4-2008

An Overlay Discovery Algorithm towards a Pure Distributed Communication System

Charilaos Tsivopoulos
Sheffield Hallam University

Jawed Siddiqi
Sheffield Hallam University

Babak Akhgar
Sheffield Hallam University

Shahram Rahimi
Southern Illinois University Carbondale

Morvarid Bassir
University of Maine - Main

Published in Tsivopoulos, C., Siddiqi, J., Akhgar, B., Rahimi, S., & Bassir, M. (2008). An overlay discovery algorithm towards a pure distributed communication system. Fifth International Conference on Information Technology: New Generations, 2008. ITNG 2008, 212-217. doi: 10.1109/ITNG.2008.63

An Overlay Discovery Algorithm towards a Pure Distributed Communication System

Charilaos Tsivopoulos, Jawed Siddiqi, Babak Akhgar
Informatics Group
Sheffield Hallam University
Sheffield, UK
{ c.tsivopoulos, j.i.siddiqi, b.akhgar}@shu.ac.uk

Shahram Rahimi
Department of Computer Science
Southern Illinois University,USA
rahimi@cs.siu.edu

Morvarid Bassir
Research Assistant, WiSe-Net Laboratory
University of Maine, Orono, ME, USA
mbassir@eece.maine.edu

Abstract

The present paper proposes an architecture for a pure peer-to-peer communication system that is free from centralized coordination and knowledge. To meet this decentralization requirement the current paper focuses on a novel Overlay Discovery Algorithm which allows peers to connect to the network in an efficient and dynamic fashion. The system consists of three core modules which enable peers to meet, organize and communicate respectively. In addition, a series of simulation results is presented as a proof of concept for the Overlay Discovery Algorithm.

Keywords: P2P, distributed, bootstrap, peer-to-peer, overlay discovery

1. Introduction

Peer-to-peer (P2P) networks have recently experienced rapid growth leading to a wide range of distributed systems with varying architectures. The common underlying principle of all P2P systems is the exchange of decentralized resources among the network nodes [1]. The P2P paradigm has several gains to offer and therefore, many technologies have diminished or eliminated the centralized features from their design. In turn, the resulting systems can enjoy: ad hoc communication, out of the box operation [2], higher resilience and robustness [3], cost effectiveness [4] and greater scalability. The achieved level of decentralization dictates the magnitude of the mentioned benefits. Ideally, a pure P2P system will be able to exhibit the highest performance in the above metrics.

In order to build a pure P2P system, a peer must be able to perform the two core tasks of joining the overlay without using any centralized mechanisms. First, find the Internet Protocol (IP) address of a contact node to join the overlay. Second, after joining the overlay, locate other nodes and desired resources.

The overall research direction of the current effort addresses both steps. However, this paper focuses on the first task (bootstrapping phase), addressing the deficiency of existing approaches that do not exhibit the desired degree of distribution, via the Overlay Discovery Algorithm (ODA).

Section 2 lists and describes the existing mechanisms used for the bootstrapping phase and identifies their weaknesses so as to justify the development of the ODA. Section 3 initially provides an overview of the system's architecture, then focuses on the Overlay ODA and details its design and operation. Section 4 presents and discusses the simulation results for the ODA. The conclusions drawn are based on the experimental evidence the theoretical foundation of the proposed algorithm. Finally, Section 5 contains a

978-0-7695-3099-4/08 $25.00 © 2008 IEEE
DOI 10.1109/ITNG.2008.63

discussion on the ODA and lists future steps in the research.

2. Limitations in the current SOA

The role of the bootstrapping phase is to enable peers to find online nodes and connect to the overlay. The present paper considers that the existing approaches do not address this task with the desired degree of decentralization [5] [6]. They are either centralized, offer low efficiency or may fragment the overlay and hinder the resources' global availability. The most popular of the up to date bootstrapping mechanisms along with their deficiencies are previewed below.

Bootstrap Servers: This is the most widely adopted method and utilizes a number of dedicated bootstrap nodes. Peers are aware a-priori of the servers' IP addresses and try to connect to one of them whenever they wish to join the overlay. It is obvious that this approach introduces centralized and static features which limit the system's scalability; namely, the bootstrap servers are potential points of failure and bottlenecks. The consequences being extra resources in terms of cost, time and manpower are consumed to build and maintain the infrastructure.

History list: Alternatively, a peer may maintain a list of address nodes that were met in the past [7]. The next time the peer wishes to connect, it will attempt to contact one of the nodes in the list. This approach is completely decentralized; however its performance is doubtful and not efficient at all. If the list contains a large number of nodes the bandwidth and time overhead for finding one online can be very high. Moreover, if the overlay is dynamic it is highly possible the list to become outdated very soon and subsequently the user will fail to join the overlay. Finally, if peers don't have identical lists it is highly possible the overlay to be fragmented into sub-overlays. If different versions of the list bear no node in common then the owners of the lists will not be able to meet as to form a single overlay. As a result, a peer will not have access to the entirety of the resources but rather to a subset of them which resides in its sub-overlay. Since no corrective mechanisms are in place as to ensure lists persistence, fragmentation is prominent. Even in the case that the lists are big enough to ensure that at least one node will be in common the effort to find it will be inexpedient.

Employing Network Layer Mechanisms: Peers can use their knowledge of the underlying network topology to find an online node to connect [12]. So, if a peer knows that other peers reside in the same network segment it can try to connect to them by using multicast or broadcast mechanisms. This approach could be very efficient in a small subnet with a low peer churn rate. However, if the network size grows or the probability of other peers to be online is low then the generated broadcast traffic and time overhead will increase dramatically. This is because more messages must be sent out to locate an online peer. In addition, this approach could not span over many subnets since broadcast traffic is usually blocked. Finally, it is very difficult for sub-overlays belonging on different network segments to merge. Hence, peers cannot access resources on other sub-overlays.

3. System Architecture

The proposed system's design employs three core modules as illustrated in Figure 1.

The Overlay Discovery (OD) module incorporates a mechanism that enables peers to meet and join the overlay in a decentralized fashion; it will be explained in Section 4 and the proposed algorithm removes the need for Bootstrap nodes.

The P-Grid Overlay module is responsible for the organization of the overlay and the interactions that take place among online peers. It is based on the P-Grid package (p-grid.org) which is an implementation of the P-Grid algorithm [8].

The Communication Application module is responsible for the users' communication and exchange of resources.

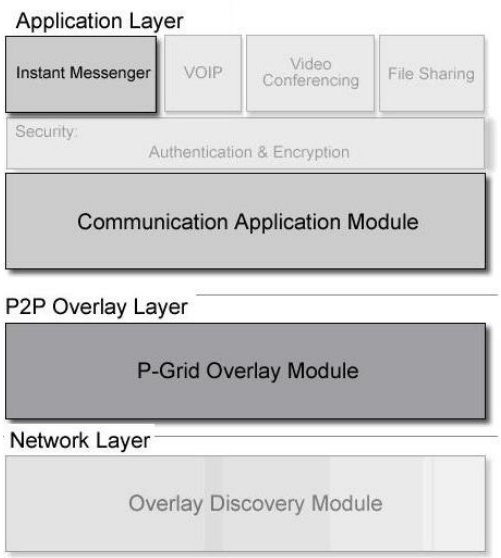

Figure 1. System Overview.

The current fully operational modules are the P-Grid Overlay and the Communication Application. At this point the application supports only instant messaging and has the potential in future to incorporate other types of

resources exchange. In regard to the OD module, it is at the stage of the theoretical background development and design. It is to this that we now turn our attention.

3.1 Overlay Discovery Algorithm (ODA)

The fundamental concept of the ODA is that every peer has to maintain an IP List, containing an adequate number of the IP addresses that have ever accessed the overlay along with their probability to be online. Peers will consult their IP List as to find the most likely node to be online. The reason for tagging IP addresses with a probability indicator is twofold. First, is for a peer to connect to the overlay with as little time and communication overhead as possible. It is quite reasonable an IP address that it is online most of the time to be also online at the time a peer makes a new attempt to connect. Second, is to define a meeting point for all peers and create bridges between the potentially different meeting points. Since all peers will try to connect to the IP addresses with the highest online probability, it is highly possible to meet with each other. Hence, the creation of sub-overlays is avoided.

Before continuing further and describing the ODA operation it would be useful to divide its structure into four sections. The first explains how the *Online Probability Indicator* of an IP address is calculated. The second describes how peers can meet for the first time. The third section provides an overview of how peers build their IP Lists. The last describes how a peer can search through its IP List as to find an online node.

Online Probability Indicator: The first thing a peer has to do is to determine the IPs probabilities. In order to achieve that every peer maintains a record of all the IP addresses it has ever used to access the network along with the exact time it accessed and left the overlay with each IP address. It must be reminded that a peer/user can access the network with multiple IPs over time. Using the above collected information a peer first calculates the following probabilities for each of its used IPs:

T_f = Time since the IP *first* appeared on the overlay, $T_f > 0$.

T_o = Total time the IP is *online*, $T_f > T_o$.

In turn, T_f and T_o are used to calculate the Online Probability Indicator (O) for every IP address that equals:

$$O = (T_f - T_o) / T_f$$

This enables every peer to know the Online Probability Indicator for all the IPs it has ever used.

The First Meeting: Now let's assume that we wish to give birth to a new overlay. How can we do that if peers have never met? There are two alternative mechanisms that can be used. The suitability of each method depends mainly on the overlay's size. The first one is ideal for small overlays while the other is better suited for big ones. More specifically**,** in case that the overlay has a small size and all peers are in the same subnet, peers may utilize a *Flooding-Like* mechanism as to meet. Every peer will send a "Hello" message to all neighbour IP addresses on its right until it finds an online node. For example if a peer's IP address is X, it will first send a message to X+1 then to X+2 and so on until it finds an online peer. In this way every peer will meet its neighbours, and through them the rest of the overlay. In this case a bootstrap node is not required at all and the network will operate in a pure P2P fashion from day one.

On the other hand, if a large scale network is the case, the *Temporary Bootstrap Node* approach can be used. More specifically, every peer receives an IP List containing a single IP address artificially set with online probability equal to 1 (100%). This IP address belongs to a Bootstrap node. In this way peers will meet for a period of time at a specific node as to build consistent enough IP Lists. As the network matures, the popularity of the Bootstrap IP will be gradually decreased by the system administrator to allow the overlay to become independent. When the overlay is considered mature enough, the Bootstrap node is completely removed. In order for a new user to join the overlay for the first time it must either receive an IP list from an old user or perform the Flooding-Like approach.

Building the IP List: After peers meet each other and knowing what information to collect, the next step is to start exchanging and processing this information so as to build their IP List. The exact mechanism is still under development. However, the primary mechanism is illustrated in Figure 2 and described below.

Note: Regarding the content of the figures below, IPX indicates the X^{th} most popular IP address. For example IP2 is the second most popular IP address in the List. The most popular IP of an overlay and a subnet is represented by a dark grey node while the rest are in a light gray.

The mechanism is as follows; whenever a peer (IPX) connects to the overlay (Step 1), it receives the latest version of the *IP List* from the Bootstrap node (Step 2). In turn, the peer performs a search to find other online peers belonging on the same subnet (Step 3). All peers belonging on the same subnet use the same IP addresses to access the overlay and therefore collect correlated statistical information. It must be reminded that different peers might have used the same IP to access the network at some point in past. This means that various peers may

have Online Probability Indicator information for the same IP. In turn, these peers will exchange their gathered common information to calculate the IP popularity of each IP in their subnet (Step 4). If one of the subnet's IPs has a popularity indicator higher than the current most popular IPs then the IP List will be updated (Step 5) and forwarded (Step 6) to some of the most popular Bootstrap nodes. From that point on every peer that connects to the overlay receives the updated IP List.

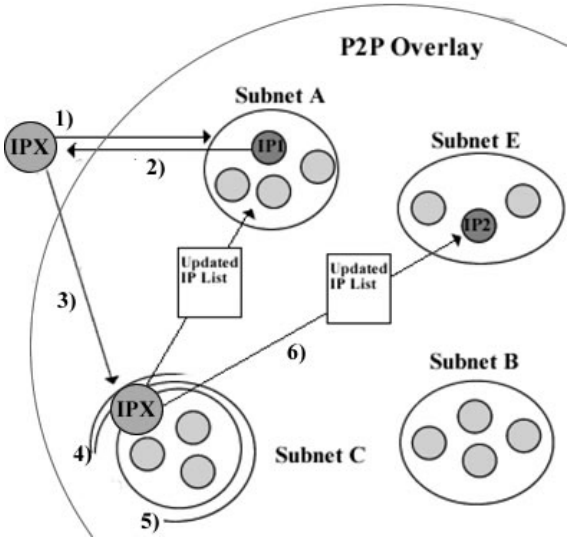

Figure 2. Building the IP List.

Searching the IP List: At this point it is assumed that a peer achieved the creation of, or was given access to, a valid IP List. Hence, we explain how a peer can use it to find an online node to which it can connect. The mechanism is quite straightforward. Having in mind that the IP addresses in the List are ranked based on their online probability indicator, a new peer chooses the first and most popular IP address and tries to connect. If the connection fails, it continues to the second one and so on. The search range *(R)* will define the number of IP addresses that will be checked before the peer stops its attempt to connect. Figure 3 below, illustrates an unsuccessful search using *R* equal to N, where N is one more than the number of unsuccessful connection attempts.

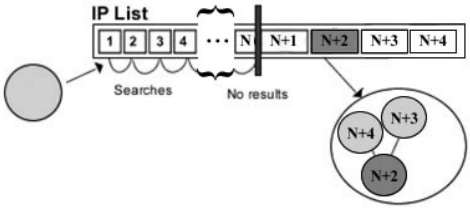

Figure 3. Search Process.

4. Simulations

The goal of this series of simulation experiments was to study the behavior of the Overlay Discovery module and define under what circumstances an overlay can be fragmented and how the search efficiency can be performed.

4.1 Simulation Methodology and Tools

The simulation environment was developed in Java. The system was considered to be at the mature state and therefore all peers had identical IP Lists. The simulation resembled an overlay of size 40 to 100 peers. The IP Lists had sizes ranging from 60 to 200 IP addresses. Every IP address on the List had a popularity indicator ranging from 0 to 1. At any given time all, some or none of the peers could be online. A given simulation cycle lasted for 1,000 "time instances," every peer could access the overlay zero or more times using single or multiple IP addresses from the IP List. None of the enhancements were included. The search range R varied among the different simulation cycles.

4.2 Observational Model

First of all, the study of the simulation results led to the creation of a formula that can be used to calculate the optimal search range *R* which guarantees that no fragmentation occurs. As a result, optimal usage of network and computational resources is achieved.

It was observed that an overlay is fragmented only when a new peer fails to find and connect to a currently online node and creates its own sub-overlay instead. The reason for doing so is because the IP address of the contact node is out of the search range (Figure 3).

Before we move further, the following two probabilities should be defined:

P (F): is the probability of the overlay to be fragmented.

$P_R(F)$*:* is the probability of failing to find a currently online IP address in the search range *R.* (Figure 3).

Since the "failure to find a currently online IP address in the search range" results in the "overlay's fragmentation" therefore:

$$P(F) = P_R(F) \qquad (1)$$

The probability of all IPs in the search range *R* to be offline is:

$$P_R(f) = P_1(Off) * P_2(Off) *\ldots * P_R(Off)$$

Where $P_i(Off)$ is the probability of the i^{th} IP in the List to be offline. Therefore,

$$P(F) = \prod_{i=1}^{i=R} P_i(Off) \qquad \textbf{(2)}$$

The formula above can be used from now on to calculate the optimal search range *R* which guarantees that no fragmentations occur.

4.3 Validation of the Observation Model

A number of simulation experiments have been conducted in order to check the above formula **(2)** and demonstrates its validity. For example the scenario below was simulated for 10 cycles:

The IP List had size equal to 200.

The number of users was 100.

The first three IPs had online probability equal to 0.9 and offline probability *P(Off)* = 0.1each.

The rest IPs had probabilities<0.5.

The search range was *R*=3.

The results indicated that on average only in one instance out of 1,000 a sub-overlay was created, which means that *P(F)*=0.001. This finding was also validated by the formula as follows: *P (F) = 0.1 * 0.1 * 0.1=0.001*

The above formula has great benefits to offer to overall performance of the algorithm; namely, it can be used to calculate the optimal size of the IP List which can be decreased down to the search range while fragmentations can still be avoided. Subsequently, peers can minimize their bootstrap phase efforts and utilize less physical storage. Regarding the above scenario, if *P(F) = 0.001* is satisfactory, the IP List's size *(S)* can be decreased from 200 to 3. Therefore, peers don't have to store the rest of the 197 IP addresses. If a smaller *P(F)* is desired, based on formula **(2)** the value of the optimal search range *R* can be recalculated.

4.4 Inherited Resilience to Fragmentation

The findings below are of great importance because they indicate that the algorithm has incorporated resilience to fragmentation. Surprisingly, it was observed that even when sub-overlays where created the system was able to overcome them without the use of any *Corrective* mechanism. More specifically, the life expectancy of a sub-overlay is quite short. Overlays once fragmented do not remain fragmented for ever because over time a sub-overlay prevails and the rest disappear. This is because sub-overlays constructed of IPs with low online probability will gradually shrink, as these IPs go offline very soon. On the other hand, new peers use their List and connect to more popular sub-overlays that are growing. The actual process is described below and is represented as a sequence of time instances where an event takes place.

At the time instance T=1 a peer with the IP7 going online. The search range used is R=10. However, as we can see in the figure below, IP7 fails to find the online overlay because the current most popular IP is out of range.

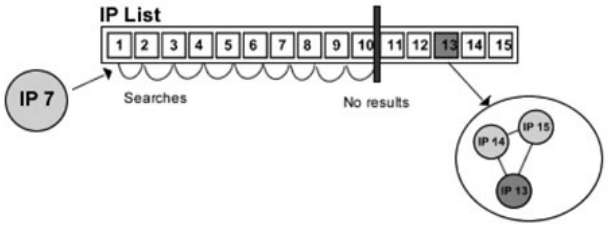

Figure 4. T=1.

Therefore the peer with IP7 creates its own sub-overlay.

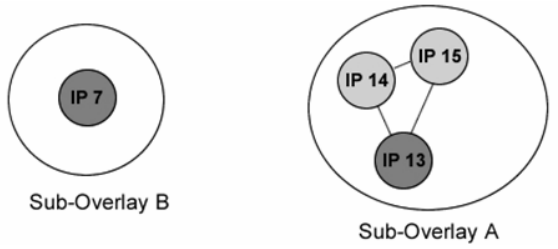

Figure 5. T=1.

During the next time slot T=2, two new peers go online and connect to the current most popular address which is IP7. As a result sub-overlay B grows in size. At the same time IP14 and IP15 go offline because of their low online probability and sub-overlay A is shrinking. During the experiments it was observed that the online probability for IP addresses such as IP14 and IP15 was around 0.15.

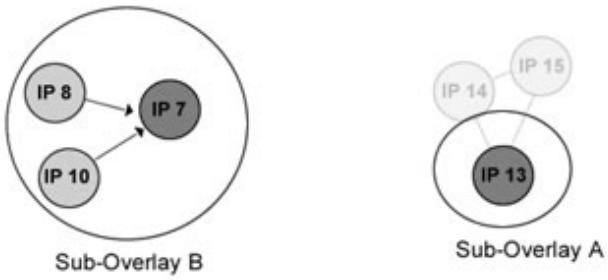

Figure 6. T=2.

Note: Faded nodes represent peers that went offline.

Finally at the time instance T=3, IP13 goes offline and sub-overlay A stops to exist. Sub-overlay B is from now on the only overlay and new peers continue to connect to it. Note: Nodes colored with both light and dark gray represent IPs that were previously the most popular ones.

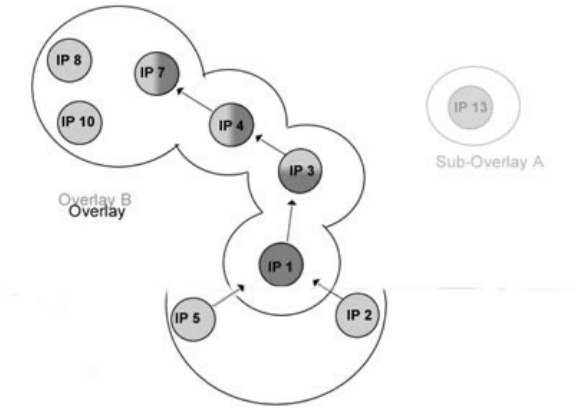

Figure 7. T=3.

5. Conclusions and Future Enhancements

It can be concluded that the proposed algorithm can address all of the existing bootstrapping approaches limitations. Undoubtedly, the OD module is still in its infancy; nevertheless the simulation results have validated that all of the benefits below are feasible.

1) Peers are able to connect to the overlay without the need for dedicated Bootstrap nodes (Section 3.1).

2) The IP List (Section 3.1) in conjunction with the produced formula (Section 4.2) ensures that peers can access the overlay with optimal communication and time overhead.

3) Finally, the current approach prevents and limits the creation of sub-overlays. The simulation results (Section 4.4) demonstrated that ODA is resilient to fragmentation.

Future steps that are being considered are as follows. The crystallization of the IP List building mechanism for the OD algorithm that is currently under development. Load balancing features are to be incorporated so as to uniformly distribute the load over the contact nodes. Moreover, a number of extra features can be added to the algorithm's operation to improve it. So far, these are the *Time Factor* metric and a *Corrective* mechanism.

Regarding the first, let's acknowledge the fact that there are periods of the day during which an IP is more likely to be online. By utilizing the *Time Factor* metric we can take advantage of this phenomenon. For example, in a period of time (e.g 24h) an IP address having an online probability equal to 0.5 may correspond to an online probability of 1 for the half duration (12h) and of 0 for the other half. Therefore, in this context the total 0.5 indicator is rather misleading since it does not reveal the whole truth about this "IP's habits". By introducing the Time Factor, peers can check which IPs are more likely to be online at the time they wish to connect which leads to better decision making. Concerning the second feature, the purpose of the *Corrective* mechanism is to ensure that the overlay is not fragmented. Variations in the IPs ranking among IP Lists may lead a group of peers to connect to node "A" and another group to node "B". In this way sub-overlays "A" and "B" are created. To avoid such events peers will try to connect not only to the first most popular IP but also to a number of subsequent less popular IPs. In this way a bridge will be built between sub-overlays and they will finally merge. The greater the number of IPs each peers connects to the greater the chances to avoid the creation of sub-overlays and to preserve consistency.

References

[1] Kan, G., "Gnutella, Peer-to-Peer: Harnessing the Power of Disruptive Technologies", A. Oram (ed.), *O'Reilly Press*, USA, 2001.

[2] Singh, K. and Schulzrinne, "H. Peer-to-peer Internet Telephony using SIP", *Columbia University Technical Report CUCS-044-04*, New York, 2004.

[3] Kurmanowytsch, R. Kirda, E. Kerer, C. Dustdar, S. "OMNIX: A Topology-Independent P2P Middleware.", *CAiSE Workshops*,2003.

[4] Roussopoulos, M. Baker, M. Rosenthal, D. T. G. Maniatis, P. and Mogul. J. "*P2P or Not P2P? In Proceedings of the 3rd International Workshop on Peer-to-Peer Systems (IPTPS04),* 2004.

[5] C. Cramer, K. Kutzner, T. Fuhrmann, "Bootstrapping Locality-Aware P2P Networks", *Proc. IEEE International Conference on Networks (ICON) 2004*, Singapore, 2004, vol. 1, pp. 357--362.

[6] S. Martin and G. Leduc "Ephemeral State Assisted Discovery of Peer-to-Peer Networks**",***Proc. of IEEE ACNM'07*,Germany, 2007.

[7] I. Clarke, T. W. Hong, S. G. Miller, O. Sandberg, and B. Wiley,"Protecting Free Expression Online with Freenet", *IEEE Internet Computing*, 2002,vol. 6, no. 1, pp. 40–49.

[8] Aberer, K. "P-Grid: A self-organizing access structure for P2P information systems", *Sixth International Conference on Cooperative Information Systems* (CoopIS 2001), Italy, 2001.