Bridging Distributed Hash Tables in Wireless Ad-hoc Networks

Lawrence Cheng
Department of Computer Science, University College London, Torrington Place, London, WC1E 6BT, UK.
l.cheng@cs.ucl.ac.uk

***Abstract*—The adaptation of structured P2P networks, i.e. Distributed Hash Tables (DHTs), to wireless ad-hoc networks has been investigated in recent years. Existing work assume all peers would come to an agreement on establishing one uniform DHT across the entire network. However, in reality, there isn't a de-facto standard of DHT implementation, different DHTs co-exist. We present a novel protocol, known as the *DHT-gatewaying* model, which enables cross-DHT searching between multiple DHTs of different implementations.**

Keywords-ad-hoc; cross-DHT searching; Distributed Hash Tables (DHTs); gatewaying; self-organisation; wireless networks.

I. INTRODUCTION

Distributed Hash Tables (DHTs) have been used to implement structured Peer-to-Peer (P2P) overlays i.e. overlays with topologies that are tightly controlled [1]. Example DHT protocols are Content Addressable Networks (CANs) [5], Pastry [6], Chord [7], and more. Recent years, research effort has been made to adapt existing DHT techniques for dynamically changing and heterogeneous wireless ad-hoc networks [2][3][4]. In [9][10], techniques for enhancing the efficiency of deploying DHTs in wireless networks were proposed. Techniques for optimising DHT routing locality, which is important for deploying DHTs in wireless networks due to limited bandwidth availability, were presented in [11]. However, these approaches were designed based on a fundamental assumption: that at some point in time, large number of (wireless) nodes would come together, and would express their willingness to establish among themselves a DHT using the same DHT technique (i.e. all peers use the same DHT implementation, such as a 160-bit Chord-DHT) [14]. As a result, a single, *common DHT keyspace*[1], is established (or to be established) among all participating peers. This is not a practical assumption, since currently, there isn't a dominating DHT implementation that is supported/preferred by *all* P2P applications in the network. In reality, a more practical assumption is that peers will join different DHTs (that are currently available in the network) based on their own capabilities and preferences [14]. As a result, peers in a (large) network are members of *different* co-existing DHTs. In our previous work [14], we have addressed this issue, and developed a DHT bootstrapping protocol for establishing different DHTs in a large-scale wireless network that suit individual's needs. Figure 1 illustrates how multiple DHTs may co-exist in today's Internet and/or in a wireless environment. Note that nodes are interconnected via either ad-hoc links or the Internet. Node Z is physically connected to node N and node M, the latter are members of the 256-bit Chord-DHT (i.e. DHT 3). These nodes are marked in squares with thick black lines. We will discuss more about these nodes later on in the text.

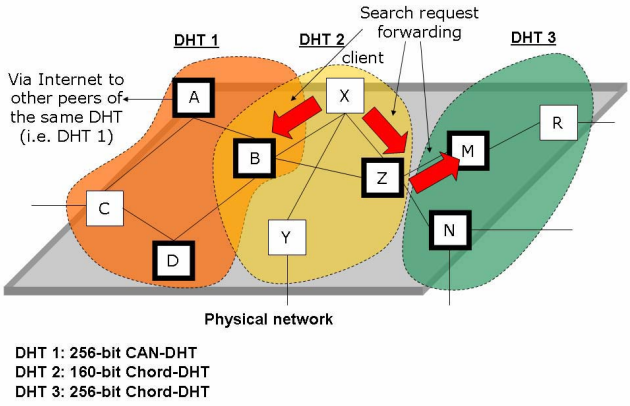

Figure 1. Example of co-existing DHTs in the network

Note that, as shown in Figure 1, existing DHT protocols would enable node A to search for a piece of data within DHT 1 only. *Cross-DHT searching*, on the other hand, would cover more peers; hence improves the quality of the search. By cross-DHT searching, we mean that a client (that originates a data search request) can search not only in its DHT, but also in other nearby, established DHTs. In other words, the scope of the search is larger because more nodes are searched. This paper presents a novel protocol, known as the DHT gatewaying model, which enables scalable and efficient data search across *homogeneous*, *heterogeneous*, and *assorted* DHTs that co-exists in a wireless/wired network[2].

[1] By a common DHT keyspace, we mean that members of the DHT share the same keyspace structure (i.e. one DHT keyspace shared among all peers).

[2] By homogeneous DHTs, we refer to the DHTs that use the same DHT implementation and keysize (i.e. both DHTs are 160-bit Chord-DHTs). By heterogeneous DHTs, we refer to the DHTs that use the same DHT implementation, but different keysize (i.e. a 160-bit Chord-DHT and a 256-bit Chord-DHT). Assorted DHTs means the DHTs use different implementations and/or keysize (i.e. a 160-bit Chord-DHT and a 256-bit CAN-DHT).

1930-529X/07/$25.00 © 2007 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2007 proceedings.

II. BACKGROUND

Different schemes to enable cross-DHT searching were proposed. In our previous work [8], we have evaluated these approaches. Due to space limitation, a summary will be provided here. In [12], it was explained that *complete merging* (which uses standard DHT protocols such as [5][6][7] and random joining) is inefficient. In [12], the first protocol for enabling cross-DHT searching was proposed, known as *simple merging*. Simple merging merges peers of different Chord-DHTs into one Chord-DHT by discarding all DHTs but one. Peers joining the DHT are given special keyspace portions to minimise overhead. Simple merging differs from traditional approaches, such as complete merging, in the sense that less overhead is incurred because the random keyspace partitioning process is avoided. In [8], we presented a DHT composition (and decomposition) protocol that enables peers of a CAN-DHT to be absorbed into another one with low overhead. A common feature of the protocols presented in [8] and [12] is that they merge all nodes into one DHT. Clearly, discarding DHTs and forcing individual peers to (re)join another DHT increases overhead, especially if the original occupancy of the DHTs being discarded is large[3]. This is not preferred for bandwidth limited wireless networks. In addition, existing approaches assume that the peers of the discarded DHTs support the implementation of the DHT that they are about to join. This assumption, as explained, is not practical. A more practical approach would be to retain existing DHTs, and to allow searches to be conducted *across* existing DHTs, without (re)establishing (new) P2P relationships.

III. THE DHT-GATEWAYING PROTOCOL

A. Assumptions

We assume that only some peers in the network support different types of DHT implementations. Some peers are directly connected to peers that are members of other DHTs. These peers are known as the *Virtual Gateways* (VGs) in this paper (those marked in thick line squares in Figure 1). Readers are referred to our previous work [14] for more detail on DHT routing optimisation and DHT maintenance. Note that we are dealing with mobile ad-hoc devices; thus, we assume that all devices have some, but limited processing capabilities (e.g. processing power, memory, power, etc.).

B. Basics of the Gatewaying Protocol

The gatewaying model is designed for supporting cross-DHT searches across homogenous, heterogeneous, and assorted DHTs. The incompatibility problem between different DHTs is avoided by not modifying existing DHT keyspace structure, but to establish links between them and use compatible/suitable nodes to carry out request conversions. We will present our design using the network layout shown in Figure 1. A sequence diagram that shows the basics of the gatewaying protocol is shown in Figure 2. Suppose a client (i.e. node X) wants to conduct a search for a piece of data. Note that because node X is a member of DHT 2 only, the (initial) search is limited to within DHT 2. Therefore, node X will hash the identifier of the piece of data of interest using the supported algorithm of DHT 2 (say, SHA-160), and use the resultant key identifier to locate (indirectly through DHT overlay routing) the peer in DHT 2 that holds the data (i.e. a simple DHT search operation).

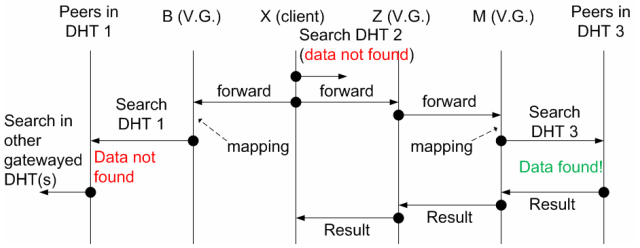

Figure 2. A sequence diagram illustrating simple gatewaying

Assuming that the data cannot be found in DHT 2 (i.e. MP3 file not found), node X then decides to search for the same piece of data in the (entire) network (i.e. cross-DHT search). For cross-DHT search, the simplest way would be for node X to forward its original search request (not the corresponding key identifier) to node B and node Z respectively. These nodes will forward the search request to peers of DHT 1 and DHT 3 respectively, as if they are the original data search requester (i.e. they act on behalf of node X in a transparent fashion). More specifically, because node B is a member of DHT 2 *and* DHT 1, node B will "map" the search to a format that is supported in the DHT 1. By mapping, we mean that the data identifier will be hashed using the 256-bit hash algorithm that is supported in DHT 1. Note that node Z, however, is not a member of DHT 3 (because it has the same problem as node X that it does not support the implementation of DHT 3). But because node Z is physically connected to node N and node M (which are members of DHT 3), node Z will forward the search request to one of the nodes (e.g. node M), which will carry out the mapping on behalf of node Z.

Once the mapping is done, the search will be conducted in DHT 1 and DHT 3 respectively. If the search is successfully, the result will be returned to node B and node Z respectively (because in the point of view of the peers in DHT 1 and DHT 3, these are the "original" requesters). Node B and node Z will forward the search result(s) to node X. To reduce bandwidth usage, for data-specific items (such as a named MP3 file), the search stops when the data is located. For network-wide information (such as "find the link with the best throughput to the Internet"), the search continues from DHT 1 and DHT 3 respectively to their immediately connected DHTs (except DHT 2) until all interconnected DHTs have been searched. To avoid looping, a Time-To-Live (TTL) value is added to the original search request from node X; this value is decremented each time the request traverses a DHT domain.

[3] A 160-bit DHT could, theoretically, support upto 2^{160} peers. Although in practise this number is unlikely to be reached; however, DHT is designed for scalable content-addressing in *large-scale* networks. Thus, it is fair to assume that the number of peers in a DHT tends to be large.

1930-529X/07/$25.00 © 2007 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2007 proceedings.

IV. The Protocol Design

A. Distributed Mapping of Search Requests

We explained that mapping is needed to map a data request to a form that is supported by the current DHT implementation. The overhead of mapping is dependent on two relative factors: the number of mapping requests, and the capabilities of the virtual gateways. We have assumed that virtual gateways have limited capabilities, so distributed mapping is therefore preferred. Distributed mapping is more scalable: when a search request is passed to a virtual gateway that is overloaded, the virtual gateway does not carry out the mapping itself, but passes the request onwards to other members of the targeted DHT where the mapping will be carried out.

More specifically, when a virtual gateway is overloaded, it passes the requests (that were originated from members of other DHTs) to one of the peers that are also a member of the same DHT as itself (not to a member of the DHT where the client's request originated from). This is a peer that owns the virtual gateway's immediate neighbouring overlay keyspaces i.e. one of the virtual gateway's immediate overlay neighbours. Note that immediate overlay neighbours are the peers that a DHT node is aware. The immediate overlay neighbour will carry out the mapping on behalf of the virtual gateway; and when it also becomes overloaded, it will pass the requests onto one of its immediate overlay neighbours (except the one from which the mapping request originally came from). Hence, balanced loading is achieved. The benefit of our arrangement is that the path between nodes is minimised (routing locality is assumed), thus reduces unnecessary overhead on long distance (i.e. multi-hop) communications between mobile devices (which share a limited bandwidth and have limited capabilities). Another advantage is that, no addition state maintenance overhead is added by this arrangement: this is because the standard DHT protocols require peers to keep up-to-date status of their immediate overlay neighbours; so, peers can re-use the available information to distribute load.

B. Virtual Gateway Determination

We have mentioned that a peer is a virtual gateway either when it is a member of more than one DHT, or when it is physically connected to a node that is a member of a different DHT. The former is known to the virtual gateway itself (that it has more than one P2P application installed locally and it is connected to more than one DHT); the latter is discovered during a simple handshake process between peers of different DHTs that are physically connected. This handshake involves passing on basic DHT identification information (such as an identifier of the DHT) of the communicating peers. For example, when node Z establishes ad-hoc links with node M and node N respectively (when they are in close physical proximity), the nodes exchange DHT identity information. If the received identity information differs from the local DHT identity, this means that the neighbouring node is a member of another DHT. The nodes will keep state of this information. Note that this handshake contains lightweight DHT identity information only (i.e. an integer that represents the DHT ID). Note further that, this handshake takes place between *physical* neighbouring nodes only. Due to the nature of ad-hoc connections (i.e. limited range), the number of physical neighbouring nodes to one node is always limited. Thus, the additional overhead is marginal. The handshake would need to be re-established only when the physical link is re-established.

C. Self-Organising Gateway Pointers

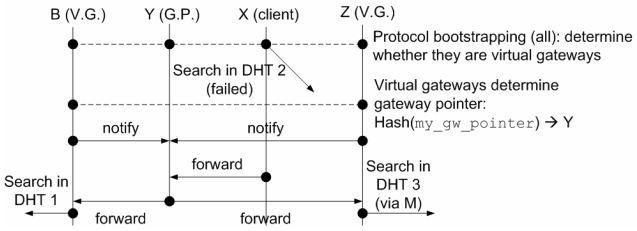

Figure 3. Gatewaying with gateway pointers

Note that, virtual gateways must announce themselves, so that peers of the same DHT as the virtual gateway know: a) it is possible to do cross-DHT searches to enrich their data searches; and b) to which member(s) in the DHT the cross-DHT search requests should be forwarded. If a peer has determined that it is a virtual gateway, but does not want to carry out mappings for others, it may avoid doing so simply by not disclosing its new status (i.e. as a virtual gateway) to others. As discussed in an earlier section, a simple solution for announcement would be to arrange every peer in the network – when becoming a virtual gateway – to multicast to all nodes in the DHT, indicating its new status. This arrangement, however, does not scale well and assumes (relatively) static virtual gateway status (suitable for fewer notifications). To avoid explicitly notifying *all* nodes in the DHT, a more scalable solution is to arrange a subset of members of the DHT to keep track of the "where-about" of virtual gateways. These nodes are known as the *gateway pointers* in this paper. Peers will forward cross-DHT search requests to these nodes, which will subsequently pass the requests to the virtual gateways of the DHT. In other words, these nodes form a special service overlay in a DHT: members of this service overlay act as receptionists, which (re)directs cross-DHT requests to virtual gateways. The sequence diagram for gatewaying with gateway pointers is shown in Figure 3. The benefit of having gateway pointers is that, instead of having all nodes to be kept up-to-date with all real-time virtual gateway status, only a subset of nodes in the network need to keep real-time status of virtual gateway information (i.e. there is no need for *all* nodes in the network to be updated with up-to-date information on virtual gateway status). The gateway pointers – as the name suggests – keeps a list of up-to-date virtual gateway's network addresses (e.g. IPs). The solution is therefore more scalable and manageable.

Note that there is no need for dynamic negotiation to select gateway pointers in our solution. For peers to locate gateway pointers, they simply map a common identifier to a keyspace identifier (by hashing the identifier using the algorithm supported by the DHT implementation), and route their requests through DHT overlay routing to the peer that holds the keyspace which includes the keyspace identifier. If the corresponding keyspace is hosted locally, the peer knows it is a gateway pointer. For scalability, more than one gateway

1930-529X/07/$25.00 © 2007 IEEE

pointer may be needed. To locate the second gateway pointer, a peer hashes twice the common identifier (Figure 4). More specifically, a peer may randomly select an integer *m*, and hash *m* times to determine the m^{th} gateway pointer in the DHT. *m* should be ≤ *n* (i.e. the number of desired gateway pointers in the DHT). The only requirement is for the DHT implementation to set a value of *n*. This value may depend on the occupancy of the DHT (i.e. more peers, more gateway pointers to achieve scalability). Currently, we assume that the value of *n* is fixed at DHT bootstrapping. This is acceptable, given that our current scenario involves P2P wireless devices that are connected via the Internet with fixed P2P hosts (such as home PCs). This includes the large number of home users that are connected via the Internet, which tends to be steady [13]. Thus, the *total* number of peers (i.e. including both wireless and wired users) is therefore steady.

SHA-256(`my_gw_pointer`) → keyspace point that refers to the first `gw_pointer` (#1)

SHA-256(SHA-256(`my_gw_pointer`)) → another keyspace point that refers to the second `gw_pointer` (#2)

Figure 4. Multiple hashing to derive keyspace points which refer to gateway pointers

A virtual gateway, like other ordinary peers, carries out the mapping procedure described above to locate the gateway pointers. It periodically sends messages to one of the gateway pointers (through DHT overlay routing) to indicate that it has become (or still is) a virtual gateway that connects to DHTx. This message includes the network address of the virtual gateway itself and the ID of the gatewayed DHT. The intercepting gateway pointer keeps this information locally in a virtual gateway address table, and propagates the information to other gateway pointers by multicast. Note that because gateway pointers are a subset of peers of the DHT, the scale of this multicast is much smaller than multicasting to all peers in the DHT. If an entry is not updated after a timeout period, the entry is deleted automatically on all gateway pointers (which presumes the corresponding virtual gateway is dead, or no longer reachable, etc.). When a peer needs to carry out a cross-DHT search, it locates the gateway pointer(s) by the same procedure. The search request is first sent to one of the gateway pointers, which will subsequently forward the search request to the virtual gateway(s) using its virtual gateway address table. Our solution is robustness: should a gateway pointer drops out, its keyspace will be taken over by its immediate overlay neighbour. The neighbour then becomes a gateway pointer. Should in any case a device is supposed to be a gateway pointer (because it has the corresponding keyspace), but it is unable/unwilling to do so, it becomes an *incapable node*. The incapable node may appoint one of its immediate DHT overlay neighbours to become a deputy gateway pointer. Any request received by the incapable node will be forwarded to this neighbour. If the neighbour is unwilling to become a gateway pointer as well, the neighbour may appoint one of its immediate overlay neighbours, and so on.

V. EVALUATION

A. Setup Overhead

In this evaluation, we compare the DHT-gatewaying approach with the complete merging approach, and the simple merging approach presented in [12]. Data search will be carried across two 16-bit DHTs (each with a maximum capacity of 65,536 peers). We assume the DHTs are either 20% or 50% occupied (i.e. a 20% occupancy of a 16-bit DHT means there are 2^{16} x 20% nodes in the DHT). According to [12], complete merging requires all but one of the DHTs to be discarded. Each peer then joins the remaining DHT using standard DHT protocol. The simple merging approach requires each peer to be map to an appropriate key map in the remaining DHT, and the new joining peer updates its new immediate overlay neighbour(s). The DHT-gatewaying protocol requires virtual gateways to notify gateway pointers. For the gatewaying approach, we assume that either 50% or 10% of the nodes in the gatewayed DHTs are virtual gateways. The number of gateway pointers (i.e. the *n* value) is fixed at 1,000. This value can be adjusted to any value, depending on the scale of deployment. Figure 5 shows the results.

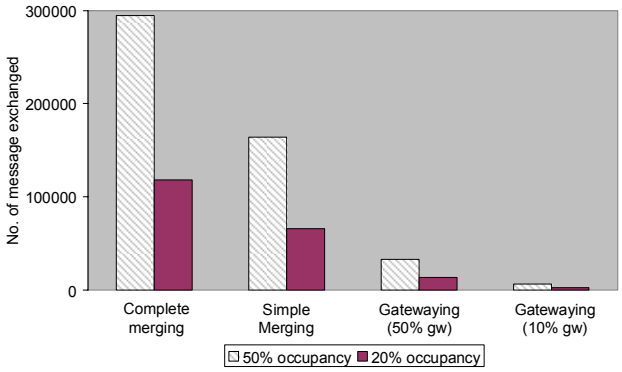

Figure 5. Setup overhead under different approaches

From the results, it is clear that the DHT-gatewaying model creates much less setup overhead than other approaches. This is because, the complete merging approach requires each of the peers of the discarded DHT to go through a series of communications prior to joining a DHT. The peer must first contact an existing members of the destination DHT, then calculates randomly the keyspace that it wants, obtains the keyspace from the original owner directly… etc., and neighbourhood information must be updated to reflect the new entry. Note that because each node joins *individually* with the remaining DHT, and each joining process involves a series of message exchanges (as explained); so, the total number of message exchanged is therefore *proportional* to the occupancy of the discarded DHTs, and depends on the complexity of the joining process. In contrast, the gatewaying model setups cross-DHT searches with the lowest overhead because it requires neither keyspace re-assignment, nor neighbourhood table updates, nor changes to finger tables. What is needed for gatewaying is for the virtual gateways to notify the gateway pointer(s) of their existence. The total number of message

1930-529X/07/$25.00 © 2007 IEEE

exchange is therefore dependent on the number of virtual gateways only.

B. *Operational Efficiency*

We have discussed that gatewaying does not modify the original keyspaces of the gatewayed DHTs; so, technically, two gatewayed DHTs consist of two individual DHTs, that would need to be searched individually (because they do not have a common keyspace). If the complete/simple merging model were used, the search would be conducted in one single DHT. This is because these protocols merge the DHTs into one. Therefore, it is not surprising that, data searches in terms of total *overlay hop count* in a DHT composed through complete/simple merging would be more efficient than data searches in gatewayed DHTs. However, overlay hop count does not give an accurate evaluation of the efficiency of the models. In gatewaying, searches are conducted in different (connected) DHTs (more-or-less) *simultaneously* (see later). Figure 6 illustrates the time delay differences between different approaches. The y-axis shows the number of distinctive DHT keyspace after merging/gatewaying: for complete/simple merging, this number is always one because the DHTs are merged into one. For gatewaying, because the resultant (gatewayed) DHT is formed by several DHTs which retain their original keyspace structures, the value therefore depends on the number of DHTs that are being gatewayed. Note that a minor delay is introduced when searching through gatewayed DHTs. This time delay is needed for forwarding searches between gatewayed DHTs through gateway pointers and virtual gateways.

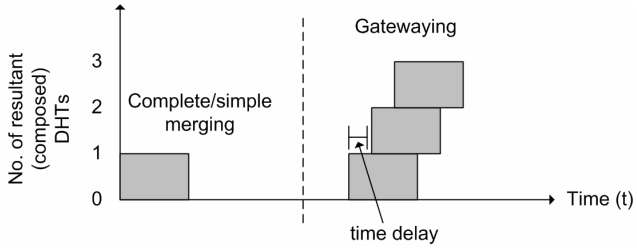

Figure 6. Time delay between different approaches

VI. CONCLUSION

Due to the rapid growth of more powerful wireless end user devices and the increasing demand of media sharing services through P2P applications, the use of DHTs in mobile wireless networks has been investigated in previous research. Existing research work assumes one common DHT keyspace structure. We have discussed in this paper that, since there isn't a de-facto implementation of DHT, in reality, different DHTs co-exist in the network. Since existing DHT protocols allow searches to be conducted within the same DHT only, there is a need to investigate a solution that searches across different DHTs, that covers more nodes in order to enrich the search results. The solution must be distributed, scalable and efficient due to limited bandwidth availability and limited processing power on end-user wireless devices. In this paper, we presented the DHT gatewaying protocol, which searches across homogeneous, heterogeneous, and assorted DHTs in a self-organising and load balancing manner. Our protocol is applicable to bridging DHTs in both wireless and wired networks (i.e. the Internet). To the best of our knowledge, our protocol is the first of its kind.

ACKNOWLEDGEMENT

The author would like to thank Dr. Steve Hailes for his advises and support.

REFERENCES

[1] E. Lua, J. Crowcroft, M. Pias, R. Sharma, S. Lim, "A Survey and Comparison of Peer-to-Peer Overlay Network Schemes", in IEEE Communications Survey and Tutorial, March 2004.

[2] J. Li, J. Jannotti, D. D. Couto, D. R. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing", in Proceedings of the 6th ACM International Conference Mobile Computing and Networking, Boston, Massachusetts, USA, August 2000, pp. 120–130.

[3] The IETF Mobile Ad-hoc Networks (MANets) Working Group, http://www.ietf.org/html.charters/manet-charter.html

[4] B. Mathieu, M. Song, A. Galis, L. Cheng, K. Jean, R. Ocampo, Z. Lai, M. Brunner, M. Stiemerling, M. Cassini, "Self-Management of Context-Aware Overlay Ambient Networks", accepted as short paper in the Proceedings of the 10th IFIP/IEEE Integrated Symposium on Integrated Network Management (IM), Munich, Germany, May 2007.

[5] S. Ratnasamy, P. Francis, M. Handley, R Karp, S. Shenker, "A Scalable Content-Addressable Network", in Proceedings of the 2001 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM), San Diego, CA, USA, Aug 2001, pp. 161-172.

[6] A. Rowton and P. Druschel, "Pastry: Scalable, Distributed Object Locaiton and Routing for Large-Scale Peer-to-Peer Systems", in Proceedings of IFIP/ACM Middleware, Hiedelberg, Germany, Nov 2001, pp. 329-350.

[7] I. Stocia, R. Morris, D. Karger, M. Kaashoek and H. Balakrisnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications", in Proceedings of the 2001 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM), San Diego, CA, USA, Aug 2001.

[8] L. Cheng, R. Ocampo, K. Jean, A. Galis, C. Simon, R. Szabo, P. Kersch, R. Giaffreda, "Towards Distributed Hash Tables (De)Composition in Ambient Networks", in Proceedings of the 17th IFIP/IEEE Distributed Systems: Operations and Management (DSOM), Dublin, Ireland, Oct 2006.

[9] H. Pucha, S. Das, Y. Hu, "Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks", in Proceedings of the 6th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), English Lake District, UK, Dec 2004.

[10] T. Zahn, J. Schiller, "MADPastry: A DHT Substrate for Practicably Sized MANETs", in Proceedings of the 5th Workshop on Applications and Services in Wireless Networks (ASWN), Paris, France, Jun 2005.

[11] S. Zhou, G. Ganger, R. Steenkiste, "Balancing Locality and Randomness in DHTs", Technical Report, CMU-CS-03-203, Nov 2003, http://www.pdl.cmu.edu/PDL-FTP/stray/CMU-CS-03-203.pdf

[12] T. Heer, S. Gotz, S. Rieche, K. Wehrle, "Adapting Distributed Hash Tables for Mobile Ad Hoc Networks", in Proceedings of the 4th IEEE International Conference on Pervasive Computing and Communications Workshop (PERCOMW), Pisa, Itably, March 2006, pp. 173-178.

[13] The Register, "The BitTorrent Measurements Analysis", http://www.theregister.co.uk/2004/12/18/bittorrent_measurements_analysis/

[14] L. Cheng, K. Jean, R. Ocampo, A. Galis, P. Kersch, R. Szabo, "Secure Bootstrapping of Distributed Hash Tables in Dynamic Wireless Networks", in Proceedings of the IEEE International Conference on Communications (ICC), June 2007, Glasgow, UK.