Bootstrapping P2P Overlays in MANETs

Afzal Mawji and Hossam Hassanein
Telecommunications Research Lab, School of Computing
Queen's University
Kingston, Ontario, K7L 3N6, Canada
{mawji, hossam}@cs.queensu.ca

***Abstract*—Peer–to–peer networks are very popular but the problem of bootstrapping them has largely been ignored. In a fully decentralized environment such as a mobile ad hoc network (MANET) the usual bootstrapping solutions, which typically require a centralized service, are not possible. We present a method of bootstrapping P2P overlay networks running on MANETs which involves multicasting P2P overlay join queries and responses, and caching results at all nodes. Node choose which overlay members to join to based on a utility function that considers both the distance in hops and the overlay neighbors' available energy. Simulation results show that the P2P overlay can closely reflect the underlying topology, which reduces energy consumption, that caching the join requests reduces the number of messages required to join the overlay, and that compared to Random Address Probing, there is less overhead and significantly less delay.**

I. Introduction

Peer–to–peer (P2P) networks are increasingly popular and their uses vary over many different application types, such as file sharing, Voice–over–IP, gaming and instant messaging. It seems natural to combine mobile ad hoc networks (MANETs) and peer–to–peer networks together so that a P2P overlay runs on a cooperative MANET since both types of networks have many similarities. Both are fully decentralized networks, both must dynamically organize themselves, both must deal with frequent topology changes, both attempt to be resilient to failure, and both perform the routing function.

Despite the similarities, it is unclear that simply adopting existing P2P overlay techniques and using them in MANETs is desirable, since there also differences. P2P overlays tend to be very large–scale with millions of users and are designed for deployment on the "edge" of the Internet, where the nodes generally do not move about. On the other hand, MANETs tend to have far fewer nodes, the devices are severely resource–constrained in comparison, and the links between nodes usually have higher delay. Energy consumption is of great concern and users are also geographically nearby one another.

This paper presents Overture, which is to the best of our knowledge the first to address the problem of bootstrapping P2P overlays in MANETs. The word overture has an introductory connotation, just as the bootstrap procedure is the introductory part of overlay membership. Overture uses the idea of multicasting to all nodes, which is impractical on the Internet but not in MANETs. When a node wishes to join a P2P overlay, it multicasts a network–wide join request. When a join request is received, if an overlay member has open slots for direct connections to more peers, it will multicast a network–wide response. All nodes, whether or not they belong to an overlay, will cache this response. After a sufficient time, the original requesting node will have one or more responses to select from. The requesting node evaluates a utility function for each response which reflects its preference for choosing closer overlay members or ones with more energy available. A closer member would typically result in greater responsiveness and also more closely match the underlying topology of the network, while a member with greater energy is more likely to remain alive longer.

In this paper, the term node refers to a device in the MANET that is not a part of any P2P overlay. This includes devices that are in the process of joining an overlay. The terms peer and overlay member are used interchangeably and refer to a device in the MANET that is part of a P2P overlay. A peer is a neighbor if there is a direct connection to it in an overlay. A join request represents an attempt to become part of a P2P overlay, while a connection request is a message to a specific peer in an attempt to become its neighbor.

Simulation results show that the overlay can closely reflect the underlying topology of the network as demonstrated by the average neighbor's hop distance, and that the delay to join an overlay is significantly lower than RAP, even when RAP has the advantage of selecting from only valid MANET addresses. In addition, the use of caching reduces the number of messages that must be transmitted while trying to join an overlay.

The rest of the paper is organized as follows. Section II describes the bootstrap algorithm Overture, including the multicast query and multicast response, the use of caching, and how the utility function is used to select neighbors. Section III presents our simulation results and analysis. Section IV looks at related work and Section V gives our conclusions.

II. Overture

Nodes discover overlays by finding peers that are already participating in a P2P network. This is accomplished by having the node first examine its local cache for previously–known peers. If none are found, or the information is stale, the node multicasts a join request. When a peer receives a request, it may multicast a reply and all nodes in the network cache the information. The node wanting to join the overlay then selects the best peers based on criteria discussed in Section II-C.

A. Multicast Query and Multicast Reply

The idea of multicasting both a join request and reply has been used previously in Hassanein *et al.* [1]. All nodes in the

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

MANET must join a multicast group which is used for both sending join requests and responding to them. Any multicast routing protocol may be used, such as the one presented in [1], Multicast Ad Hoc On–Demand Distance Vector (MAODV) [2], or On-Demand Multicast Routing Protocol (ODMRP) [3]. For unicast messages, any unicast routing protocol may be used. Pseudo–code for the actions of each node in the MANET for the Overture bootstrapping algorithm is given in Figure 1. For simplicity, the pseudo–code shown assumes that a node will only attempt to connect to one peer at a time.

When a node wishes to join a P2P overlay, it multicasts a join request (*mc_join_request* message) to the multicast group. It is possible for multiple overlays to exist in the MANET, reflecting different P2P applications. If the node knows which overlay it wishes to join, this can be indicated in the initial request. Otherwise, the join request is considered generic and nodes belonging to all overlays in the MANET may respond.

All peers in the overlay that are willing to accept a connection from the node multicast a response (*mc_join_reply* message) back to the group. This response includes the peer's address and remaining energy. If a peer in the overlay is not willing to accept any new connections, it ignores the request.

All devices that receive the response cache the information, even if they are already part of a P2P overlay. Nodes in the MANET may receive multiple responses for which they then use the selection criteria presented in Section II-C to connect with the number of neighbors they require. Current members of the overlay may decide that the peer response they have received represents a better neighbor in the overlay, either because it is closer or has more energy. In this case, the peer may send a unicast connection request (*conn_request* message) to that peer. If an open neighbor space is available, a *conn_reply* message will be received. Otherwise a *conn_full* message will be received.

B. Caching

When deciding to join the P2P overlay, a node first consults its local cache and selects peers based on the information contained therein. If there are fewer known peers than the node wishes to connect to, or the information about a peer is believed to be out of date, the node proceeds with a multicast query as described in Section II-A. Otherwise, unicast connection requests (*conn_request* messages) are sent to the desired peers.

Cached information is maintained in a soft state, meaning that it is time limited. When a multicast response is received, it is placed in the cache with a fixed time-to-live value. This value is then decremented as time passes, and the entry is removed from the cache when it reaches zero. Even though it is expected that most peers will remain in the overlay for a prolonged period, the information concerning their remaining energy and whether or not they are accepting new neighbors is not long–lived.

When a node caches the response only the information provided in the peer response, which includes its address and remaining energy, is stored, not the route to it and so a large cache is not needed. If the cache is full, the new entry will replace the one with the shortest TTL value.

Non-overlay Nodes:

```
join_overlay:
  sort cache by TTL
  remove expired entries
  sort cache by utility value
  retrieve peer with highest utility
  if cache_hit
    send conn_request to peer
  else {
    multicast mc_join_request
    set timeout value for mc_join_request

timeout for mc_join_request:
  set best_peer = null
  for each entry in cache {
    calculate utility value P for entry
    if utility > best_peer.utility
        set best_peer = cache entry
  }
  send conn_request to best_peer

received mc_join_reply:
  add reply to cache

received conn_response:
  add peer to list of neighbors

received conn_full:
  remove entry from cache
  call join_overlay function
```

Overlay Peers:

```
received mc_join_request:
  if no available neighbor slots
      or wrong overlay {
    ignore mc_join_request
  } else {
    multicast mc_join_reply
  }

received mc_join_reply:
  add reply to cache
  if joined to same overlay as reply {
    calculate utility value P for responding peer
    if response.utility > lowest neighbor.utility
      send conn_request to peer
  }

received conn_request:
  if have available neighbor slots {
    reply with conn_response
    add node to list of neighbors
  } else {
    reply with conn_full
  }

received conn_response:
  remove neighbor with lowest neighbor.utility
  add peer to list of neighbors

received conn_full:
  ignore
```

Fig. 1: Pseudo-code for Overture bootstrap algorithm

C. Selecting Peers

When a peer provides its response to a join request by sending a *mc_join_reply*, it includes its address so that the potential neighbor knows how to contact it. It also provides its remaining energy level, e. When a node receives the response it then combines this information with its knowledge, provided by the underlying routing protocol, of the distance in hops

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

to the peer, h. In this paper, we focus on an unstructured Gnutella–like [4] overlay for simplicity.

The determination of the best peers is determined by the Cobb-Douglas utility function [5] $P = h^{\alpha} \times e^{1-\alpha}$. h and e are both normalized, and α is a parameter indicating the preference of a nearer peer to a longer–lived one. When $\alpha = 0$, this indicates that the node wishes to select the peer with the most remaining energy regardless of its distance. This may result in an overlay topology which does not closely reflect the underlying network topology, however the connection to the peer is likely to remain for a longer period of time. On the other hand, when $\alpha = 1$, this indicates that the node wants a neighbor that is closer, no matter how little energy it has remaining. The benefit of matching the overlay topology to the underlying network is that fewer hops are required when overlay neighbors communicate, which results in reduced energy usage network–wide.

When a node joins an overlay, it must connect to n peers, where n varies depending on the P2P network being used. The selection algorithm presented in this section is applied to all responses, and the list is sorted based on utility. The n top peers are selected and *conn_request* messages are unicasted to them. If a peer rejects the connection request because its neighbor list is full, a *conn_full* is received and the node simply selects the next peer on the list and tries again. When a *conn_response* is received, this indicates that the node has been successful in selecting a neighbor and the peer is added to the neighbor list.

III. Simulation Results and Analysis

This section evaluates the performance of Overture for MANETs with different sized P2P overlays for different distance–energy preference values (α) and compares it to Random Address Probing (RAP) with extra knowledge. In RAP nodes send join requests to random addresses in the hope that they are connected to the overlay.

A. Simulation Parameters and Metrics

The MANETs are simulated in *ns–2*, have an area of 1000m $\times$ 1000m, and contain 60 nodes. It is assumed that only one P2P overlay exists, as this makes no difference in how the bootstrapping algorithm works. All nodes are evenly distributed in the simulation area. The random waypoint model is used for mobility with nodal velocities distributed according to a uniform distribution with minimum speed of 1 m/s and maximum speed of 3 m/s. The pause time is uniformly distributed with a mean of 60 seconds and the simulation time is set to one hour. MAODV is used as the multicast routing protocol and AODV is used as the unicast routing protocol. A peer maintains a maximum of three connections to other peers.

In each simulation experiment, the distance–energy preference parameter (α) in the utility function is varied. In addition, experiments for bootstrapping using RAP were also performed in order to compare them with Overture. Experiments for Overture were carried out both with and without caching turned on in order to verify its benefit. Each experiment begins with the overlay existing in a steady state. During the remainder of the experiment, a randomly selected node attempts to join the overlay every 20s, and a randomly selected peer leaves the overlay every 20s. A file transfer is generated between two random overlay nodes every 20s. Several runs were completed for each experiment with the results averaged.For the RAP scheme the first three nodes to respond are the ones selected for connection. There are a total of 60 MANET nodes in each experiment, but the number of P2P overlay nodes is restricted to no more than 30, 45, or 60, depending on the experiment.

TABLE I
Energy Consumption Constants Used in Simulation

m_{send}	1.89	$mW \cdot sec/byte$
b_{send}	246	$mW \cdot sec$
m_{recv}	0.494	$mW \cdot sec/byte$
b_{recv}	56.1	$mW \cdot sec$
$b_{sendctl}$	120	$mW \cdot sec$
$b_{recvctl}$	29.0	$mW \cdot sec$

When conducting experiments with the normal Random Address Probing scheme, nodes were virtually never able to connect to the overlay at all due to the random nature of selecting addresses from a relatively large address space and the comparatively few nodes that are part of the overlay. On the Internet, where within a subnet there may be hundreds or even thousands of overlay nodes, the chances of a successful probe are much more likely. In a MANET, where there are far fewer overlay nodes, the RAP technique is much more likely to meet with failure. Therefore, in order to provide a useful comparison, the Random Address Probing technique was given additional information in the form of the addresses of all nodes existing in the MANET so that a valid random MANET node would be chosen each time.

The energy consumption model used in the simulations is the linear model proposed by Feeney [6]. Each MAC layer operation takes a certain amount of power as defined by $cost = m \times size + b$ where m is the incremental cost of the operation, b is the fixed cost, and $size$ is the amount of data sent or received. The constants are obtained by physical measurements for a Lucent IEEE 802.11 WaveLAN PC Card from [6] and are summarized in Table I.

B. Simulation Analysis

In P2P overlays on MANETs, it is more efficient if the overlay topology reflects the underlying network, because if two neighboring peers are on opposite sides of the network, a single message must travel the entire network diameter. This results in not only higher delay due to the distance the message travels, but also in greater energy use for each MANET node that is involved in forwarding data. Therefore, Overture allows the user to connect to proximate peers. Figure 2 shows the results for several values of the preference parameter, with and without caching, and for RAP. An α–parameter value of 1.0 indicates that nodes are attempting to select the shortest hop distance peers as neighbors, and as can be seen, they are generally successful at doing so. Overture allows peers that are already part of the overlay to try to connect to nearer neighbors when they receive a response to a query initiated by another

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

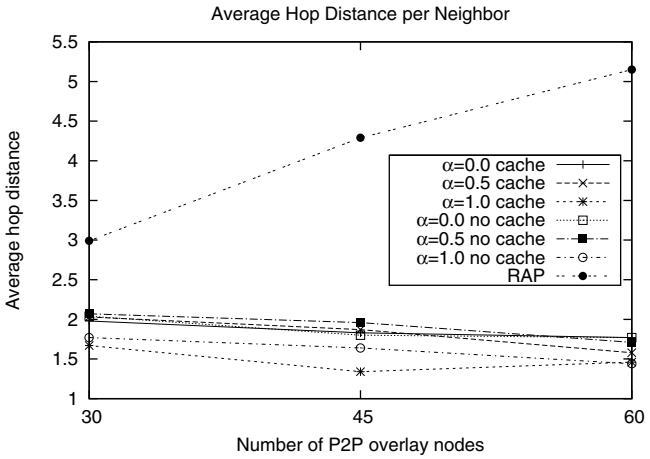

Fig. 2: Average neighbor hop distance

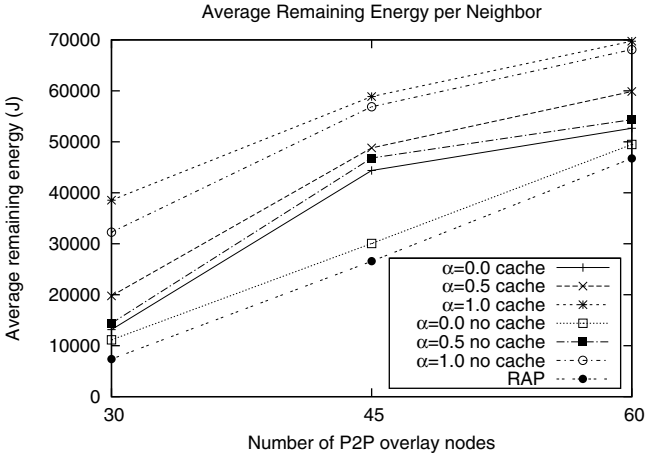

Fig. 3: Average neighbor remaining energy

node. This helps peers to maintain shorter connections and is seen by the much lower neighbor distance when compared with the RAP experiments.

When connecting to peers, it seems like a good idea to select ones that have more energy remaining as this is an indicator of how much longer they are likely to remain alive before their battery expires. However, selecting a longer–lived peer may require choosing one that is a further hop distance away. The utility–function used in the bootstrap algorithm allows the user to ignore distance and focus solely on energy by choosing an α–parameter value of 0.0. The average energy remaining for a node's neighbors is given in Figure 3 for several parameter values, both with and without caching, and for RAP. Lower α values actually result in greater overall energy usage because overlay neighbors will tend to be further apart, resulting in overlay traffic being forwarded by more nodes. The use of caching reduces the number of messages that need to be transmitted, which in turn increases the amount of energy nodes have available. Neighbors selected by the RAP algorithm have the lowest available energy because they tend to be further away.

It is important to consider how many messages are sent in order to join the overlay as this is a good measure of the

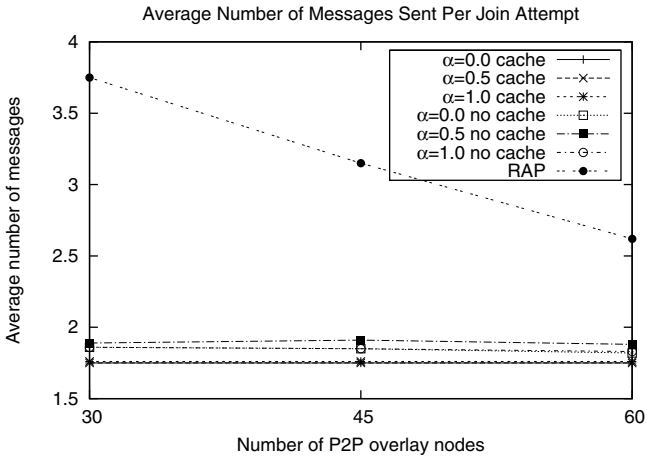

Fig. 4: Average number of messages sent per connection attempt

overhead of the algorithm. Also, energy use is directly related to the number of messages exchanged, as seen in Table I. Therefore, the fewer messages/packets sent, the better. Figure 4 indicates how many messages are sent for each connection attempt during the bootstrapping process for different α–parameter values, with caching turned on and off, and also for RAP. The messages needed are fairly constant for Overture, though slightly higher if the cache is not used. There is neglible difference for varying α values, as the parameter has no effect on the number of messages sent. Random Address Probing requires significantly more messages per connection attempt despite the possibility of Overture multicasting a request. However, as the number of overlay nodes increases the number of messages RAP requires falls due to the greater chance of contacting an overlay node. RAP cannot match Overture's performance here because the peer RAP selects to connect to may have all of its neighbor slots full, whereas this is unlikely to be the case for Overture.

Another important consideration when attempting join an overlay is the amount of time that passes between initiating the bootstrapping algorithm and successfully connecting to a peer. Figure 5 indicates how much time is required for the average join attempt for several parameter values, both with and without caching, and for RAP. The use of caching reduces the join delay due to fewer messages needing to be transmitted. As with message overhead, there is no difference when varying values of α, as the parameter has no effect on the delay. The delay for RAP is several orders of magnitude higher because it must randomly try nodes, most of which it cannot join. In between attempts, it must wait for a response before trying another node.

The goal of the cache is to reduce the number of message transmissions required, and as seen in the previous experiment, it is successful in doing so. Each cache miss results in a network–wide multicast query and potentially several multicast responses. In contrast, a cache hit requires only a few unicast message exchanges. Therefore, a higher cache hit rate results in fewer multicast transmissions. In our experiments, the cache hit rate was not set at a fixed number, but instead the

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

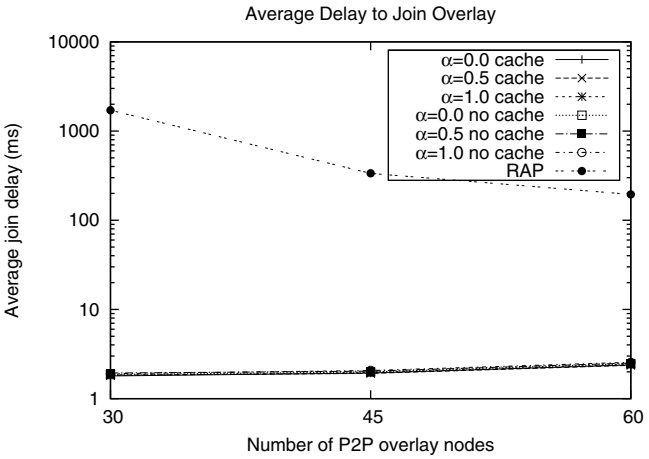

Fig. 5: Average join delay

cache was built up as would be done in a real system. A hit rate of between 71 and 75% was achieved, a fairly constant rate regardless of experiment type, resulting in significant savings in message transmissions. This indicates that this cache hit rate is a property of Overture, as it does not vary significantly with different distance–energy preference parameter (α) values.

IV. Related Work

This section examines the various bootstrapping mechanisms that exist for wired P2P networks.

Knoll *et al.* survey some bootstrapping protocols for wired P2P overlays [7]. The main techniques in use are divided into two categories, peer–based and mediator–based. The peer–based techniques include peer caching, in which each node keeps a list of previously active peers; multicasting, which is impractical on the Internet; and random address probing, in which a random address is sent a join request. Overture makes use of multicasting and a form of peer caching and in Section III is shown to be superior to RAP in MANETs.

The second category consists of mediator–based schemes, which include the use of a central server, host caches which require the node to retrieve a list of peers from a specific URL, server lists, and trackers. None of the mediator–based approaches are practical in MANETs because they all require some sort of centralized infrastructure, which cannot be assumed available in a MANET.

Conrad and Hof propose a generic bootstrap service designed for the Internet [8]. In their approach, all nodes belong to a specific bootstrap overlay, from which bootstrap information about a particular P2P overlay can be acquired. This is similar to our work, in which all network nodes must belong to a specific multicast group. Conrad and Hof use a variation of RAP called Local RAP which checks IPs in the local subnet first since they are more likely to be active. To distribute the P2P network information, any structured protocol can be used.

The authors propose the use of a "salt value" to prevent all queries from going to a single node since some P2P networks are enormously popular. This salt value is added to the name when hashing and spreads out the keys. To support small bootstrap networks, which would have many missed salt queries, the Conrad and Hof propose an adaptive salt window algorithm, which adapts the salt value based on a window which varies with the size of the network.

Castro *et al.* similarly propose the idea of a universal overlay to provide the bootstrap service [9]. Their design is based on the structured overlay Pastry [10] and provides mechanisms to advertise and discover services, to contact nodes, and also to acquire the application code required to install the desired service. They use multicasting, and provide persistent storage and distributed search.

V. Conclusions

This paper presented Overture, a bootstrapping algorithm for finding and joining a P2P overlay that is running on a mobile ad hoc network. To our knowledge, this is the first bootstrapping algorithm proposed for these types of networks. Nodes multicast an overlay join request to the entire MANET, and if a peer has an available neighbor slot open, a multicast response is sent. All nodes, whether or not they are part of an overlay, cache the response for possible future use. After a sufficient time, the querying node chooses among the best available peers based on a utility function which indicates its preference for choosing a closer or longer–lived neighbor.

Simulation results demonstrate that Overture connects to closer nodes than the RAP scheme, uses less energy, and also requires fewer messages and significantly less delay to join the overlay. The cache was shown to significantly reduce the number of packets sent. In the future, we will consider delay and bandwidth factors in addition to hop distance for determining node "closeness".

References

[1] H. Hassanein, Y. Yang, and A. Mawji, "A new approach to service discovery in wireless mobile ad hoc networks," *Int. J. Sensor Networks*, vol. 2, no. 1/2, pp. 135–145, 2007.

[2] C. E. Perkins and E. M. Royer, "Multicast operation of the ad-hoc on-demand distance vector routing protocol," in *ACM/IEEE MobiCom*, Seattle, WA, 1999, pp. 207–218.

[3] S. J. Lee, W. Su, and M. Gerla, "On–demand multicast routing protocol in multihop wireless mobile networks," *Mobile Networks and Applications*, vol. 7, no. 6, pp. 441–453, 2002.

[4] The annotated gnutella protocol specification v0.4. [Online]. Available: http://rfc-gnutella.sourceforge.net/developer/stable/index.html

[5] H. Varian, *Intermediate Microeconomics: A Modern Approach, 6th ed.* W. .W. Norton & Company, 2003.

[6] L. M. Feeney, "An energy–consumption model for performance analysis of routing protocols for mobile ad hoc networks," *Mobile Networks and Applications*, vol. 6, no. 3, pp. 239–250, 2001.

[7] M. Knoll, A. Wacker, G. Schiele, and T. Weis, "Decentralized bootstrapping in pervasive applications," in *IEEE International Conference on Pervasive Computing and Communications Workshops (PerComW)*, 2007, pp. 589–592.

[8] M. Conrad and H.-J. Hof, "A generic, self-organizing, and distributed bootstrap service for peer-to-peer networks," in *IWSOS 2007, LNCS 4725*, 2007, pp. 59–72.

[9] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron, "One ring to rule them all: service discovery and binding in structured peer-to-peer overlay networks," in *ACM SIGOPS European workshop*, 2002, pp. 140–145.

[10] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, November 2001, pp. 329–350.