Towards Distributed Hash Tables (De)Composition in Ambient Networks

Lawrence Cheng[1], Roel Ocampo[1], Kerry Jean[1], Alex Galis [1], Casba Simon[2], Robert Szabo[2], Peter Kersch[2], Raffaele Giaffreda[3]

[1]University College London, Electrical Engineering Department, Torrington Place, London, WC1E 7JE, UK {l.cheng, r.ocampo, k.jean, a.galis}ee.ucl.ac.uk
[2]Budapest University of Technology and Economics, Dept. of Telecoms and Media Budapest, Hungary {simon, szabo, kersch}@tmit.bme.hu
[3]British Telecommunications PLC, raffaele.giaffreda@bt.com

Abstract. When different wireless networks come in close proximity there is often a need for them to logically combine, or *compose*. We focus on a known research problem particularly in Ambient Networks (ANs), where heterogeneous Distributed Hash Tables (DHTs) contained in these wireless networks need to merge or divide as a result of these dynamic (de)composition processes, respectively. We present two novel DHT (de)composition models for ANs, known as *absorption* and *gatewaying*, that are designed to handle (de)composition of DHTs in different AN network environments, with minimal disturbance to existing member nodes.

Keywords: Ambient Networks, Composition, Decomposition, Distributed Hash Tables.

1   Introduction

The aim of the Ambient Networks (ANs) project [7] is to develop the next generation wireless networks. An AN consists of potentially large numbers of independent, heterogeneous mobile nodes that can logically interact with each other to share a common control space, known as the Ambient Control Space (ACS) [7], for resource sharing. Given that an AN may consist of large number of AN nodes, there is clearly a need for a distributed and scalable management data storage and retrieval mechanism for each AN. Previous research works [1][2][4] have suggested that Distributed Hash Tables (DHTs) is a candidate. Exa mple DHTs are Content Addressable Network (CAN) [3], Chord [6], Pastry [5], and others. However, much of the existing research mainly focuses on optimising DHT routing and scalability; and usually assumes a common DHT across the entire network that any nodes can join [2]. Since ANs may constantly compose and decompose with other ANs[1], we argue

---

[1] By AN composition, we refer to process of which the ACSs of two (or more) ANs interact with each other, to establish a common ACS between the two (or more) ANs for resource sharing. AN decomposition refers to the process of a common ACS being divided [7].

2   **Lawrence Cheng1,** Roel Ocampo1, Kerry Jean1, Alex Galis1, Casba Simon2, Robert Szabo2, Peter Kersch2, Raffaele Giaffreda3

that the successful use of DHTs in implementing various distributed management components in ANs depends on an ability to efficiently *compose* and *decompose* DHTs. By DHT (de)composition, we refer to member nodes of homogeneous or heterogeneous[2] DHTs of different ANs interacting with each other to share distributed information.

The challenge is that DHT (de)composition in ANs must be conducted in a resource-limited environment i.e. a wireless environment. Thus, in addition to the need for a more efficient underlying routing algorithm in DHTs (which is beyond the scope of this paper), there is a need to minimise the disturbance caused by the (de)composition process to existing member nodes of the DHTs. By minimising the disturbance, we mean to minimise: (a) the amount of network overhead incurred by the (de)composition process, and (b) the amount of storage data that needs to be (re)distributed to other nodes in the DHT *after* (de)composition has completed (during which keyspace might have been re-assigned to new members). As far as we are aware, there has not been a vast amount of research work on DHT (de)composition; a DHT merging process designed for DHTs based on the Chord protocol was presented in [2] (see later). In this paper, we shall outline two novel (de)composition models for CAN-based DHTs for AN. We shall also discuss the mappings of our models to DHT implementations other than CAN. We start our investigation based on CAN-based DHTs because of its design simplicity, which could help readers understand this new research challenge (of DHT (de)composition). Furthermore, this would enable us to gain a better understanding of the design requirements of DHT (de)composition, and put ourselves in a better position to evaluate, experiment with, and to tailor a generic DHT (de)composition model that would also cover other DHTs.

2   Background

In this section, we will start off with explaining some of the assumptions that we have made when designing our approaches. Then, we shall provide an overview of our approaches, by discussing their unique features and triggering factors.

2.1   Assumptions

To simplify our discussion, we assume that each AN has DHT-based management components. Each AN node will have its own keyspace in the DHT once it has joint a DHT; and each node maintains a coordinate routing table that keeps IP addresses and virtual coordinate zones of its neighbours in the approach as indicated in [3]. To simplify our discussion, we assume one DHT establishment per AN. Remember that we are interested in DHT (de)composition in this paper; thus, (pre)establishment of

---

[2] By homogenous DHTs, we refer to DHTs that use keyspace of the *same* keysize (e.g. both uses 160-bit keyspace). By heterogeneous, we refer to the opposite (e.g. 160-bit keyspace Vs. 256-bit keyspace).

DHT in each AN is assumed. Readers are referred to [1][8] for more detail on optimized DHT establishment techniques for wireless networks.

2.2   An Overview on the Two Models

We present two DHT composition models for ANs, known as *absorption* and *gatewaying*. The absorption model (Fig. 1a) refers to two (or more) individual DHTs (that are owned by two or more ANs respectively) completely merging together, resulting in one *uniform* DHT across the composing ANs. The gatewaying model (Fig. 1b) refers to *bridging* two (or more) individual DHTs together without modifying their original keyspace. Both approaches enable information sharing between DHTs, but are tailor-designed to accommodate *different* network environments (see later).
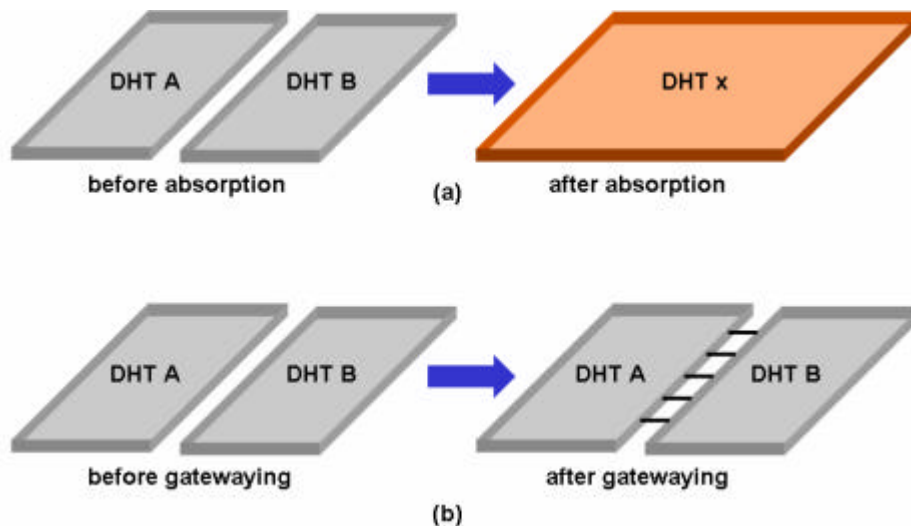

Fig. 1. The generic models of absorption and gatewaying

2.3   Triggering Factors for the Two Models

Although we do not intend to specify the exact criteria when absorption or gatewaying between DHT should be triggered[3], but to illustrate the differences between absorption and gatewaying, first, consider this deployment scenario: when a coach arrives at a train station, an unknown (but potentially large) number of

---

[3] We believe these issues should be defined by the corresponding AN/DHT Service Providers (SPs) (readers are referred to [7] for more details). Also, SPs should define policies for guiding nodes when nodes are presented with multiple joining offers from different (nearby) DHTs.

4   **Lawrence Cheng1,** Roel Ocampo1, Kerry Jean1, Alex Galis1, Casba Simon2, Robert Szabo2, Peter Kersch2, Raffaele Giaffreda3

passengers will be getting off the coach and walking towards the station. Assume each passenger is a passenger node, and DHTs have already been established among member nodes of the station and the coach respectively. Because the number of passenger nodes getting off the coach is unknown (i.e. a very dynamic situation), it would be difficult to establish a *fresh*, *new* DHT (e.g. passenger DHT) among members of such a highly dynamic group in *real-time*. Instead, because the station DHT is readily available, should passenger nodes (those getting off the coach) wish to become members of a DHT (say, to share information), they should be *absorbed* into the (more static) DHT (i.e. the station DHT). The absorption model is therefore designed to minimise the network overhead on members (whether they are existing members of a DHT or not) when many nodes attempt to join practically at the same time.

The gatewaying model is suitable when one (or more) of the ANs/DHTs come together within reachable distance, but member nodes of the AN/DHTs remain (relatively) static. For example, when two wagons are joined together at an intermediate train station, member nodes of the two wagons are likely to be static (some passengers might get off the train but the majority would stay). There is a need to bridge between the DHTs, so that the DHTs can share information.

3   The Absorption Model

3.1   The Protocol

One way of enabling absorption (as identified in [2]) would be to allow nodes (of a discarded DHT, or do not belong to any DHT) to join a stable DHT, by negotiating with nodes of the stable DHT individually using the standard procedure as [3]. Typically this would require each of the joining nodes to randomly select a keyspace, and obtain keyspace directly from the node that "owns" it in the stable DHT. We refer this as *simple merging*. The advantage of simple merging is its simplicity, i.e. no changes to the existing protocols are needed. But the drawback is that *all* key-value pairs hosted on member nodes of the discarded DHT or on individual nodes would have to be re-distributed to nodes of the stable DHT. Also, if keyspace is randomly selected, many nodes in the stable DHT must also update their neighbourhood information each time a new node joins (see later for more details). In the case of large scale deployment, this could potentially create unnecessary network traffic, which is not desirable, particularly in wireless networks.
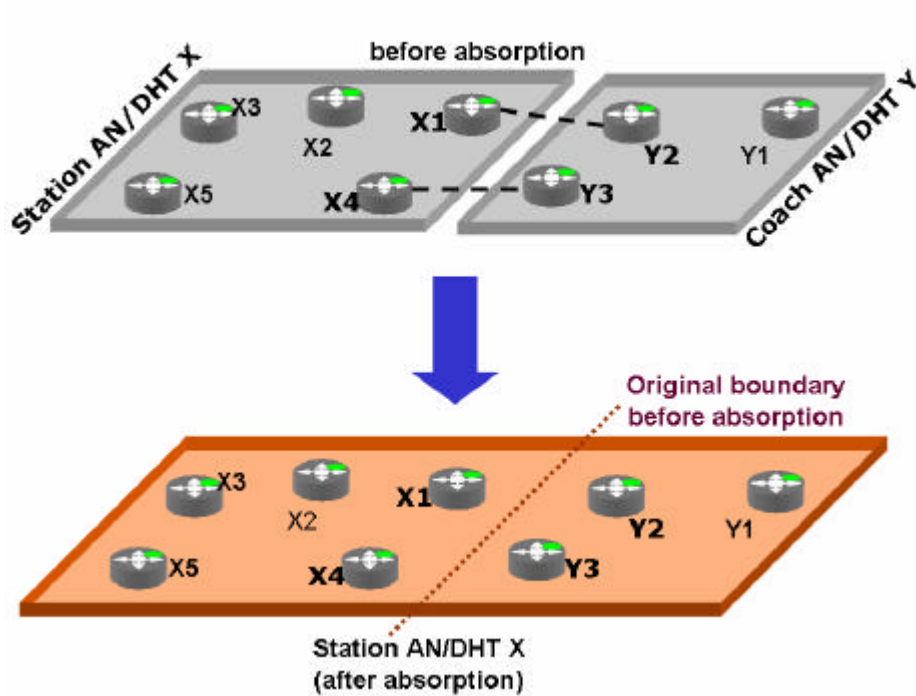
Fig. 2. The absorption model

We suggest that, absorption negotiations should be conducted through the point(s) of contact between the (two) ANs only. Points of contact are the nodes that have *physical connections* with other ANs. For example, Y2 and Y3 are the points of contact of AN/DHT Y (Fig. 2); whereas X1 and X4 are the points of contact of AN X. In this example, we assume that nodes in AN/DHT Y (i.e. the coach AN/DHT) will join with AN/DHT X (because the station AN/DHT is more stable). Instead of Y2 and Y3 randomly selecting points in the keyspace of any nodes of DHT X (which is the case in [2][3]), X1 and X4 will give up some of their keyspace to Y2 and Y3 respectively, by carefully selecting appropriate keyspace from *within* the keyspace that they own. Note that the entire absorption process is a transient process, which ceases to operate after a timeout. After the timeout, new nodes will be joining the (unified) DHT under the normal procedure; that is, by randomly selecting keyspace from any nodes in the DHT (see later).

Note that a key feature of absorption is that keyspace is selected within a keyspace (instead of splitting), with the goal of reducing the level of disturbance to neighbouring nodes. Fig. 3[4] shows the resultant keyspace partitioning under different approaches. Fig. 3a shows the original keyspace of DHT X (before composition). If nodes of DHT Y are allowed to randomly select keyspace from member nodes of DHT X, or if X1 and X4 simply split up their keyspace for Y2 and Y3 respectively, the resultant keyspace might end up as shown in Fig. 3b. As a result, X1, X2, X3, X4

---

[4] For simplicity, we illustrate our examples using a 2-dimentional coordinate space.

6   **Lawrence Cheng1,** Roel Ocampo1, Kerry Jean1, Alex Galis1, Casba Simon2, Robert Szabo2, Peter Kersch2, Raffaele Giaffreda3

and X5 would *all* have to update their neighbourhood information. This is obviously less desirable. However, by carefully selecting keyspace within the owner's keyspace (Fig. 3c), only node X1 and X4 would have to update their keyspace respectively. Once dedicated members of DHT Y (i.e. Y2 and Y3) have been assigned with keyspace, they will (re)distribute the keyspace to other members of DHT Y wishing to join DHT X (i.e. Y1) (Fig. 3d). This arrangement is again to minimise the disturbance to existing member nodes of DHT X.
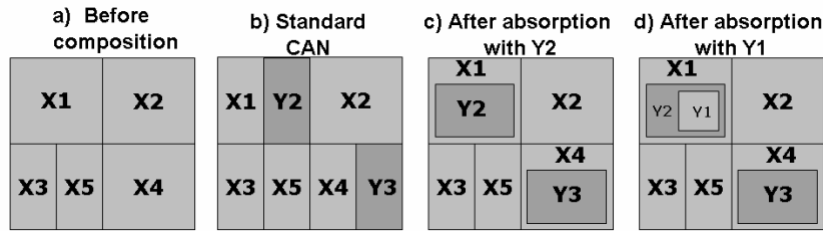

Fig. 3. Resultant keyspace ownership in different approaches

Imagine if all the remaining nodes of DHT Y (i.e. just Y1 in this case, and potentially many more) join DHT X using the standard procedure: if the scale is large (i.e. many nodes joining at once), existing member nodes of DHT X would have to expend significant resources on the tasks of keyspace partitioning and updating neighbourhood information. When absorption is used, remaining nodes (e.g. Y1) should *not* obtain keyspace from nodes other than their points of contact (i.e. Y2 and Y3). When node Y1's `join_DHT` request traverses through Y2 (or Y3), Y2 should terminate the request; Y2 should select *within* the portion of its assigned keyspace, and return the selected keyspace to Y1. The same approach is repeated between Y1 and other nodes of DHT Y: when Y1 has intercepted a `join_DHT` request from other nodes of DHT Y, Y1 will terminate the request, and will response with a selected portion of keyspace within its own keyspace. In this way, each node is responsible for (re)distributing keyspace; thus, a distributed approach for keyspace (re)distribution among the joining nodes is achieved.

3.2   Discussion

The advantage of the absorption approach is that once keyspace has been assigned by X1 and X4 to Y2 and Y3, all other existing member nodes of DHT X (except X1 and X4) are not disturbed. The process is entirely transparent to other nodes that are existing members of the original DHT X; and other nodes that were members of DHT Y may join DHT X through Y2 and Y3 (and subsequently through Y1 once Y1 has obtained its keyspace from Y2 and Y3), which is also a transparent process to existing member nodes of DHT X. Unlike the simple merging approach, our approach requires only a few nodes of the DHT (i.e. DHT X) to be disturbed, and with much less network traffic (i.e. negotiation and communications between AN/DHT are conducted between the points of contact only). Furthermore, because each node is capable of

(re)distributing keyspace, keyspace distribution is achieved in a distributed and scalable manner.

It may appear that the requirement of explicit assignation of keyspace by keyspace owners (e.g. Y2) to new joiners (e.g. Y1) violates the random balancing rules in DHTs (i.e. nodes should randomly select keyspace for balance loading). However, we argue that absorption is a transient procedure. After the timeout, new nodes must join through the normal procedure; the keyspace would eventually be randomly and evenly distributed on average. It is possible that all nodes in the DHTs have physical connectivity with each other (i.e. a fully meshed structure of physical connectivity). In this case, one may argue the use of absorption, because effectively all nodes are points of contact. However, if all (wireless) nodes are physically interconnected, then the nodes must be within close physical range. This implies that the number of participating nodes in this merging would be limited. Thus, the amount of network traffic created by, say, a simple merging, would have much less effect. It should be noted that the use of dedicated points of contact for handling keyspace does not affect scalability of the absorption model. The points of contact are responsible for *initially* collecting a portion of keyspace from nodes of the other DHT (i.e. DHT X), and redistributing keyspace to their *immediate* neighbours only. Once the immediate neighbours have obtained their keyspace from the points of contact, the immediate neighbours shall intercept (and terminate) any traversing `join_DHT` requests from other member nodes (of DHT Y), and (re)distribute keyspace to those nodes. Therefore, keyspace (re)distribution to member nodes of DHT Y is carried out in a distributed fashion.

Once new nodes have joint a DHT, they may distribute their local data to other nodes in the DHT if desired (i.e. the `put(key, value)` operation). To minimise traffic caused by many nodes putting data onto many other nodes at one time, provisioning [4] is made in our approach to upload pointers only, instead of the actual piece of data. For exa mple, if Y3 needs to put a piece of data on X2, a pointer is put instead of the actual piece of data. The pointer refers to the actual location of where the data is residing on (e.g. the data source). Thus, the amount of data storage traffic caused by (many) new joining nodes is reduced. One may argue that this arrangement increases the round-trip delay for retrieving a piece of data. However, this approach is ideal for situation where the data to be stored requires frequent updating, such as real-time bandwidth monitoring data. Instead of the data source continually updating the data values on a remote node, a requester – once obtained a pointer to the data source through the DHT – can contact the data source *directly* (readers are referred to [4] for more details).

The arrangement of giving out keyspace to a physical neighbouring node also enhanced routing locality. This is because the resultant overlay DHT neighbourhood reflects the underlying physical neighbourhood. In the standard CAN approach, keyspace ownership are randomly distributed. This means that two neighbouring overlay nodes may be in fact physically distanced. This has a major impact on routing especially in wireless networks; because if routing locality is not addressed, the overhead to route from one overlay node to another can be significantly much higher. However, at the time of absorption, routing locality is optimised in the portion(s) of the keyspace that is being given out during absorption. Note that this approach does not result in creating de facto gateways. It may appear that in Fig. 2 the absorption

8   **Lawrence Cheng1,** Roel Ocampo1, Kerry Jean1, Alex Galis1, Casba Simon2, Robert Szabo2, Peter Kersch2, Raffaele Giaffreda3

approach turns X1 to be the de facto gateway of Y2 on the DHT overlay (because to route to Y2 you must always route through X1). However, according to the underlying network connection, X1 is the physical gateway to Y2 *prior to* absorption beings. Therefore, the absorption approach does not create new gateways, but the overlay gateways are the results of absorption that reflects the underlying physical network. Furthermore, we have discussed that the absorption model is a transient process: so in the longer run, when more nodes join through the standard CAN approach, the overlay routing will become more balanced.

4   The Gatewaying Model
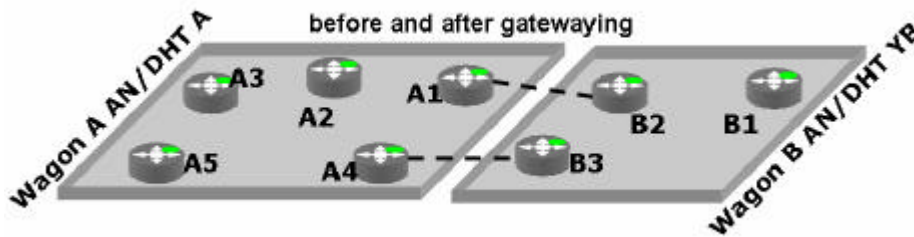
4.1   The Protocol

Fig. 4. The gatewaying model between two wagon AN/DHTs

We have mentioned in an earlier section that through gatewaying, the composed AN/DHTs would be able to share information, but at the same time retaining their original keyspace. Existing DHT approaches usually assume only one common DHT (i.e. one common keysize); however, due to the dynamic and heterogeneous nature of ANs, there is a need to support composition between DHTs of different keysize. Thus, there are two environments in which gatewaying may be deployed: gatewaying between DHTs that use keyspace of the same keysize; and gatewaying between DHTs that use keyspace of different keysize. When gatewaying between DHTs of the same keysize, nodes of the composing DHTs are notified of the existence of the other DHTs that are now becoming accessible, as well as their own gateways to the other DHTs [5]. Gateways are the points of contact which have physical connections with nodes of another DHTs; but they serve a different purpose from the points of contact in the absorption model.

---

[5] As an initial approach, notifications are sent to member nodes of a DHT through multicast. Multicast is chosen for its simplicity.

For example, nodes of DHT A (Fig. 4) must be notified that A1 and A4 are the gatewaying nodes to another DHT (i.e. DHT B). The gateways do no more than notification (i.e. they will not request for keyspace). To gain access to another (gatewayed) DHT, member nodes would need to maintain the state of at least one of their gateways[6]. Let's say A5 would like to retrieve a piece of information (after DHT gatewaying). We provide two options for this node to do a search over the gatewayed DHTs: (a) a sequential minimal-evaluation search, and (b) a parallel search. Sequential minimal-evaluation search is ideal for locating unique pieces of data (e.g. a particular video file); whereas parallel search enables a node to locate network-wide information e.g. a node that needs Internet access may wish to search all gatewayed ANs for the best available Internet throughput link that.

In the sequential minimal-evaluation search, A5 computes the location of the information in its home DHT (DHT A) in the same way as stated in the standardised protocol [3], and tries to get the information from the node (of DHT A) that is supposed to hold the information. Suppose the information of interest is not stored in DHT A (i.e. data not found); with gatewaying, the search will continue searching for the same piece of information from other gatewayed DHTs (i.e. DHT B). The node of DHT A which fails to provide A5 with the requested information, say, A3, will inform one of its gateways (i.e. A1), and requests the gateway to search for the same piece of information in other (gatewayed) DHTs. The request is conducted through the gatewaying nodes i.e. A1 and B2. B2 (i.e. the corresponding gatewaying node of DHT B) will try to locate the piece of information on behalf of A1, and will fetch over the results to A1 (and subsequently node A5) if the information can be located in DHT B. The search terminates as soon as the information is located (hence the term *minimal evaluation*), and is not forwarded to other DHTs that may be similarly gatewayed in this scenario. In contrast, in a parallel search, as the name implies, the query is forwarded simultaneously (through the gateways) to all DHTs through their respective gateways.

If the composing DHTs' keyspace are of different keysize, we use a similar approach as above, except that B2 must use the correct hash algorithm to compute the correct location in DHT B. For instance, if DHT A's keyspace is 160-bit whereas DHT B's keyspace is 256-bit, B2 must use SHA-256 to compute the correct keyspace of the requested information. Note that when a new AN node wishes to join a DHT that has already been gatewayed, the new node joins the DHT that it is in contact with.

4.2   Discussion

The advantage of gatewaying is its simplicity and reduction in network overhead. It enables information retrieval across DHTs without modifying existing keyspace structure. Thus there is no need to update neighbourhood information on each node in the gatewayed DHTs (which would be required in simple merging or, to some extent,

---

[6] Ideally for robustness, member nodes should maintain as much state of its gateways as possible. However, there is a trade-off between overhead and robustness. As an initial design, we require each node to maintain at least one of its gateways.

absorption). Also, the chances of successfully retrieving a particular piece of information increase as the number of gatewayed DHTs increases; which enhances the robustness of the overall data retrieval process (i.e. more likely to locate the piece of data of interest). The scale of state maintenance at one gateway is not dependent on the size of the neighbouring networks/DHTs, but depends only on the number of immediate neighbouring gateways, which makes our approach scalable. More importantly, because there is no change to ownership of the keyspace of the gatewayed DHTs, there is no need for data (pointer) (re)distribution, which reduces network overhead. The downside is that the gatewaying model is only applicable when member nodes of the to-be-gatewayed DHTs are relatively static. It may appear that the use of gateways for inter-DHT communications would result in some centralised processing (on the gateways). However, it should be noted that not all `get(key, value)` requests are processed by the gateways; for example, cross-DHT data search takes place only when data retrieval within a DHT fails in sequential minimal-evaluation search, and stops as soon as the data of interest is found; whereas parallel search is used for searching specific-types of information only (e.g. network-wide information). This mode therefore trades off slightly higher traffic costs and processing overhead (on the gatewaying nodes), to enhance overall robustness of the data retrieval process (when comparing to searching for data within one DHT only i.e. no gatewaying), and achieves potentially faster and more comprehensive searches over the entire composed space, with the possibility of returning multiple results from all gatewayed DHTs.

5   DHT Decomposition in ANs

By AN decomposition, an AN is virtually divided into two (or more) ANs, the decomposed ANs do not recognise the existence of each other in the view of control and management[7]. There are several decomposition scenarios. Decomposition between DHTs that were composed through gatewaying is the simplest form. This may happen when, using our previous train scenario for this example, two wagons (which were gatewayed) are detached. Nodes in the gatewayed DHT are informed that the other DHT no longer exists, and the DHTs are said to be decomposed. Decomposition of nodes of a DHT that share one unified keyspace (i.e. may have previously composed through absorption) is slightly more complicated. A node may leave a DHT without establishing its own DHT or joining with another DHT. For instance, a train passenger switches off his laptop. This situation can be considered as a node departure, and can be handled through the standard procedure as specified in [3]: the departing node either handovers its keyspace to a neighbour (i.e. a clean approach), or the unoccupied keyspace will be taken over by its neighbours when its neighbours think it is dead [3]. In other circumstances, a set of nodes may decompose from a DHT, and would like to have a DHT of their own but using the original assigned keyspace. This can be achieved by the departing nodes simply by expanding

---

[7] This does not necessary mean the decomposed ANs are *physically* disconnected: we are referring to a separation of control space during decomposition of ANs.

their own keyspace as if some other nodes have departed from the DHT (Fig. 5). For instance, if 2, 3, and 5 are the departing nodes, they will expand to occupy the remaining space that appears to be "left over" by 1 and 4. The same applies to those who did not leave the DHT. The result would be two separate DHTs, but nodes would be able to retain the original assigned keyspace structure. The last scenario would be when some departing nodes decided to create a new DHT among themselves. In this case, they must discard the original DHT, and creates a new DHT from scratch.
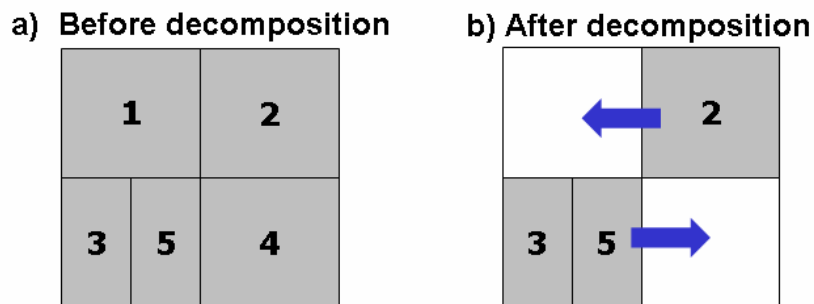

Fig. 5. DHT Decomposition

6   Conclusion & Future Work

We presented two novel models for DHT (de)composition in ANs under different networking environments. We have discussed how our designs minimise the disturbance to existing member nodes during DHT (de)composition, taken into account scalability and efficiency. We believe the research work presented in this paper gives us the insight for developing more generic models for other DHT implementations such as ring-based DHTs. As future work, we intend to investigate further into the inter-node communications during DHT (de)composition in ANs, and deploying our approaches on other DHT protocols.

Acknowledgement

References

1. 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.

12   **Lawrence Cheng1,** Roel Ocampo1, Kerry Jean1, Alex Galis1, Casba Simon2, Robert Szabo2, Peter Kersch2, Raffaele Giaffreda3

2. 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 (PERCOM), Pisa, Italy, March, 2006, pp. 173-178.
3. S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, "A Scalable Content-Addressable Network", in Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM), San Diego, CA, USA, August 2001, pp. 161-172.
4. R. Ocampo, L. Cheng, K. Jean, A. Prieto, A. Galis, "Towards a Context Monitoring System for Ambient Networks", to appear in the Proceedings of the 1st International Conference on Communications and Networking in China (Chinacom), Peking, China, 2006, temporarily available at: http://www.ee.ucl.ac.uk/~lcheng/Papers/CHINACOM_2006.pdf
5. A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems", in Proceedings of the IFIP/ACM Middleware, Heidelberg, Germany, pages 329-350, November, 2001, http://research.microsoft.com/~antr/PAST/pastry.pdf
6. I. Stoica, R. Morris, D. Karger, M. Frans Kaashoek and H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications", in Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM), San Diego, CA, USA, August 2001.
7. R. Campos, C. Pinho, M. Ricardo, J. Ruela, P. Poyhonen, C. Kappler, "Dynamic and Automatic Interworking between Personal Area Networks using Composition", in Proceedings of the 16th IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), Berlin, Germany, Sep 2005.
8. 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.