the corresponding node in the DHT. Discovery mechanisms on top of DHTs guarantee that a given key can be found in the network (if present) in a limited number of hops. However, DHT-based approaches need intensive maintenance on hash table updates. Moreover, the use of key-words for searching a resource strongly limits the expressiveness of the queries.

On the other hand, the use of unstructured overlays enables scalable and efficient solutions that obviate the need for a structure [16–19]. In an unstructured P2P overlay, links among nodes are established arbitrarily. These architectural solutions are particularly simple to build and manage, with little maintenance costs, yet at the price of a non-optimal organization of the overlay. Peers locally manage their connections to build some general desired topology and links do not depend on the contents being disseminated [6]. Unstructured overlays are quite useful when the number of nodes is very high, with very frequent topology changes and churns, i.e. high number of nodes joining and leaving the system. They are employed in several communication scenarios, ranging from classic distributed P2P systems to wireless delay tolerant and opportunistic networks.

As concerns the discovery of resources, while many mentioned P2P structured approaches (e.g. those employing DHTs) limit the expressiveness of a query by forcing nodes to search resources based on a limited number of key-words, systems built over unstructured overlays may support partial-match and complex queries. This is because resources can be characterized in many different forms. A profile can be associated to a resource that describes it. Hence, queries can be made that exploit such resource profiles. A disadvantage of such a kind of unstructured P2P solutions is that it might be more difficult to discover rare items, when compared with more popular ones. Thus, their performance depends heavily on the search mechanisms implemented on top of the communication overlay.

To disseminate queries several alternatives exist. These range from simple flooding or unicasting to the use of more sophisticated methods based, for example, on machine learning [20]. For instance, discovery in Gnutella is based on a flooding protocol; discovery requests are relayed to all neighbors until a matching resource is found or a timeout occurs. It has been recognized that such a flooding-based algorithm used in Gnutella does not scale, since each query generates a huge amount of traffic and large systems quickly become overwhelmed by the query-induced load.

When we look at other possible approaches, a main distinction is whether the search method is informed or uninformed. Informed schemes rely on the partial knowledge of the network that nodes have discovered previously; and thus, these mechanisms are based on some heuristics to effectively exploit the knowledge each node has on the network. In contrast, in uninformed schemes nodes know nothing of the surrounding network; the network must be explored until some resource is found. This can be done in a systematic way, e.g. breadth-first search, depth-first search, flooding. Alternatively, random search protocols can be exploited, e.g. random walk, gossip-based probabilistic forwarding, probabilistic flooding [21–23].

Data replication strategies, coupled with random walk search, are another option [24–26,18]. According to a random walk protocol, when a node receives a query, it relays it to a randomly chosen neighbor. A sequence of random walks can be executed if the number of results of a query is lower than a certain
threshold. With respect to flooding, such protocol strongly reduces the number of propagated messages. Furthermore, the bottleneck risk is limited for the receiver because during a random walk, the maximum number of responses that can be obtained in a short time is the depth of the random walk itself, whereas with flooding the number of received responses can be exponential [27].

According to best-neighbor-based propagation, queries are not flooded over the network, but directed to only some selected nodes, i.e. to those for which a query hit is more probable [28, 29]. A similarity forwarding algorithm is employed that caches some information on previous requests and discovery results. Information maintained on the cache expires after some deadline.

Just a few works employ gossip solely to discover contents (not general resources) [30, 17, 31]. These schemes are inspired by the theory of epidemics, since a node that gossips a content may be seen as it infects its neighbor. Weaker reliability guarantees are provided for better scalability. The interesting idea is that by employing gossip, dissemination strategies can be devised which ensure a small probability of delivering contents just to a little portion of nodes, and a very high probability of delivering contents to (almost) all destinations [32–34]. The idea underlying gossip protocols dates back to the original USENET news protocol, NNTP, developed in the early 1980’s. Seminal works that employ gossip protocols to build effective distributed systems have been presented in [32, 35]. Gossip protocols can exploit push or pull approaches. In a push protocol, each node gossips periodically to disseminate its contents. Conversely, in a pull protocol, a node solicits the transmission of information from other nodes to compensate for loss of information.

Gossip algorithms employed in an unstructured network may represent an interesting option to discover resources. As mentioned, despite the fact that these solutions are very simple and do not have information on how to route queries, they provide statistical guarantees on the number of nodes being involved in a data dissemination process. Moreover, these solutions are very resilient to network changes and churns, since they do not rely on the existence of one or more nodes. A high level of scalability is ensured; in fact, their properties are preserved as the size of the system increases. Not only, nodes have a load that depends on their degree (i.e. the higher the number of neighbors of a node, the higher the number of messages the node will probably send).

The topology of the overlay has thus a strong influence on the performances of the content dissemination, and it can be selected based on the preferred characteristics that the overlay network must guarantee. For instance, scale-free networks have a very low diameter (i.e. ranging from log log N to log N, being N the number of nodes). This means that a message may require very few hops to reach the other part of the network (if correct links are exploited). However, in these nets hubs (i.e. peers with higher degrees) will likely sustain a higher workload than the other low-degree nodes. Conversely, if a network has uniform degree distribution (where nodes have approximately the same number of neighbor nodes), the workload is equally shared among all peers. However, the diameter of the network increases, and so does the number of hops needed to cover the whole network with a broadcast [36].

The question we are dealing with in this paper is if gossip-based approaches (augmented with local information about neighbors) can be effective to discover resources in highly distributed and dynamic P2P scenarios. In particular, the communication protocol we consider is a mix of a push gossip-based scheme and a best-neighbor-based propagation, since each node relays messages to those neighbors that hit the query and gossips the message to others. The proposed theoretical framework allows to evaluate the performance of the presented communication protocol in discovering resources given a certain network topology with given statistical characteristics. Moreover, it allows to understand how the gossip probability can be tuned to make the protocol effective, given the network topology.

3. System model

We consider a P2P system built on top of an unstructured overlay network. (Note that in the following we use the terms “peer” and “node” as synonyms.) Peers are organized in a way that does not depend on the distribution of resource items in the system [36]. Rather, a pseudo-random attachment process is employed that allows to shape the overlay based on a specific network topology. This is a simple way to build a network, since no sophisticated overlay management techniques need to be exploited. Moreover, there is no central component that controls the dissemination of generated queries.

Notice that the overlay is unstructured, i.e. there is no association between nodes and the resources they hold. Hence, the system does not employ any routing tables or mechanisms employing some knowledge base storing a list of relationships between resources and nodes, as done in certain P2P systems like NeuroGrid or Freenet. Each time a node generates a novel query for a resource (identified by a profile), it disseminates a message containing it to its neighbors (the algorithm is explained in the next section). Each node receiving such a message acts as a relay and forwards the query to other (neighbor) nodes. The dissemination is based on pure local decisions: in fact, peers employ a mixed strategy that combines gossip together with a local knowledge of (profiles of) resources maintained by their neighbors.

As a final remark, the model is thought to characterize the execution of the protocol at a given instant during the evolution of the network, assuming that network changes are slower than a given execution of the communication protocol.

3.1. Overlay network

We consider a set of nodes organized as a P2P overlay network. Each node n is connected to a given subset of nodes, whose number is specified using some probability distribution. We do not impose any restriction on the overlay, which can be generated using any kind of algorithm and attachment protocol executed when peers join the network.

We denote with $p_i$ the probability that a peer n has i neighbors (the number of nodes connected to a node n is usually referred as its degree). We assume that the overlay has a high number of nodes. This assumption comes from the fact that the solution we are studying is thought for very large and highly dynamical systems. If the number of nodes is low, or in presence of a relatively stable network, probably the use of an unstructured solution might be avoided, since other approaches can be proficiently employed, such as centralized solutions or structured distributed systems [37–39, 20, 40]. The high number of nodes, together with the random nature of contacts among peers in the overlay, augments the probability of having a low clustering in the network [7].

Queries are included within messages spread through the overlay. The message contains additional information also, such as the id of the node originating the query (e.g. its IP address, with other additional application dependent information). When a query is submitted, we assume that direct communication occurs among neighbor nodes only. Hence, to disseminate information through the overlay, peers must act as relays and forward messages to their neighbors. Once a query hit occurs, i.e. the node has some resource items matching the query, the node can contact the originating node directly by sending a message.

1 We use bold fonts to identify real entities in the distributed system, e.g. host nodes or message queries; all this in order to distinguish them from mathematical elements of the model, during the discussion.
It is clear that the topology of the overlay has a strong influence on the performance of the discovery process [33]. For instance, if a scale-free network is employed, then the network has a low diameter [41]. However, a scale-free net contains a non-negligible fraction of peers which maintain a high number of active connections, and hence they sustain a workload higher than the other low-degree nodes [42–44]. Conversely, if a network has a more uniform degree distribution, then the workload is equally shared among all peers. However, the diameter of the network increases, and so does the number of hops needed to cover the whole network with a broadcast [36]. The framework employed in this work allows to assess how the topology of the overlay impacts the effectiveness of the distributed protocol by specifying the node degree probability distribution. We focus on the network coverage and on the ability of the dissemination scheme to spread a query, depending on the topology of the overlay.

3.2. Resource discovery

In this paper, we focus on the searching technique, i.e. how to distribute the query in the unstructured network. We are not interested in the query matching process, i.e. how to make proper queries and how these are matched against the existing resources. Indeed, the dissemination protocol is independent to this method.

We can model the system as a set of resources representing whatever kind of data or facility to be exploited by nodes. Each resource class is defined by a set of attributes which specify its characteristics. There can be several resource instances (items), described by one or a set of key attributes with an associated value. Queries are made to locate resources matching certain constraints. In general, we might think that each resource item has a "profile" describing it. A variety of description languages exist to describe resources, depending on the resource type. For instance, if the resource is a computing facility, a possible solution is to use the Composite Capabilities/Preference Profiles (CC/PP), or some derivation of the protocol [45]. Conversely, if the resource is some data content, then it might be described by its name combined with additional metadata associated to it. It is responsibility of the query matching process to understand if, given a query, a resource item with a given profile matches that query. In any case, the dissemination of the query does not depend on how resources are characterized and identified.

4. The protocol

Two main activities are accomplished by peers. The first one is concerned with the management of resources hold by the peer. The other one refers to the distribution of a received/generated query.

4.1. The resource management protocol

The resource management protocol is very simple (see Algorithm 1). When a peer n holds a novel resource item, it informs its neighbors (lines 3–6 in the algorithm). In particular, a message is sent specifying that a novel resource is available with a given profile.

In turn, when a node m receives a message containing a declaration that its neighbor n holds a novel resource item, m adds a related entry in its neighbor table to store the resource profile (line 14 in the algorithm). This way, each time m receives a query that hits that resource item, m forwards the query to n.

When a resource item becomes unavailable at a node, it informs its neighbors that will remove the related entry (lines 8–11 in the algorithm). In turn, upon reception at n of a control message from a node m, stating that such resource item is no longer available at m, the related entry is removed from n’s cache (line 16).

Algorithm 1 Resource management protocol executed at node n

1: \( N \leftarrow n’s\) neighbors
2: \( \text{Require: Nov} \) resource item available with profile rp
3: \( m \leftarrow \text{msg} = ("\text{available"}, rp) \)
4: \( \text{for all } m \in N \text{ do} \) (send the resource profile to all neighbors)
5: \( \text{send}(msg, m) \)
6: \( \text{end for} \)
7: \( \text{Require: Resource item with profile rp no more available} \)
8: \( m \leftarrow \text{msg} = ("\text{unavailable"}, rp) \)
9: \( \text{for all } m \in N \text{ do} \) (remove the profile)
10: \( \text{send}(msg, m) \)
11: \( \text{end for} \)
12: \( \text{require: Reception of a control message from a peer m for a resource profile rp} \)
13: \( \text{if availability of a resource item rp} \) then
14: \( \text{ADDINCACHE}(m, rp) \) (new resource at m)
15: \( \text{else} \)
16: \( \text{REMOVEFROMCACHE}(m, rp) \) (unavailable resource at m)
17: \( \text{endif} \)

4.2. The query distribution protocol

The distribution of a query is based on a push protocol [33, 46]. Algorithm 2 shows the pseudo-code of the algorithm executed at each peer n when a query must be disseminated. The used notation is summarized in Table 1. It is worth mentioning that such code describes only the distribution of the query. We implicitly assume that another software module is in charge of analyzing the query, comparing it with the profiles of resource items hold by the considered node, and those cached in the neighbor table, and in case managing the query hit.

As concerns the query distribution, once a node n makes a novel query, or upon reception of a novel query from a neighbor m, n checks if it has handled it already (we assume that each query can be univocally identified through the message id of the originating node). In such a case, n drops the query (lines 1–3). This avoids that multiple copies of the same query are processed and disseminated, thus limiting the amount of messages in the network. Then, the query is processed and if there is a query hit the sender of the query is contacted (lines 4–9). In particular, the node that has the resource contacts the node that created the query. This happens through a direct communication between the two peers.

Then, n forwards the query to its neighbors, unless the (TTL) associated to the message has reached a 0 value (lines 10–13). (In this last case, in fact, the message does not need to be forwarded elsewhere, since the maximum number of hops has been reached.) In particular, n forwards the query to the subset of neighbors holding some resource items whose profile matches the query (lines 14–18). Then, n considers the remaining set of its neighbors, i.e. those nodes that do not hold resource items matching the query. For each node in this subset, n gossips the message with a probability \( \gamma \leq 1 \) (lines 19–23).

An important aspect is concerned with the (TTL) value, employed to avoid that messages are forwarded forever in the net. In particular, such (TTL) must be sufficiently large to guarantee that the message can be spread through the whole network. Finding the “optimal” value for the TTL is a very important issue. In fact, several works employ very different values to set such value. For instance, the works presented in [33, 46–49] employ in their simulations values of the TTL ranging from 6 up to 100. Flooding protocols usually set the value of the TTL equal to 7, as done in Gnutella. This is because, by flooding a message, the number of reached peers increases almost exponentially, and in many networks (e.g. scale-free ones and small worlds) such value is widely sufficient to cover the whole network [50, 51].
decreaseTTL

Table 1

<table>
<thead>
<tr>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \gamma ) := gossip probability</td>
</tr>
<tr>
<td>( f_i ) := probability that a node forwards a query to ( i ) neighbors</td>
</tr>
<tr>
<td>( T_i ) := probability that following a link, a node is reached that forwards a query to ( i ) neighbors</td>
</tr>
<tr>
<td>( F ) := generating function of ( f_i )</td>
</tr>
<tr>
<td>( \overline{F} ) := generating function of ( T_i )</td>
</tr>
<tr>
<td>( p_i ) := probability that a peer has degree equal to ( i )</td>
</tr>
<tr>
<td>( q ) := excess degree probability, i.e. probability that following a link a node is reached which has ( i ) links other than the considered one</td>
</tr>
<tr>
<td>( (r) ) := average number of nodes that receive a query</td>
</tr>
<tr>
<td>( r_i ) := probability that ( i ) peers receive a query, starting from a given node</td>
</tr>
<tr>
<td>( \overline{r} ) := probability that ( i ) peers receive a query, starting from a given link</td>
</tr>
<tr>
<td>( R ) := generating function of ( r_i )</td>
</tr>
<tr>
<td>( \sigma ) := probability that a node has a resource item matching the considered query</td>
</tr>
<tr>
<td>( (s) ) := average number of owners of a matching resource item that receive the query during the search process</td>
</tr>
</tbody>
</table>

Algorithm 2 Query distribution protocol executed at node \( n \)

Require: Query \( Q \) generated at \( n \) \( \lor \) \( Q \) received in a message relayed by a neighbor peer \( m \)

1: if \( Q \) already handled then
2: return |
3: end if |
4: if \( \text{QUERY_HIT}(Q) \) then
5: \( s \) := \( \text{SENDER}(Q) \) |
6: \( rf \) := \( \text{PROFILE_MATCHINGRESOURCE}(Q) \) |
7: \( \text{msg} \) := \(" available", \( rp \) \) |
8: \( \text{SEND}(\text{msg}, s) \) |
9: end if |
10: \( \text{DECREASE_TTL}(Q) \) |
11: if \( \text{TTL}(Q) = 0 \) then
12: return |
13: end if |
14: \( N \leftarrow n \) ’s neighbors \( \setminus m \) \( (m = \text{NULL} \text{ if } Q \text{ originated at } n) \) |
15: \( I \leftarrow \{ i | i \in N \land i \text{ has a resource item matching } e \} \) |
16: for all \( i \in I \) do
17: \( \text{SEND}(Q, i) \) |
18: end for |
19: for all \( i \in N \setminus I \) do \((\text{gossip to the remaining neighbors})\) |
20: if \( \text{RANDOM}() < \gamma \) then
21: \( \text{SEND}(Q, i) \) |
22: end if |
23: end for |

In essence, the TTL might be set based on an estimation of the network diameter (i.e., the largest value among the shortest paths between any two nodes in the network). This estimation can be obtained from the degree probability distribution, and in most kinds of nets it is usually a low number. However, answering the question if such a value is sufficient to spread contents on the network (and in general to identify the optimal TTL value), is however not trivial. In fact, when we consider the diameter of a network, we are focusing on the shortest paths among nodes, and in general this does not mean that the average path length among nodes scales with the diameter value. This happens in several complex networks. For instance, it is recognized that given two Web pages, they are just a few clicks away if the shortest path is traced (small world phenomenon). However, very long alternative paths exist between two Web pages; and in general, it might be difficult to select the shortest path only with local information.

Moreover, in the matter of this issue, a related aspect is that the proposed protocol imposes that each node handles a given message only once (nodes cache the ids of handled queries); hence, queries do not make cycles. Furthermore, if the network has a low clustering, then locally it appears as if it has a tree-like structure. Therefore, each time a node selects a neighbor to gossip, then it likely chooses to follow a path that cannot be taken otherwise.

All this (i.e., the possibility of having long paths and the avoidance of redundant transmissions for a query) might suggest that a high value of the TTL should be preferred, since a too low value of the TTL might cause that the query ends along a path, without being spread through the network. On the other hand, if a too high value of the TTL is selected, it might happen that while a query is traveling through the network, a node, that has handled it already, overwrites/cancels the information on that query it has on its cache; and thus, if that node receives that query once again, it reprocesses the query.

In substance, the value of the TTL should be tuned based on several factors, such as the network topology, the cache size of nodes and, probably most important, the gossip probability. In fact, the higher the gossip probability (i.e. the protocol resembles a flooding) the lower the TTL needed to spread a message through the network.

5. System analysis

In this section, we analyze the performance of the decentralized P2P protocol presented in the previous section. We specifically focus on the coverage of the overlay. This allows to estimate the amount of query hits of a resource search \((s)\). We denote with \( \sigma \) the probability that a node has a resource item, i.e. \( \sigma \) represents the portion of nodes in the overlay having some resources matching the needs of the peer making the request.

We model each single query dissemination as a standalone activity. In other words, the model treats the distribution of generated queries as independent tasks. This is a correct assumption if peers have a buffer cache whose size is sufficiently large to handle simultaneous queries passing through it. Otherwise, the model should be extended to consider possible buffer overflows.

Of course, upon a query hit, the two peers can directly communicate to exchange/provide use of the resource. In particular, the node that has the resource contacts the node that created the query. This happens through a direct communication between the two peers, that does not influence the rest of the network. For this reason, there is no need to consider such part of the protocol in the model.

We consider networks with a large number of nodes. Following the approach presented in [41,7], we assume that links among nodes are randomly generated, based on a given node degree distribution [52]. This does not represent a problem, since the overlays we are considering here are synthetic communication networks, which can be built using whatever algorithm chosen during the network design phase. A consequence of the random nature of the attachment process is that, regardless of the node degree distribution, the probability that one of the second neighbors (i.e. nodes at two hops from the considered node) is also a first neighbor of the same node, goes as \( N^{-1} \), being \( N \) the number of nodes in the overlay. Hence, this situation can be ignored since the number of nodes is high.
5.1. Degree and excess degree distributions

We denote with \( p_i \) the probability that a peer \( n \) has degree equal to \( i \). Starting from \( n \), another measure of interest is the number of connections (links) that a node \( m \), which is a neighbor of \( n \), may provide, other than the one that connects \( m \) with \( n \). In particular, the probability that, following a link in the overlay, we arrive to a peer \( m \) that has other \( i \) links (hence its total degree is \( i + 1 \)) is

\[
q_i = \frac{(i + 1)p_{i + 1}}{\sum_j ip_j}.
\]

The probability \( q_i \) is often referred as the excess degree distribution [41]. Probabilities \( p_i \) and \( q_i \) represent two similar concepts i.e. the number of contacts of a considered peer (its degree), and the number of contacts obtained following a link (its excess degree), respectively. In the following, we introduce measures obtained by considering the degree \( p_i \) of a node, and considering the excess degree \( q_i \) of a link. In this last case, with a slight abuse of notation we denote all the probabilities/functions related to the excess degree with the same letter used for the degree, with an arrow on top of it, just to recall that the quantity refers to a link.

5.2. Probability of dissemination

Given a peer \( n \) in charge of relaying a query, the probability that \( n \) forwards it to \( i \) of its neighbors is

\[
f_i = \frac{1}{2} \left( \frac{\Gamma - (1 - \sigma)(1 - \gamma)}{\Gamma' + (1 - \sigma)(1 - \gamma)} \right) \frac{(j + 1)p_{i + 1}}{\sum_j ip_j} \gamma^j (1 - \Gamma)^{j-i},
\]

which is obtained by considering all the possible cases of \( n \), having a degree higher than \( i \), which forwards the query to \( i \) neighbors either because there is a query hit (with probability \( \sigma \)), either because there is no query hit but \( n \) decides to gossip the query to the node nevertheless (with probability \( (1 - \sigma)(1 - \gamma) \)). Moreover, \( n \) does not gossip the query to its remaining \( N - i \) neighbors, that would not generate a query hit (with probability \( (1 - \sigma)(1 - \gamma) \)). In the rest of the discussion, for the sake of a more readable presentation, we denote \( \Gamma = \sigma + (1 - \sigma)(1 - \gamma) \) and \( \Gamma' = (1 - \sigma)(1 - \gamma) \).

A similar reasoning can be made to measure the probability that, following a link we arrive to a node that forwards the query to \( i \) other nodes. This probability is readily obtained by substituting, in (1), \( p_j \) with \( q_i \), i.e.

\[
\overline{f}_i = \Gamma \sum_{j \geq i} q_j \left( \frac{(i + 1)p_{i + 1}}{\sum_{k \geq i} kp_k} \right) \gamma^j (1 - \Gamma)^{j-i}.
\]

To proceed with the reasoning, we need to introduce the generating functions for \( f_i \), \( \overline{f}_i \), as well as for \( p_i \), \( q_i \), i.e.

\[
G(x) = \sum_i p_i x^i, \quad \overline{G}(x) = \sum_i q_i x^i, \quad F(x) = \sum_i f_i x^i, \quad \overline{F}(x) = \sum_i \overline{f}_i x^i.
\]

If we consider the \( F \) generating function, we have

\[
F(x) = \sum_i f_i x^i = \sum_i \Gamma^i x^i \sum_j \frac{(j + 1)p_{j + 1}}{\sum_{k \geq 0} kp_k} \gamma^j (1 - \Gamma)^{j-i} = \sum_j \frac{(j + 1)p_{j + 1}}{\sum_{k \geq 0} kp_k} \Gamma^j x^j (1 - \Gamma)^{j-i} = \sum_j \frac{(j + 1)p_{j + 1}}{\sum_{k \geq 0} kp_k} \Gamma^j x^j (1 - \Gamma)^{j-i} = \frac{G(\Gamma x + 1 - \Gamma)}{\Gamma'},
\]

where \( \Gamma' = (1 - \Gamma) \).

One might notice that all the coefficients of the introduced generating functions are probabilities. In fact, \( G(1) = \sum_i p_i = 1 \), as well as \( F(1) = \sum_i f_i = 1 \), and so on. Now, it is also possible to evaluate the average of the values \( f_i \), by calculating the derivative of \( F \) measured at \( x = 1 \), since \( F'(1) = \sum_i f_i \gamma^{-1} \).

\[
F'(x)|_{x=1} = \frac{dG}{dx} \left( \Gamma x + 1 - \Gamma \right) \bigg|_{x=1} = \Gamma G'(1) = \Gamma (p).
\]

\[
5.3. Number of receivers and query hits

We can now consider the whole number of nodes reached by a message starting from a given node, regardless of the number of hops. Let denote with \( r_i \) the probability that \( i \) peers receive a query, starting from a given node. Similarly, denote with \( \overline{r}_i \), the probability that \( i \) peers are reached by the query dissemination, starting from a link. In general, \( \overline{r}_i \) can be defined using the following recurrence,

\[
\overline{r}_0 = 0, \quad \overline{r}_{i+1} = \sum_{j=0}^i \sum_{a_1 + a_2 + \cdots + a_j = i} \overline{r}_{a_1} \overline{r}_{a_2} \cdots \overline{r}_{a_j}.
\]

Eq. (9) can be explained as follows. It measures the probability that following a link we disseminate the query to \( i + 1 \) peers. (The case \( \overline{r}_0 \) is impossible, since at the end of a link there must be a node.) In general, one peer is that reached at the end of the link itself. Then, we consider the probability that the peer has other \( j \) links (varying the value of \( j \)). Each link \( k \) allows to disseminate the query to \( a_k \) peers, and the sum of all these reached peers equals to \( i \).

Similarly, we can calculate \( r_i \) as follows

\[
r_0 = 0, \quad r_{i+1} = \sum_{j=0}^i \sum_{a_1 + a_2 + \cdots + a_j = i} \overline{r}_{a_1} \overline{r}_{a_2} \cdots \overline{r}_{a_j}.
\]

In this case, we start from the peer itself, considering it has a degree equal to \( j \); and as before, from its \( j \) links we can reach \( i \) other peers, globally.

The use of generating functions may be of help to handle these two equations [53]. In fact, if we consider the generating functions for \( r_i \) and \( \overline{r}_i \),

\[
R(x) = \sum_i r_i x^i, \quad \overline{R}(x) = \sum_i \overline{r}_i x^i.
\]
then, after some manipulation typical for generating functions (e.g. [7]) we arrive to the following result

\[ \overline{R}(x) = x \sum_{i=0}^{\infty} f_i \overline{R}(x)^i = x \overline{F}(\overline{R}(x)) \]  

(12)

and, similarly,

\[ R(x) = x \sum_{i=0}^{\infty} f_i \overline{R}(x)^i = x \overline{F}(\overline{R}(x)). \]  

(13)

From the generating functions, we might recover the elements \( r_i \) composing them. Unfortunately, Eqs. (12) and (13) may be difficult to solve, depending on the degree probability distribution \( p \) which controls the whole introduced measures \([7]\).

But actually, we are not interested that much in the single values of \( r_i \), \( \overline{r}_i \), In fact, it is easier and more useful to measure the average number \( \langle r \rangle \) of peers that receive a given query through the dissemination protocol. To this aim, we can employ the typical formula for generating functions \( \langle r \rangle = \overline{R}(1) \) \([53]\). In fact, taking the first equation of (11), differentiating and evaluating the result for \( x = 1 \), and since \( r_0 = 0 \), we have

\[ \overline{R}(x)|_{x=1} = \sum_i i r_i, \]

which is the mean value related to the distribution of \( r_i \) coefficients. We already observed that the coefficients of the introduced generating functions are probabilities, and thus \( F(1) = \sum_i f_i = 1 \), and similarly \( \overline{F}(1) = 1 \), \( R(1) = 1 \), \( \overline{R}(1) = 1 \). Hence, taking (13) and differentiating

\[ \langle r \rangle = \overline{R}'(1) = \left[ \overline{F}(\overline{R}(x)) + x \overline{F}'(\overline{R}(x)) \overline{R}'(x) \right]_{x=1} \]

\[ = 1 + \overline{F}'(1) \overline{R}(1). \]  

(14)

Similarly, from (12),

\[ \overline{R}'(1) = \left[ \overline{F}(\overline{R}(x)) + x \overline{F}'(\overline{R}(x)) \overline{R}'(x) \right]_{x=1} \]

\[ = 1 + \overline{F}'(1) \overline{R}'(1). \]  

(15)

Thus,

\[ \overline{R}'(1) = \frac{1}{1 - \overline{F}(1)}. \]  

(16)

This last equation allows to find the final formula for \( \langle r \rangle \).

\[ \langle r \rangle = 1 + \frac{F(1)}{1 - \overline{F}(1)} \]

\[ = 1 + \frac{\Gamma(p)^2}{(1 + \Gamma(p)) - \Gamma(p^2)}. \]  

(17)

5.4. Discussion on percolation

As it is quite typical in complex network theory, it is actually easier to examine infinite networks rather than just large ones. The analysis of infinite networks, under conditions similar to those of large scale networks, allows to understand important peculiarities of the real networks and on protocols executed by their nodes. For instance, it is possible to understand if a message can percolate through the network. This assumption is perfectly reasonable in our scenario, since we consider very large dynamical systems (with a number of nodes that tends to infinity) where peers know only their neighbors and manage contents based on local knowledge about their neighbors.

Eq. (17) has a divergence when \( (1 + \Gamma(p)) \neq \Gamma(p^2) \), which signifies that the query reaches an infinite number of nodes, i.e. the query percolates through the network. By looking at the parameters, this situation depends, first, on the nodes’ connectivity, i.e. the node degree probability distribution \( p \). In fact, the degree probability distribution determines if the overlay has a giant component (i.e. the largest subset of connected nodes which scales with the network size, and thus has a number of nodes whose limit tends to \( \infty \)), rather than being partitioned into a set of components of limited size \([7]\). The query can be spread to a large (infinite) number of nodes only when there is such a giant component; otherwise, i.e. when the network is partitioned into a high number of components of limited size, the query can be sent to a limited number of nodes only. Studies exist that allow to understand how to build networks with a giant component \([7,54]\).

Second, the value of \( \sigma \) has influence on both the number of nodes to be reached holding some resource item matching the query and on the dissemination of queries. In fact, the higher \( \sigma \) the higher the probability that a node has some of its neighbors which generate a query hit; these nodes will be receivers of the query and subsequently they will act as relays for such query.

Third and final, the gossip probability \( \gamma \) determines if the message query is spread through the network even when the amount of nodes holding a resource item in the overlay for a given query is small, i.e. when \( \sigma \) has a very low value. Of course, setting \( \gamma = 1 \) allows to flood the query to the whole component (from which the query has been originated). This might represent a fair choice when the network has a tree-like structure, or when the network has a very low clustering. Conversely, a low value for \( \gamma \) should be employed when there are loops in the overlay.

A completely different scenario is concerned with the situation when the network is formed by limited clusters only (there is no giant component). In such a case, in fact, the number of reached nodes does not grow proportionally with the network size, and a finite number of nodes might receive a query.

6. Experimental results

This section presents an assessment performed to validate the model discussed in the previous section and evaluate whether effective resource discovery mechanisms can be built on top of an unstructured P2P system. The evaluation is performed by considering the analytical model and results obtained through a simulation of the distributed protocol. The two approaches provide similar outcomes. In particular, when the theoretical model estimates that an infinite amount of nodes is reached through the dissemination, simulations show that a significant portion of the simulated network receives the queries, as expected.

For the reasons explained above, the focus here is on network coverage. In fact, since we employ a probabilistic model, from \( \langle r \rangle \) we can recover the amount of query hits, represented by a given percentage of the receivers.
Another important metric to consider is the number of messages sent in the network. In this sense, the protocol ensures that peers disseminate a given query at most once. Moreover, the tree-like structure of the overlay limits that multiple copies of the same query are received by a peer.

We consider two kinds of networks, i.e. random graphs and scale-free networks. According to a random graph, peers have the same probability to attach to other links. In substance, when the overlay is generated, a link between each pair of peers is created with a certain probability $p$. Hence, based on this model the average degree is $\langle k \rangle = pN$, being $N$ the network size. It is well known that when the number of peers $N$ is large, nodes’ degrees may be well characterized using a Poisson distribution $\exp(-\langle k \rangle)$. Several works employ this construction tool for generating random graphs [7].

On the other hand, scale-free networks gained a lot of interest in recent years. These networks are characterized by a degree distribution following a power law $\sim p^\gamma$. They are characterized by the presence of hubs, i.e. nodes with degrees higher than the average, that have an important impact on the connectivity of the net. The interest on scale-free networks in this work relates to the fact that several P2P systems are indeed scale-free networks [43,41].

6.1. Theoretical model

We employed the framework presented in Section 5 to assess the performance of the dissemination protocol, based on the overlay network topology, i.e. node degree distribution, the probability $\sigma$ that a node has a resource item matching the query, and the gossip probability $\gamma$. Fig. 1 shows the number of nodes receiving a query when the unstructured overlay has a topology based on a Poisson node degree distribution with mean value $\lambda = 5$ (we tested the framework with other $\lambda$ values, obtaining similar results). Lines in the chart correspond to the whole number of receivers (i.e. relay nodes and query hits), while points correspond to the number of query hits. Results are obtained varying the value of $\sigma$ (on the x-axis), i.e. the portion of nodes in the overlay that have resource items that would correspond to a query hit.

From these two figures it is easy to see that, for each specific $\gamma$ value, there is a phase transition, i.e. as $\sigma$ is varied there is an abrupt increment on the number of receivers (and query hits), passing from a limited value to $\infty$, i.e. the query percolates through the network. This phase transition depends on the parameters used to set the distributed system. In fact, the value of $\sigma$ not only represents the resource available probability, but it influences the query distribution in the overlay also (a node forwards with probability 1 the query to each of its neighbors that have matching resource items). Finally, the value of $\gamma$ does not change the trend of the curves; basically, the higher $\gamma$ the smaller the value of $\sigma$ to have a transition.

Similar considerations can be made for Fig. 2, where the estimated amount of receivers and query hits is reported for a scale-free network with a degree distribution $\sim p^\gamma$, with $\lambda = -3.3$. Also in this case, each curve corresponds to a specific $\gamma$ value, while varying $\sigma$. The chart shows that for each curve there is a phase transition, where the number of receiving nodes passes from a limited (low) value to an infinite number.

As mentioned, we are employing a probabilistic model and the interest is on the trend of the results, based on a given network topology and on the behavior of the protocol. Results show that, given a number of receivers of a given query, the amount of query hits is proportional to this value.

6.2. Finding one resource, at least

In many P2P resource discovery systems, nodes look for a single resource to retrieve/acquire/exploit. In other words, in certain situations we are not interested in finding as many resources as possible; rather, a single query hit is sufficient. It is clear that the lower the resource availability $\sigma$ the more difficult is to find a given resource and thus the higher the number of nodes to ask in order to retrieve such a resource. It is worth noticing that, as shown in the previous results, above the phase transition the number of query hits diverges. Hence, it is sufficient to employ a simple gossip communication protocol to find a single resource.

In any case, given a network topology and a specific setting for the $\sigma$ parameter, the level of message dissemination on the network is controlled by the gossip probability $\gamma$. In this section, we show the minimum value of $\gamma$ to guarantee that at least one resource is found, given a certain overlay topology and a resource availability. Fig. 3 shows such a value for different settings of $\sigma$, when the network topology is a Poisson degree distribution with mean $\lambda = 5$. Outcomes demonstrate that it is sufficient to employ very low values of the gossip probability to find a resource, even when the resource availability is low. This confirms the effectiveness of the proposed approach.

As concerns scale-free networks, we recall that with higher values of $\lambda$ the moments of the degree distribution diverge, i.e. when $\gamma > -2$ the mean diverges, when $-2 < \lambda < -3$, the network has a giant component and the mean is finite but the variance and higher moments diverge [55]. Hence, in these cases the query easily percolates through the network and resources are
found with high probability. For this reason, we put the focus on overlays with a lower value of $\lambda$, with respect to those cited here above. Fig. 4 shows the minimum value of $\gamma$ when the overlay topology is a scale-free network with an exponent $\lambda = -3.3$. Based on this considered value of $\lambda$, it is possible to see that when we are dealing with rare resource items, a high gossip probability $\gamma$ is needed to ensure that a resource is found. However, as $\sigma$ augments, the needed value of $\gamma$ decreases.

### 6.3. Simulation

In order to assess the theoretical model proposed in the paper, we have built a discrete-event simulator mimicking the presented protocol. The simulator was written in C code. Pseudo-random number generation was performed by employing the GNU Scientific Library, a library that provides implementation of several mathematical routines for numerical and statistical analysis [56]. The simulator allows to test the behavior of a given amount of nodes employing the protocol explained in Section 4.

The simulator generates a random network based on a chosen degree distribution. In particular, once having (randomly) assigned a specific target degree to each node, using the selected degree distribution, a random mapping is made so that links are created until each node has reached its own target degree. During the initialization phase, for each node a random choice was made to distribute resources; the resource availability was set based on a probability $\sigma$, i.e. for each network node, the resource was present with probability $\sigma$.

We varied the network topology, the number of nodes and statistical parameters characterizing the network degree distribution. For each network setting, we repeated the simulation using a corpus of 20 different randomly generated networks. For each network, we analyzed the dissemination of 400 queries sent by random nodes. In the results that follow, for each generated network we show the average number of receiving nodes, i.e. query hits and relays; this number allows to understand if the distributed protocol is able to disseminate the query through the unstructured network, using the presented protocol.

#### 6.3.1. Poisson degree distribution

Here, we show results for networks generated through a Poisson degree distribution. Fig. 5 shows results obtained from simulation and the theoretical model. We simulated different corpuses of networks, varying the number of nodes and the value of the gossip probability $\gamma$. Each point in the chart corresponds to the average number of receivers for a simulated network. The line corresponds to the theoretical value measured using Eq. (17). It is possible to observe that all results from the simulations lie near the theoretical value, regardless on the considered number of simulated network nodes. Hence, the model is able to capture the behavior of the distributed protocol.

Figs. 6 and 7 show results obtained in our simulations when $\gamma = 0.1$ (resp. $\sigma = 0.1$), while varying $\sigma$ (resp. $\gamma$) above the phase transition. According to the model, the system is above the phase transition. Hence, assuming an infinite number of nodes in the network, an infinite number of nodes receive the message query, and due to the $\sigma$ setting, an infinite number of query hits is obtained. As concerns simulations, instead, we expect that a non-negligible portion of nodes is reached during the dissemination of a query. Of course, since the dissemination is based on rather low values of $\gamma$, $\sigma$ probabilities, and since the network clustering of these considered networks is quite low (a random attachment process was employed to build links in the network [41,7]), it is unlikely that all network nodes receive the query being disseminated. In fact, because of the tree-like structure of the network, every time a link is not employed to disseminate the message, it is likely that some branch (and consequently some sub-graph) of the overlay is cut away. Indeed, results confirm our outlook. A non-negligible portion of nodes is reached in each configuration (with respect to the network size). Yet, the whole overlay is not covered completely. The amount of the reached nodes increases with the varied parameter $\sigma$ (resp. $\gamma$).
Fig. 6. Model vs. simulation: Poisson degree distribution, $\lambda = 5$, varying $\sigma$ above the phase transition. The chart reports the number of receiving nodes through simulation. The theoretical model returns an infinite number of nodes (being the modeled overlay an infinite graph), not shown here.

Fig. 7. Model vs. simulation: Poisson degree distribution, $\lambda = 5$, varying $\gamma$ above the phase transition. Number of receiving nodes obtained through simulation (the model returns an infinite sub-graph).

Similar results were obtained for different networks built varying the statistical parameters of the random graph (not shown here). All this confirms that the protocol is able to spread a given query in the network in random graphs with Poisson degree distributions.

6.3.2. Scale-free networks

To build scale-free networks, our simulator implements a construction method which has been proposed in [57]. The interesting aspect of this algorithm is that it differs from other proposals, which build networks with a power law distribution by continuously adding novel nodes and edges, hence having networks that grow in time [42]. Conversely, the method in [57] builds a network of fixed size, characterized by two parameters $a$, $b$. More specifically, the number of nodes $y$ which have a degree $x$ satisfies $\log y = a - b \log x$, i.e. $y = \lceil e^{y_x} \rceil$. Thus, the total number of nodes of the generated network is

$$N = \sum_{x=1}^{\lfloor e^{\frac{y-0.5}{x}} \rfloor},$$

being $\lfloor e^{\frac{y-0.5}{x}} \rfloor$ the maximum possible degree of the network, since it must be that $0 \leq \log y = a - b \log x$. Once the number of nodes and their degrees have been determined, edges are randomly created among nodes until nodes reach their desired degrees.

Fig. 8 shows some examples of networks built with our simulator, implementing the construction method proposed in [57]. In particular, the chart reports, for three different settings of $a$, $b$, the number of nodes which have a given degree, in a log–log scale. It is possible to appreciate how such distributions are almost linear in a log–log scale, hence confirming they all follow a power law function.

As made above for random graphs, Figs. 9 and 10 show results obtained in our simulations when we employ a scale-free network topology, with $\gamma = 0.1$ (resp. $\sigma = 0.1$), while varying $\sigma$ (resp. $\gamma$), above the phase transition. Again, based on the model an infinite number of receivers is reached (assuming a network of infinite size). From the simulations, a non-negligible portion of nodes is reached during the dissemination of queries, that increases together with the $\gamma$ (resp. $\sigma$) parameter. Indeed, it is interesting to observe that when $\gamma = 0.6$, $\sigma = 0.1$ almost all network peers receive the query during the dissemination, and thus, almost all owners of some matching resource item receive the queries. In the scenarios reported in the pictures, in fact, we employed scale-free networks generated through $a = 6$, $b = 1$, resulting in networks composed of 2482 nodes. In this case, simulation results provide average results above 2200 nodes. A similar behavior is obtained
Fig. 10. Model vs. simulation: scale-free network, $a = 6, b = 1$, varying $\sigma$ above the phase transition. Number of receiving nodes obtained through simulation (the model returns an infinite sub-graph).

Fig. 11. Model vs. simulation: scale-free network, $a = 6, b = 1.1$, varying $\gamma$ above the phase transition. Number of receiving nodes obtained through simulation (the model returns an infinite sub-graph).

when $\sigma = 0.6, \gamma = 0.1$. Again, this result is in accordance with the outcomes from the model, stating that an infinite number of nodes is reached with these settings.

Figs. 11–20 show similar results for different networks settings. A significant portion of network nodes is reached, whose size increases together with the $\gamma, \sigma$ values. Again, all this confirms that the theoretical model is able to predict that a given query, distributed in an unstructured P2P system, can percolate through the whole overlay.

7. Conclusions

This paper analyzed the performance of an unstructured P2P overlay network that exploits a very simple gossip-based dissemination strategy to perform a P2P resource discovery. The mathematical analysis has been performed by resorting to the complex network theory. Results show that by tuning the gossip probability, it is possible to spread queries through the overlay, without the need to resorting to sophisticated dissemination strategies built on top of costly structured distributed systems. This is true when networks are large in size and the number of owners of some resource items matching the query is not negligible. Of course, the use of more sophisticated approaches, such as adaptive dissemination schemes built on top of gossip-based strategies [33], can be useful as well, yet at the cost of additional interactions and communication among nodes.

In this work we focused on the network coverage. As concerns the communication overhead, it is evident that the use of more
costly solutions (in terms of network management and node workload), such as centralized approaches or structured overlays, would provide better performances. In any case, the protocol considered in this work limits the amount of messages sent in the network, since each node relays a given query only once. Hence, no duplicate transmissions occur on a link. Moreover, the
low clustering guarantees that tree-like overlays are obtained; this limits the possibility that a peer receives multiple messages containing the same query. This is accomplished without the need (and the costs) of maintaining a structured overlay. Indeed, the proposed approach represents an interesting alternative to deterministic algorithms executed on top of structured overlays when dealing with large scale and highly dynamic systems.

References

Stefano Ferretti is an assistant professor at the Department of Computer Science, University of Bologna. He received the Laurea degree (with honors) and the Ph.D. in Computer Science from the University of Bologna respectively in 2001 and 2005. He is the chair of the Workshop on Distributed Simulation & Online gaming (DISIO). His current research interests include distributed and multimedia systems, synchronization algorithms, complex and computer networks, wireless networks, distributed simulation.