Wireless Netw (2013) 19:219–236
DOI 10.1007/s11276-012-0461-0

Distributed Hash table-based routing and data management in wireless sensor networks: a survey

Ghofrane Fersi · Wassef Louati · Maher Ben Jemaa

Published online: 15 June 2012

Abstract Recent advances in Wireless Sensor Networks (WSN) have led to a great breakthrough in sensors design and features. These technological novelties have brought additional challenges to WSN. Sensornets are seeking for new approaches for efficient data routing and management. The last few years have witnessed the emergence of several approaches that build Distributed Hash Tables (DHTs) over WSN. DHTs are initially conceived for efficient data lookup in large-scale wired networks. The main objective of this combination is to manage location-independent data and nodes identification. DHT mapping over WSN brings however new challenges. This paper presents an analytical survey on applying DHT techniques in WSNs. It describes existing DHT-based routing and data management protocols and includes a detailed classification of them.

Keywords WSN · DHT · Routing protocols · Data management

1 Introduction

WSN [1–3] is constituted of small, battery-powered sensors having the ability to accomplish sensing, wireless communication and computation tasks. These sensors are densely deployed on the supervised zone. Their main function is to collect, disseminate data about the supervised phenomena and to collaborate in order to perform a common task.

In the last recent years, WSN become a suitable solution for a tremendous number of applications such as health monitoring, smart agriculture, intrusion detection applications and industrial control [4–6].

In spite of the intensive researches in this domain, WSN still face a lot of challenges such as energy optimization, fault tolerance, security, network robustness and scalability [7–9]. The energy optimization remains the most important challenge since the lifetime of a sensor depends on the battery's critical amount of energy.

Intensive researches in WSN have led to a big progress in the sensors design and features. This progress includes essentially sensors memory, CPU performance, communication bandwidth and range. This hardware improvement is accompanied with significant reduction in sensors costs.

These new trends present a double edged weapon. On one side, sophisticated sensors reduce the problem of constrained and scarce sensor resources which have a good impact on the performance of the entire WSN. This allows WSN to be widely disseminated and increasingly popular. Deployed applications on WSN become more important.

On the other side, such breakthrough poses new challenges: WSN are no more constituted of thousands of sensors but of a sheer number of sensors. Data that should be stored, treated and harvested are reaching multiple Gigabytes. All these new factors pose different problems: How to handle such progress? How to ensure efficient and timely data lookup? What are the most suitable routing protocols in such cases? Is there any preferred structure in WSN to deploy more advanced applications?

Many researches argue that combining the Distributed Hash Tables (DHT) to WSN can be a promising solution to solve all the mentioned problems. They think so because

G. Fersi (✉) · W. Louati · M. Ben Jemaa
Research Unit of Development and Control of Distributed Applications (ReDCAD), Department of Computer Science and Applied Mathematics, National School of Engineers of Sfax, University of Sfax, BP 1173-3038, Sfax, Tunisia
e-mail: fersi.ghofrane@gmail.com

DHT [10] (such as Chord [11], Pastry [12], CAN [13], etc.) have encountered revolutionary effect by providing an efficient storage service. They manage successfully the development of totally decentralized, scalable and self-organized applications. Thus, WSN would take benefit from DHT. However such combination between a wireless network and a system initially conceived for wired network brings certainly new obstacles. Different approaches have been proposed in order to apply DHT over WSN in a way that takes the advantages of such combination and avoids at most the drawbacks that can occur from this mapping.

In this paper, we discuss the applicability of DHT in WSN and present the DHT-based data management and routing solutions with a detailed classification.

The remainder of this paper is structured as follows. Sections 2 and 3 give an overview of WSN and DHT respectively. Section 4 specifies the different cons and pros of mapping DHT over WSN. Section 5 presents a taxonomy of DHT-based data management and routing protocols. Section 6 highlights existing data management and routing protocols that combine DHT to WSN. The similarities, the differences and the limits of these protocols are discussed in Sect. 7. Finally, Sect. 8 concludes the survey.

2 Wireless sensor networks

Wireless Sensor Networks consist of tens to thousands tiny wireless sensors and few base stations as depicted in Fig. 1. Sensors are able to do sensing, processing and communication tasks wirelessly. In WSN, sensors cooperate in order to supervise a given phenomenon, collect data and send it to a base station. There are two ways to deploy WSN:

- Deterministic deployment: Sensors are placed manually so that data can be routed through predetermined paths.
- Auto-deployment: Sensors are scattered randomly in a geographic zone. In this case, these sensors should themselves find the adequate communication paths and also adapt themselves to the formed distribution.

WSN have miscellaneous types of applications. They are widely deployed for environmental monitoring, medical studies, military and industrial surveillance as well as in emergency cases. WSN face multiple challenges. The most important issue is energy optimization since sensors are powered with energy constrained batteries. WSN must achieve all their functionalities without consuming important amount of energy to avoid sensors energy depletion and to extend the network lifetime. One of the most important sources of energy consumption is routing protocols. Also, bandwidth in WSN is so limited that large amount of data cannot be routed through these networks.

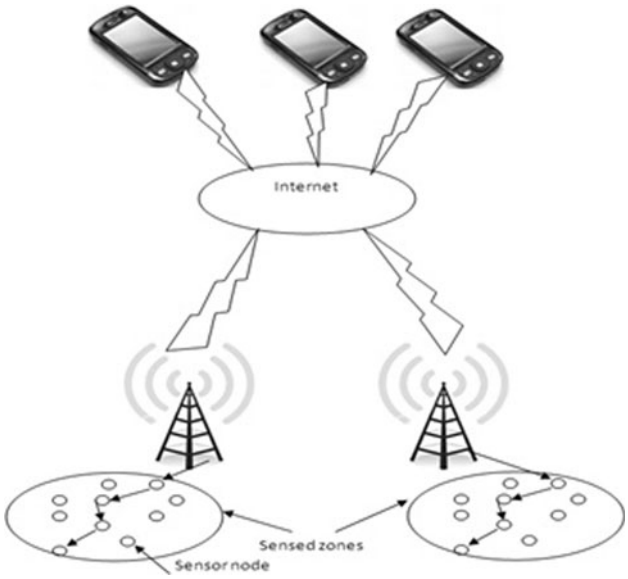


Fig. 1 Wireless sensor networks

Different routing and data management approaches have been proposed to cope with these different constraints.

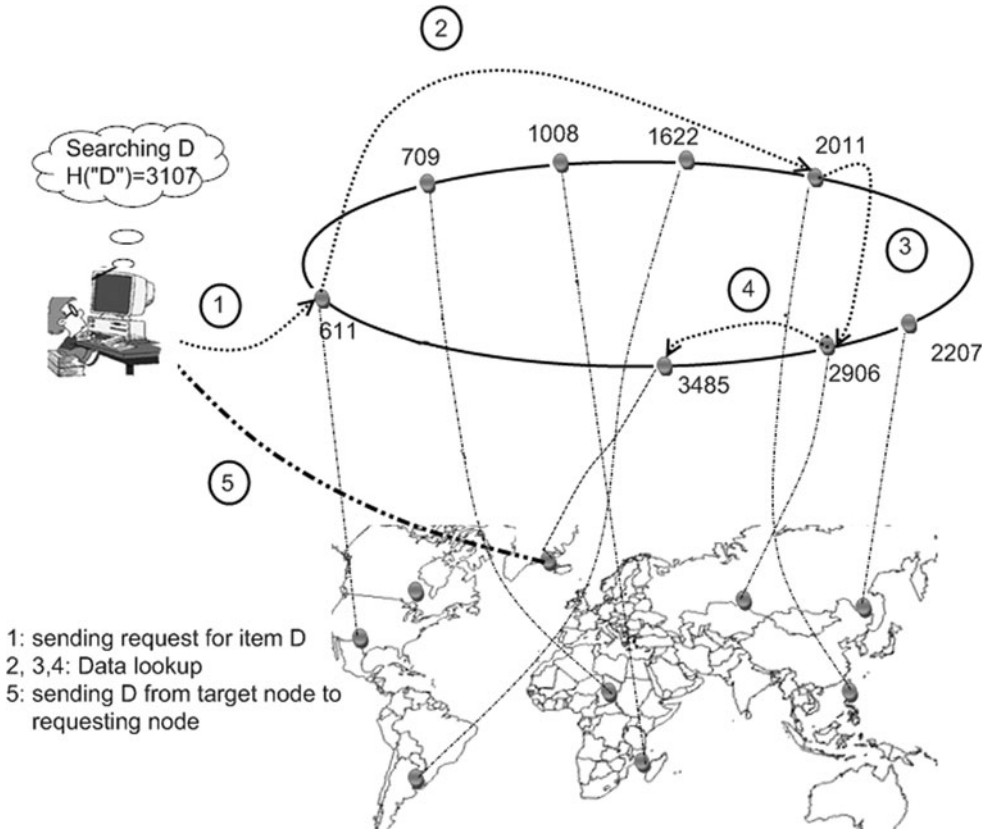
Routing protocols can be classified as follows. Data centric protocols [14–17] use advertisements before data transmission in order to avoid redundancy. Hierarchical protocols [18–22] classify nodes into clusters and perform large hops in each cluster. Cluster heads perform some message aggregation in order to save energy. Location-based protocols [23, 24] use the position information in order to forward data to the final destination. Identity-based protocols [25–29] ensure routing on the identities themselves without having location information in the packet header. DHT-based routing protocols is a sub-class of the identity-based protocols.

3 Distributed Hash tables

DHT is a popular decentralized distributed system. The main advantage of DHT is its efficient lookup service. Data in DHT are organized into (key, value) pairs. The keys are the products of a hash function that should balance the keys distribution in the whole network. Each node in DHT has a unique identifier. These identifiers belong to the same output space of the used hash function. In order to retrieve data of a certain key, it is useful to know the node that stores this key. So, first of all, the searching node hashes the key, then it routes the query to the node having the closest identifier to the key hashing. Figure 2 redrawn from [10] explains this procedure.

If a new node (having the identifier A) wants to join the DHT system, it sends a join message to an existing node. This message will be forwarded from a node to another

Fig. 2 DHT system [10]



until reaching the node master of the identifier A. At the reception of this message, this latter node updates its neighbors list and shares its own space as well as its own keys with the new neighbor.

Below, we give a short overview of two DHT systems: Chord [11] and Pastry [12].

3.1 Chord

One of the most famous DHT systems is Chord [11]. As depicted in Fig. 3, nodes as well as keys are placed on a ring. The hash function produces an m-bit identifier for both nodes and keys. The node's identifier is the SHA-1 hash of the node's IP address. The key is the SHA-1 hash of the data identifier. A key-value pair is assigned to the first node whose identifier is equal or follows the identifier of the key. This node is called successor (k). In order to ensure lookup efficiency, each Chord node maintains a finger table. This finger table contains at most m entries. The ith entry of a peer N, contains the identity of successor $n + 2^{i-1}$, where $1 \leq i \leq m$. When a node A wants to find the successor of a given key, it picks from its finger table the closest node to that key and forwards the query to it. This process is repeated until finding the responsible node. In each step in the lookup, the remaining distance is halved. Hence after $O(\log N)$ nodes, Chord finds the successor of the searched key in a N-node network.

When a node A joins the network, it initializes its finger table, its successor and predecessor. Fingers and predecessors of existing nodes are also updated in order to reflect the node joining.

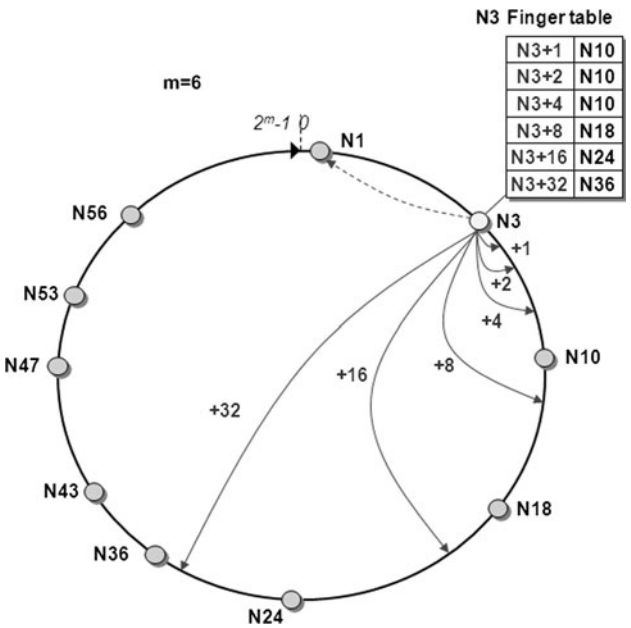


Fig. 3 Chord system [30]

When a node A goes down, its successor and predecessor will be aware of this departure, using their ping function. They update their successor and predecessor pointers and all A's keys are assigned to A's successor.

3.2 Pastry

Each node in Pastry [12] has a random 128-bit identifier. This identifier can be obtained by the SHA-1 hash of the node's public key or its IP address. Nodes identifiers are uniformly distributed in the 128-bit node identifier space. All nodes are organized into a logical circle which ranges from 0 to $2^{128} - 1$ bit. Each Pastry node maintains three tables:

- **Routing table:** has $o(\log_{2^b} N)$ rows, where b is a configuration parameter having as typical value 4. In each row i, there is i matching prefix with the current node.
- **Neighborhood set M:** contains the nodes identifiers and IP addresses of the most proximity closest $|M|$ nodes to the current node. The typical value of M is $2*2^b$
- **Leafset table:** maintains information about $|L|/2$ numerical successors and $|L|/2$ numerical predecessors of the current node. The typical value of L is 2^b.

An example of Pastry tables is given in Fig. 4 [12].

When a node receives a message, it first refers to its leafset table. If the key is covered in the leafset table, it forwards the message to the closest node for the given key. Otherwise, it picks from its routing table the node having the largest common prefix with the said key. The expected number of hops to reach the destination is $\log_{2^b} N$.

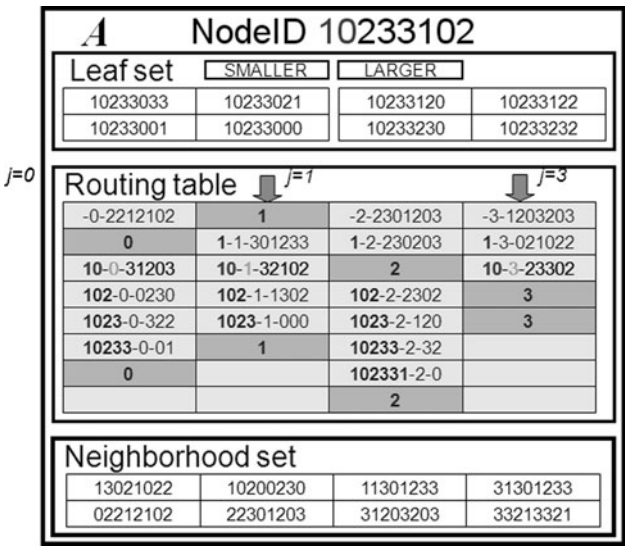


Fig. 4 Pastry routing table [12]

When a node having the identifier A joins the network, it first initializes its tables. To this aim, the new node sends a joining message to an existing node B. This message is routed until reaching the node C having the closest identifier to the node A. Nodes B, C as well as all the intermediate nodes that forwarded this joining message, send their state tables to the node A. In order to initialize the node A's neighborhood set, the node B sends its neighborhood set to the node A. Similarly, the node C helps to initialize node A's leafset by sending to the node A its leafset table. The joining cost in pastry is $\log_{2^b} N$.

The node failure or departure in Pastry can be detected when its neighboring nodes cannot communicate with it. In the routing table, this failure can be detected when the contacted node does not respond in a given period of time. All state tables are updated to replace the failed node.

In order to efficiently handle churns, [31] propose an improvement of Pastry. Bamboo has a similar design as Pastry. The only difference is the way they update the network. In Pastry this update is ensured when there are some changes in the network. Whereas in Bamboo update is ensured periodically by using proactive management traffic. The identifier space in Bamboo is 160-bit identifiers. Each node chooses randomly an identifier by hashing the IP or MAC address.

Bamboo ensures periodic recovery in order to handle failure. This means that Bamboo updates Leafset tables periodically. Nodes detect failure when they do not receive a response upon a predefined period of time.

4 Benefits and drawbacks of employing DHT over WSN

4.1 Benefits

Applying DHTs over WSNs can bring a lot of advantages to these networks.

In WSN, there are thousands of sensors that send a huge amount of messages simultaneously which should be handled and treated efficiently. When a base station wants to get information, it should have an efficient and a real-time response. DHT is able to perform an efficient lookup in a scalable and reliable manner.

Most of WSN are heavily deployed in emergency cases such as natural disasters and battlefields. In such cases, sensors are scattered from a plane and consequently, they have no information about their locations. This means that the network in these situations is auto-deployed. Consequently, proposed routing and data management protocols should ensure their functionnalities without worrying about the physical location of the participating nodes.

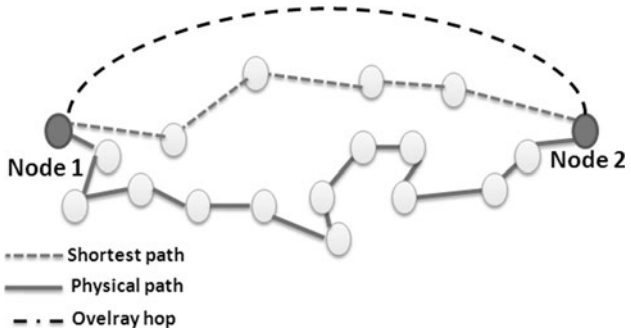


Fig. 5 Overlay vs. physical routing

Meanwhile, DHT protocols are based on nodes identifiers, which have no location semantic.

Furthermore, there are different types of applications that are built successfully thanks to DHT systems (file and resource sharing, P2P web search engine). These applications are also useful in WSN. Hence applying DHT to ensure these applications would be beneficial to WSN.

4.2 Drawbacks

There are some negative points that make the mapping of DHT over WSN a real challenge. Conventional DHT are ill-suited for WSNs. DHT function at application level for wired Internet. This means that DHT do not consider the physical structure of the underlying network. Indeed, two overlay neighbors are not necessarily two physical neighbors. An overlay hop can hence cost a lot of unnecessary physical hops as depicted in Fig. 5.

DHT routing tables should be always updated to avoid message misrouting and to ensure low path stretch. Thus, a lot of messages are sent in order to know which nodes are still available and which ones resigned. However, a big traffic of signaling messages degrades significantly the energy of sensors which leads to a dramatic reduce of the network lifetime. In addition, this maintenance traffic causes further network instability and reduces drastically the packet delivery ratio.

5 Taxonomy of DHT-based data management and routing protocols

Because of the diversity of DHT based routing protocols in WSN, the need of a general taxonomy becomes a necessity. We present a classification of these routing protocols according to several criteria reflecting fundamental design and implementation choices:

- **Structure**: Is there any distinction between the different nodes?
- **Design options of DHT combination to WSN**: How DHT is combined to WSN? Is DHT applied on the application layer and lying atop another routing protocol? Is it integrated or cross layered with the routing protocol?
- **Identifier assignment**: How the node obtains its identifier? Randomly? Depending on its physical neighbors' identifiers? Do hierarchical routing protocols assign nodes identifiers hierarchically?
- **Routing**: How the message is routed? Is source route employed in routing? Are there any constraints that should be taken into account before forwarding a message to a given node? At the case of hierarchical routing protocols, what is the relationship between master nodes and slave nodes? Is a slave node totally dependent to its master node? Can a slave node route a message to the destination or only master nodes have this ability?
- **Auto-deployment**: Can DHT-based data management and/or routing protocols be applied in the case of network auto-deployment? If yes how do they bootstrap?
- **Asymmetric link detection and utilization**: Do routing protocols detect link asymmetry that can happen in WSN? If so, is such asymmetry taken into account in routing?
- **Master nodes selection**: In hierarchical routing protocols, how master nodes are selected? Does this selection depend on nodes' features?
- **Master nodes rotation**: Do master nodes change over the time? If yes, what is the approach used to choose master nodes in each round?

5.1 Structure

In **flat DHT-based protocols**, the network is not organized into a particular structure. All nodes are considered uniform in resources. Each node plays the same role as other nodes. Flat routing is suitable in simple systems, where the network is small. When the number of sensors increases, flat routing does not scale. It cannot avoid nodes having critical amount of energy. This leads quickly to several nodes failure.

Hierarchical DHT-based protocols are required in complex systems where the network sensors are heterogeneous. They are also needed to accommodate and handle efficiently nodes failure. **Hierarchical DHT-based protocols** exploit nodes heterogeneity by assigning supplementary roles to the most powerful nodes in the network. Protocols belonging to this class, organize the nodes in a hierarchical manner. This structure is useful to optimize the energy consumption by condensing data traffic in super nodes (cluster heads) and performing large routing hops.

5.2 Design options of DHT combination to WSN

Design options of DHT combined to WSN can be classified into three classes: layered, cross-layered and integrated[1].

The first contributions that combined DHT to WSN, applied DHT at the top of the application layer and used a MANET routing protocol at the network layer. This combination is called **layered approach**.

Other protocols use a **cross-layered** approach. In this mapping design, the DHT system acts also at the application layer. However, contrarily to the layered approach, DHT interacts with the routing protocol. This means that there is a permanent communication between the application and the network layers. For example, the nodes identifiers needed in DHT data management are assigned depending on the nodes positions in order to facilitate the routing process: two virtual neighbors are also physical neighbors. Such combination reduces the implementation complexity of DHT combination to WSN.

In the **integrated** approach, DHT is directly and fully integrated with the routing process on the network layer or the Medium Access Control process on the link layer. In such a case, the nodes identifiers are assigned independently of any information related to other layers. These identifiers can be the hash of the nodes' IP addresses or the nodes' MAC addresses. This design simplifies significantly the mapping of DHT over WSN and gives the opportunity to have a significant cooperation and optimization between DHT and the routing protocol which improve drastically the network performance.

5.3 Identifier assignment

In DHT-based protocols, each node should have a unique identifier. In several protocols, a node identifier can be randomly assigned, or be the hash of the node MAC address.

In other protocols, physical neighbors are also virtual neighbors. This **virtual relative position based** technique avoids overhead resource usage caused by protocols that rely on under layer routing techniques.

Some hierarchical protocols use **hierarchical identifier assignment** so that each node has at a given level, a level-related identifier. The advantage of this method is that the node identifier reflects the level to which it belongs. However, hierarchical identifiers assignment process is more complex than that of flat identifiers.

5.4 Routing

DHT-based routing protocols can be classified according to their routing strategy. Some routing protocols employ **source routing**. Others are **location-based**. Some of DHT-based routing protocols employ at each next hop, the **neighbor selection**.

When greedy routing is adopted, it can be classified into physical and virtual greedy routing. **Physical greedy routing protocols** elect at each forwarding step, the locally optimal choice. **Virtual greedy routing protocols**, elect at each virtual hop, the next physical hop having a path to the nearest virtual hop to the destination node.

At the case of hierarchical DHT-based routing protocols, routing depends on the relationship between master and slave nodes. When a slave node is **totally dependent** to its corresponding master node, its routing role is very limited. It only forwards the query to its corresponding master node when it is not the final destination of this query. This method is beneficial for slave nodes having critical amount of energy and limited computational abilities. However, concentrating all traffic in master nodes would quickly decrease their amount of energy and can cause the bottlenecks.

In other hierarchical DHT based routing protocols, slave nodes are **partially dependent** to master nodes. In this case, master nodes are more solicited than slave nodes in routing. However, this does not eliminate slave nodes routing abilities. Slave nodes have routing tables that contain information about other nodes. This information enables them to route messages. This routing strategy saves energy in slave nodes without causing bottlenecks in master nodes.

5.5 Auto-deployment

Some DHT-based protocols present a flexible solution that allow them to be employed successfully in network auto-deployment cases. Others require a so rigid structure that disable them to be auto-deployed e.g some protocols give constraints at node placement in order to ensure good routing and/or data management performance.

5.6 Asymmetric link detection and utilization

Asymmetric links occur frequently in WSN due different causes such as transmission power disparity, interference and physical obstacles. Some DHT-based protocols detect and take advantage of this asymmetry by using it to enhance routing performances. Other protocols assume that all links are symmetric.

5.7 Master nodes selection

There are different ways to select master nodes in hierarchical DHT-based routing and data management protocols. **Nodes features-aware selection**, takes into account nodes features when choosing master nodes e.g nodes having important amount of energy are selected as master nodes

[1] The same classification has been proposed in [32] for DHT over MANET.

(cluster heads). In other routing and data management protocols, the selection does not take into account nodes characteristics, and master nodes are chosen randomly. We call this method **Master nodes random selection**.

5.8 Master nodes rotation

There are hierarchical routing and data management protocols that choose master nodes only one time. This means that once master nodes are chosen, the network backbone remains static. This avoids the overhead that can be caused in master nodes rotation. The main drawback of this technique is that master nodes die quickly if they do not have access to power.

In other routing and data management protocols, master nodes do rotations in a round-robin fashion for example. This rotation takes place after consuming a given quantum of energy. The immediate successor is elected to be the new master node. Doing master nodes rotation is benefic since it avoids the failure of master nodes. Nevertheless, the mentioned rotation condition has some limits. In some cases, even if the current cluster head consumed an important amount of energy, it can be still much more powerful than the new cluster head. Thus, a node having a critical amount of energy can be elected to be cluster head leading to its quick failure.

6 DHT-based data management and routing protocols in WSN

6.1 Pastry layered to DSR

In the layered design of [33], Pastry [12] is integrated directly atop DSR [34] routing protocol.

6.1.1 Identifier assignment

Identifier assignment is similar to Pastry [12].

6.1.2 Routing

In the layered approach, entries in routing and leafset tables are maintained in Pastry without source routes. The former routes are maintained by DSR [34] on the demand of Pastry routing state. Some modifications are applied to Pastry as well as DSR in order to ensure that the combination between the two protocols will not be expensive. Authors in [33] added to Pastry a hop count metric for proximity. DSR is modified in order to export an API that allows Pastry to have information about the proximity values for nodes it is interested in. When Pastry pings a node, DSR can use its cache to reply if there is a cached path to the node being pinged. If such path does not exist a ROUTE REQUEST is initiated by DSR. The routing cost of Pastry combined to DSR is $O(N * \log_{2^b} N)$.

6.1.3 Node joining and failure

Generally speaking, the nodes joining and departure are similar to Pastry. However, the joining procedure of Pastry is slightly modified in order to expand the ring search for locating a bootstrap node to join the network.

6.2 Chord layered to MANET routing protocols

The use of Chord [11] as an overlay network over MANET routing protocols such as OLSR [35], DSR [34] and AODV [36] is discussed in [37]. Node identifier assignment, node joining, departure as well as routing are done similarly to Chord. From a Chord peer to another the MANET routing protocol is employed. The routing cost is $O(N*\log N)$.

6.3 Bamboo layered to AODV-UU

The approach in [38] applies Bamboo [31] on top of AODV-UU [39] routing protocol.

The DHT overlay presents the virtual structure of the network.

6.3.1 Identifier assignment

Identifier assignment is done similarly to Bamboo [31].

6.3.2 Routing

Routing is ensured by Bamboo. To forward the Bamboo message from a peer to another, AODV-UU is employed. The routing algorithm complexity is $O(N*\log N)$.

6.3.3 Node joining and failure

Nodes joining and failure are handled by Bamboo. From a peer to another, AODV-UU is employed. The joining cost is $O(N*\log N)$.

6.4 Topology-based distributed Hash tables (T-DHT)

T-DHT [26] extends the functionalities of Content Addressable Network (CAN) [13] in order to be applied in WSN. It offers data centric storage as well as routing.

6.4.1 Identifier assignment

At the network startup three or more reference nodes are chosen randomly. These reference nodes flood the network with the distance between each others. The other nodes

triangulate their virtual position depending on the hop distance from each node to these reference nodes.

6.4.2 Routing

Each T-DHT node maintains in its routing table, paths to its neighbors in the coordinate system, the area each neighbor maintains and routes to some non neighbors nodes (used as "short-cuts"). The average size of this routing table is *4*. The average path length is $O(0.5 * \sqrt{N})$. T-DHT nodes forward messages to destinations by greedy forwarding.

6.4.3 Node joining

If a node wants to join the network, it only needs to find an existing node in the network. By routing a message containing its identifier, the joining node finds the zone that is responsible of its position. Whenever this zone is reached, it is split equally between the two nodes. After that, the joining node, notifies its neighboring nodes of its presence. The average cost of the joining procedure is 4.

6.5 Virtual cord protocol (VCP)

Virtual cord protocol [27, 40] is a virtual location based and DHT-like protocol. In VCP, each node has the same physical and virtual neighbors.

6.5.1 Identifier assignment

Each VCP node at its joining phase exchanges HELLO messages in order to know its physical neighbors. It computes its virtual identifier depending on the identifiers of its physical neighbors so that it becomes also a virtual neighbor to them.

6.5.2 Routing

In VCP, a node's routing table maintains only information about its physical neighbors; hence its size is m, where m is the number of physical neighbors. Each entry in the routing table only needs HELLO messages to be updated. The communication overhead to realize this update is m.

When a node receives a data packet, it employs a greedy routing and forwards the packet to the physical neighbor having the closest identifier to the destination. The path length in VCP is $\sqrt{N} - 1$ where N is the total number of nodes in the network.

6.5.3 Node joining and failure

When a node joins the network, it discovers its physical neighbors by broadcasting periodically HELLO messages. This enables the joining node to compute its identifier and to position itself in the cord.

The failure of a node is detected when its physical neighbors do not receive from it HELLO messages along a given period. The identifiers of the other nodes on the vicinity of the failed node should be updated. In order to avoid data loss in the case of node failure, data replication in the successors and predecessors are recommended.

The overhead cost of network stabilization after node joining or failure is m.

6.6 ScatterPastry

ScatterPastry [28] employs Pastry [12] in the ScatterWeb [41] platform. ScatterWeb is a research project for WSN. It is made of sensor nodes and e-Gates.

6.6.1 Identifier assignment

Every ScatterPastry node generates a unique 16 bit overlay identifier. Having such limited number of bits, increases the probability of having nodes' identifiers collisions especially when the number of nodes exceeds 200 nodes. To solve this problem, authors in [28] suggested the use of a name server in order to assign a unique identifier to each node.

6.6.2 Routing

ScatterPastry nodes use two tables for routing. The Leafset table maintains four successors and four predecessors. The routing table contains in each row i, i matching prefix with the current node. The leafset table size in ScatterPastry is $2b$, where b is a configuration parameter having as typical value 4. The average routing table size is $o(\log_{2^b} N)$.

Routing in ScatterPastry is similar to Pastry. Authors in [28] state that ScatterPastry can be integrated to the Destination-Sequenced Distance Vector (DSDV) [42] or implemented over this latter. The average lookup cost is $o(\log_{2^b} N)$.

6.6.3 Node joining and failure

Nodes joining and failure are handled similarly to Pastry.

6.7 ScatterDHT

6.7.1 Identifier assignment

Each ScatterDHT [29] node has a unique identifier that should be in the range $(0, 2^{16} - 1)$. This identifier is obtained by a specific function of ScatterDHT protocol.

6.7.2 Routing

ScatterDHT [29] associates at each node a routing table used to forward messages to the adequate physical neighbor. To route a message, each intermediate node forwards the message to the node having the numerically closest identifier to the given key. When the node notices that it is the closest one to this key, it realizes that it is the destination and receives the packet. Otherwise, if the node that receives the packet has an empty routing table, it rebroadcasts the packet to its physical neighbors. Authors in [29] argue that the estimated query path length in ScatterDHT is $O(\log N)$. The ScatterDHT routing table size is $o(\log_{2^b} N)$.

6.7.3 Node joining and failure

After obtaining its identifier, the joining node builds its routing table that has as maximum number of columns, the number of bits of its identifier and as maximum number of rows, the number of hex digits of every node identifier. To find the node having the closest identifier to the joining node, the former sends a JOIN message. This message is forwarded from a node to another with verifying at each step, that the next hop's identifier has the largest common prefix with the joining node. This procedure is repeated until reaching the requested node. The number of control messages needed to realize the join phase is $O(\log_{2^b} N)$ where N is the number of nodes in the network, b is a constant equal to 4.

Physical neighbors realize the failure of a node when they do not receive from it HELLO messages during a prefixed period. Farther nodes become aware of this failure, when they try to use it in order to route a message and they do not get a response during a given time. These nodes remove the failed node from their routing tables.

6.8 Virtual ring routing (VRR)

Virtual ring routing [43] is a scalable identity-based routing protocol. Each node in VRR has a location-independent identifier.

6.8.1 Identifier assignment

Virtual ring routing does not impose a generic way to generate nodes identifiers so that they can suite the conceived application. [43] states that this identifier can be the hash of the node's public key (160-bit SHA-1) in order to ensure security or a random 32-bit integer in order to provide backwards compatibility with IPv4 addresses. Nodes have location-independent identifiers and are organized into a virtual ring ordered by their identifiers.

6.8.2 Routing

Each node in VRR [43] maintains in its routing table, a virtual neighbor set containing generally the node identifiers of two successors and two predecessors and a physical neighbor set containing the identifiers of the physical neighbors towards virtual neighbors. The average size of VRR routing table is $rp + k$ where r is the number of virtual neighbors, p is the average path length and k is the number of physical neighbors.

To route messages, VRR [43] chooses at each step, the node having the closest identifier to the destination, and forwards the message to its corresponding next physical hop. In [43], authors argue that a packet is expected to reach the destination after forwarding the message to $O(rp)$ intermediate nodes. The expected path length in VRR is estimated to be at most $O(\text{Diam}.\log N)$ [44] where N is the total number of nodes and Diam is the network diameter.

6.8.3 Node joining and failure

When a new node A joins the network, it finds its physical neighbors by exchanging HELLO messages. It sends through one of its physical neighbors (chosen as proxy) a setup request message with the identifier A. This message is forwarded until reaching the closest node to the node A (e.g the node B). This node responds the joining node by a setup message that contains its identifier as well as its virtual neighbors set.

The physical neighbors of the leaving node X will be aware of its absence when they do not receive from it HELLO message during a given period. They remove from their routing tables all virtual paths that have the node X as next physical hop and send teardown message in order to remove the complete virtual path from all the intermediate nodes.

Local virtual paths repair are also possible at the case of nodes' failure and movements. This can be accomplished by replacing only the failed link by another alternative route. The joining/departure cost in VRR is $O(\text{Diam}.\log N)$.

6.9 Scalable source routing (SSR)

Scalable source routing [45, 46] ensures indirect routing which means that the destination of a packet is abstract and the routing protocol maps it to a concrete node.

6.9.1 Identifier assignment

Nodes identifiers in SSR can be obtained by hashing their MAC addresses into the SSR address space. The elected hash function should map all the objects identifiers to nodes addresses.

6.9.2 Routing

Scalable source routing [45] packets contain source address, destination address and source route. The former is not essentially the total route from the source to the destination. It can only be a sub route that ends at an intermediate node. This intermediate node checks its local cache in order to search an appending route to the destination. If this search fails, it appends the source route to the closest node to the destination. To ensure network consistency, each node should have a source route to its virtual neighbors. SSR is performing its routing process in a logarithmically path length. The average size of the SSR source route is $O(\log N)$ [47] where N is the total nodes number in the network.

6.9.3 Node joining and failure

In the bootstrapping phase, when a new node N wants to join the network, it searches its physical neighbors. If its physical neighbor set is not empty, it chooses from this set the closest predecessor or the closest successor (e.g node Pi) and sends to it a message that indicates that the new node is considering it as the closest predecessor or successor. At the reception of this message, the node Pi checks if its routing table has a source route to a node N' having a closer identifier to it. If such route exists, the node Pi sends to the joining node N, a message having the source route to the node N'. This procedure is continued until reaching the closest successor or predecessor to the joining node. This joining procedure converges in $O(\log N)$, where N is the total number of nodes in the network.

When an intermediate node notifies that the next hop is no more available, it informs the node that recently appended the failed node into the source route by sending SSR-ROUTE-UPDATE message containing the broken link.

6.10 Geographic Hash table (GHT)

In GHT [48], keys are hashed into a geographic coordinates. Each key-value is mapped to the node having the geographically closest location to the hash of its corresponding key.

6.10.1 Routing

GHT does not use nodes identifiers in routing, but instead, it uses a geographic routing based on nodes locations. GHT is built atop Greedy Perimeter Stateless Routing for Wireless Networks (GPSR) [49]. It uses the former to select the closest node to the destination. This node is called home node. Data are first stored in this node and other copies are stored in the perimeter nodes in order to recuperate the stored data at the case of nodes failure. The routing cost in GHT is $O(\sqrt{N})$. Each node stores in its routing table information about all its physical neighbors. The average size of this routing table is m, where m is the number of physical neighbors.

6.10.2 Node joining and failure

When a node A notices the presence of a new neighbor B, it sends to it all the events entries stored in its local database, which are closer to the neighbor B. The joining cost is then m.

When a node C realizes that the home node left the network, it starts a refresh procedure in order to choose a new home node. At the end of this procedure, one of the perimeter nodes becomes the new home node and this latter recruits replicas.

6.11 Tiered Chord (TChord)

Tiered Chord [50] is a simplified mapping of Chord [11] on WSN.

6.11.1 Identifier assignment

Nodes identifiers are obtained by hashing the unique address of each sensor (for example the MAC address) using SHA-1. Each ring should contain at least one high-powered master node.

6.11.2 Routing

Tiered Chord [50] architecture is inspired by Tenet Architecture [51] for Tiered Sensor Networks. Master nodes are organized into a virtual ring and all messages are routed clockwise.

Each master node stores information about all its slave nodes in addition to $O(\log N)$ other master nodes in its local finger table. Hence the average size of the TChord master node finger table is $O(\log N)$. Slave nodes have no information about their neighbors. And if one of the slave nodes receives a query, it first checks its own table, if it does not find the searched data; it simply forwards the query to its corresponding master node. At the reception of a query, the master node checks if this query belongs to one of its slave nodes. If so, it sends the query to it. Otherwise, it picks from its finger table, the closest master node and forwards the query to it. [50] affirms that this procedure ensures that queries are all solved in $O(\log N)$ messages.

6.11.3 Cluster heads rotation

There is no master nodes rotation in TChord. Nodes with better hardware (Masters) will be placed close to normal

nodes. These master nodes can even have access to power e.g. in buildings. Because of the physical location constraint, it does not make much sense to rotate masters.

6.11.4 Node joining and failure

When a master node wants to join the network, the successor pointer should be updated. Similarly, after a master node leaving, these successor pointers are updated. Data in each master node are replicated in the neighboring master nodes in order to avoid data loss at the case of node failure. Like Chord, the number of messages needed to stabilize a N-node network after node joining or departure is $O(\log N)$.

6.12 SNBDT protocol

SNBDT [52] is the application of NBDT [53] onto WSN. In SNBDT, nodes are organized into a hierarchical tree according to their amount of energy. Energy powerful nodes are considered as master nodes. Each master node in level i maintains information about its slave nodes in addition to $2^{2^{i-1}}$ other master nodes.

6.12.1 Identifier assignment

In SNBDT [52], each node obtains its identifier by hashing its unique address.

6.12.2 Routing

Each master node maintains in its routing table information about all its slave nodes in addition to $2^{2^{i-1}}$ other master nodes. Hence, the size of the SNBDT routing table is $O(2^{2^{i-1}})$. Queries are treated essentially by master nodes. At the reception of a query, the master node checks its own keys, if the search is not successful, then the master node picks the closest master node from its finger table and forwards the query to it.

When a slave node receives a query, it only checks its own key and if the search is not successful, it sends the query to its corresponding master node. All queries are resolved in $O(\log \log N)$ messages, where N is the total number of nodes in the network.

6.12.3 Cluster heads rotation

Peers do not apply any rotation methods. Only associated sensors to these peers should re-associate themselves to the adequate peers when their amount of energy changes.

6.12.4 Node joining and failure

Nodes joins and failures are treated in a similar way as in [53]. At the joining phase, the new node sends a JOIN request message. This message is forwarded until reaching the last node. The SNBDT [52] joining cost is $O(1)$.

When a node fails, it is deleted from the tree. If the tree becomes unorganized after this node failure, it should be rebuilt.

6.13 Coral-based VRR

Coral-based VRR [54] is an energy-aware routing protocol based on location independent identifiers. It takes into account nodes heterogeneity and adopts a hierarchical DHT in the VRR [43] routing. Sensor nodes are classified according to their available energy.

6.13.1 Identifier assignment

In Coral-based VRR [54], the node identifier can be the hash of the node's public key or a random 32-bit integer.

6.13.2 Routing

When a packet is received by a regular peer, the packet is forwarded using the VRR [43] forwarding algorithm, until finding the destination identifier or finding a superpeer. In the second case, the packet is routed in the superpeers ring using the VRR forwarding algorithm, until to find the destination identifier, or to reach the closest superpeer in the superpeers ring. In the second case, the packet descends to the regular peers level, and is routed until reaching the destination identifier. The average path length is O (Diam.log M) where M is the total number of superpeers in the network and Diam is the network diameter. The average size of the routing table is $O((r1 + r2)\ p + k)$ where $r1$ and $r2$ are respectively the number of virtual neighbors peers and the number of virtual neighbors superpeers, p is the average path length and k is the number of physical neighbors.

6.13.3 Superpeers dynamic state

Since superpeers are more solicited than other nodes in Coral-based VRR, their amounts of energy will be degraded rapidly and reach the peer threshold. In this case, a superpeer leaves the superpeers ring and will belong only to the regular peers ring. Hence, after a while, all superpeers will disappear and all nodes become regular peers. The advantage here is that the energies of the most powerful nodes have been exploited before their leaving.

6.13.4 Node joining and failure

When a node joins the network, it sets its type (a peer, a superpeer) according to a superpeer-specific threshold. If the joining node is a regular peer, it should be positioned only in the first ring (peers ring). But if the node is a superpeer, it should be positioned in the peers ring and the superpeers ring.

In Coral-based VRR [54], the failure detection and repair can be performed in a similar way as in VRR. The routing of the VRR signaling messages is performed by the proposed Coral-based VRR routing process.

6.14 Chord based network protocol for serving efficient queries in WSN (C2WSN)

In C2WSN [55], nodes are organized into two rings depending on their amount of energy. The two rings apply Chord based DHT protocol. The inner ring contains cluster heads. The outer ring includes all nodes. These nodes are split into arcs. Each cluster is responsible of an arc of this outer ring.

6.14.1 Identifier assignment

In C2WSN [55], nodes identifiers are assigned randomly.

6.14.2 Routing

When a node receives a request, it sends it to the closest cluster head. This cluster head checks its outer finger table in order to know if the requested key belongs to its cluster or not. If so, the cluster head returns the result to the requesting node. Otherwise, it picks from its inner layer cluster head finger table, the closest cluster head and forwards the message to it. The C2WSN path length is $O(\log M + \log \lambda i)$ where M is the number of clusters and λi is the maximal number of sensor nodes within a cluster i. The size of cluster head routing table is $O(\log M + \log \lambda i)$, while the size of common sensor node routing table is $O(\log \lambda i)$.

6.14.3 Cluster head rotation

In order to avoid the energy depletion of cluster heads, each cluster head applies a rotation mechanism if it has consumed the given quantity of energy $Eq = Em/\mu * r$, where Em is the maximum energy of the sensor, μ a constant and r is the number of rounds of rotations. The clockwise successor in the outer ring becomes the new cluster head.

6.14.4 Node joining and failure

When a node A joins the network, it is assumed that it learns the identity of an existing node B. The joining node A sends a message to B requesting it to find A's successor in the ring so that it can help it to build its finger table.

When the node leaves the network, the nodes that have A in their finger table, search A's successor. This is done by periodical check of successors and predecessors.

6.15 Chord for sensor networks (CSN)

Chord for sensor networks [56] has the same design as Chord. CSN network is made up of different layers. Each layer consists of multiple clusters. All cluster heads in the lower layer are considered as super sensors in the higher layer. In each cluster, nodes are classified into a logical ring and communicate with their geographically closest nodes to save energy.

6.15.1 Identifier assignment

Nodes in CSN [56] can be named in an incremental way during the cluster setup phase, in a bottom up manner. This naming can be also done in a parallel fashion, after the cluster formation. In this case, the naming starts at the highest layer and after that the naming process is triggered in all low-layer clusters. After the naming phase, sensor names are hashed using SHA-1 hash function in order to obtain nodes identifiers.

6.15.2 Routing

Chord for sensor networks [56] protocol has the ability to work on two modes based on the feature of the required application. In the energy efficient mode (EEmode), each node communicates with its closest physical neighbor node. The cost of lookup operation in EEmode is $O(M*\log N)$, where M is the maximum path length of EEmode + 1. In the Robust mode (Rmode), CSN behaves exactly like Chord. A CSN node can send a message to farther nodes so that it can realize an efficient data lookup $O(\log N)$ but with much more energy consumption. The CSN routing table size is $O(\log N)$.

6.15.3 Cluster heads rotation

To ensure network balancing, each cluster head that consumes an energy quantum $Eq = Em/\mu$ (where Em: maximum energy of a sensor, μ: a constant) chooses its clockwise successor in the ring as next cluster head.

6.15.4 Node joining and failure

In CSN [56], node joining can take place only if there is a backup node that is ready to join the network at the place of a node that fails voluntarily. In this case, the leaving node

sends all its information to the joining node. CSN does not address the case where a node fails suddenly. The CSN stabilization cost is $O(\log N)$.

7 Discussion

Table 1 illustrates a survey of current DHT-based routing and data management protocols in WSN based on their classifications. Table 2 gives an overview of the performance of each cited DHT-based protocol.

7.1 Path length

The integration of DHT to WSN is a challenging issue. One of the most important challenge is the unnecessary hops caused by the mapping of DHT to WSN. VCP [40], T-DHT [26] and CSN [56] solve this problem by assigning identifiers in such a way that two physical neighbors are also virtual neighbors. Both VRR [43] and Coral-based VRR [54] reduce the gap between physical and virtual routing by employing greedy hops.

However, T-DHT consumes a lot of energy in the identifier assignment phase since it uses message flooding. VCP [40] overcomes the T-DHT problem by the use of a novel identifier assignment strategy that avoids flooding. In addition to the naming strategy that ensures virtual and physical neighbors merging, CSN [56] offers the possibility to shorten the path at most by increasing the transmission power in such a way that lets two nodes placed so far away from each others communicate directly without any intermediate nodes. This option offers very low path stretch but consumes an important amount of energy.

Mediator nodes in SSR [45] can apply shortcuts that reduce the number of unnecessary hops. SSR is well suited to object tracking applications [57]. In [58], SSR has been further improved by choosing the most appropriate next physical hop from the list of physical neighbors. In order to accommodate SSR to link asymmetry, [59] propose to take advantage of this characteristic in the message routing. This improves routing efficiency and network connectivity.

Location information helps GHT [48] to reduce the problem of unnecessary hops, since at each step, it becomes physically closer to the destination. However, GHT does not route messages in an efficient way because it uses GPSR [49] routing protocol which does not use shortcuts. Besides, GHT uses a uniform hashing function that does not take into account neither sensor amounts of energy, nor real sensors distribution.

The hierarchical structure of Coral-based VRR [54] allows to perform big logical hops towards the destination. These large hops minimize significantly the average path length.

7.2 Data management

GHT [48] is the first DHT-based protocol that has focused on the data storage. In GHT, data are stored equitably in the same surfaces. These surfaces do not have the same number of sensors which leads to unbalanced networks. In order to enhance GHT storage performances, QNIGHT is proposed in [60]. This routing protocol uses non-uniform hashing functions. These functions scatter data approximately with the same distribution as sensors by using a similar strategy to the rejection method [61]. Generated data in [60] are considered different in their importance level. In fact, there are some data that are generated by multiple sensors (popular data) and other scarce data. Q-NIGHT differentiates between these types of data by the introduction of a new metric Q that expresses the desired quality of service.

Data management in VCP [27, 62] is combined to the routing concept. It is dependent to the node position. This ensures data storage and retrieval efficiency in the case of static networks. In the case of nodes mobility or failure, VCP [40] integrates data replication techniques in order to increase its reliability.

All the mentioned layered approaches ensure data management using existant DHT systems like Chord [11], Pastry [12] and Bamboo [31]. Since the range of sensors is very limited, data storage and retreival need the use of multiple intermediate nodes. The choice of these intermediate nodes is ensured by a MANET routing protocol.

7.3 Mobility support

Another challenging issue is the mobility support. It depends mainly on the node's identifier assignment strategy. In virtual relative position based protocols such as VCP, each node which moves from its position should change its identifier so that it becomes also a virtual neighbor for its new physical neighbors. In highly mobile domains, VCP [27, 63] uses a novel inter domain routing technique that uses Ant Colony Optimization (ACO) [64].

Location-based protocols such as GHT [48] can support low mobility degree. When a home node moves, its identifier changes and the perimeter nodes change their membership to the new home node using a refresh procedure.

Assigning position-independent nodes identifiers, enables the DHT-based protocols to handle efficiently the nodes mobility. For example in VRR [43], when a node moves, its virtual neighbors remain the same, but physical paths to these neighbors change. VRR aborts these paths and reconstruct new ones.

Table 1 DHT-based routing and data management protocols classification

DHT-based protocols	Strucure	Identifier assignment	Routing	Asymmetric link detection and utilization	Master nodes selection	Master nodes rotation	Auto-deployment	Design option
Pastry layered to DSR [33]	Flat	Random	DSR [34]	No	N/A (Not Applied)	N/A	No	Layered
Chord layered to MANET routing protocols [37]	Flat	Hash of the node's IP address	OLSR [35], DSR [34] and AODV [36]	No	N/A	N/A	No	Layered
Bamboo layered to AODV-UU [38]	Flat	Hash of the node's MAC address	AODV-UU [39]	No	N/A	N/A	No	Layered
T-DHT [26]	Flat	Virtual relative position based	Physical greedy routing	No	N/A	N/A	No	Cross-layered
VCP [40]	Flat	Virtual relative position based	Physical greedy routing	No	N/A	N/A	Yes	Cross-layered
ScatterPastry [28]	Flat	Name server	Physical greedy routing	No	N/A	N/A	Yes	Integrated
ScatterDHT [29]	Flat	Specific ScatterDHT function	Physical greedy routing	No	N/A	N/A	Yes	Integrated
VRR [43]	Flat	Hash of the node's public key	Virtual greedy routing	No	N/A	N/A	Yes	Integrated
SSR [45]	Flat	Hash of the node's MAC address	Source routing, neighbor selection	Yes	N/A	N/A	Yes	Integrated
GHT [48]	Flat	Random	Location based	No	N/A	N/A	No	Cross-layered
TChord [50], SNBDT [52]	Hierarchical	Hash of the node's MAC address	Slaves totally dependant to masters	No	Nodes features-aware	No	No	Integrated
Coral-based VRR [54]	Hierarchical	Hash of the node's public key	Virtual greedy routing, Slaves partially dependant to masters	No	Nodes features-aware	No	Yes	Integrated
C2WSN [55]	Hierarchical	Random	Slaves partially dependant to masters	No	Nodes features-aware	Yes	No	Integrated
CSN [56]	Hierarchical	Virtual relative position based, hierarchical	Slaves partially dependant to masters	No	Random selection	Yes	No	Cross-Layered

Table 2 DHT-based routing and data management protocols performance

DHT-based protocols	Path length	Routing table size	Joining and departure cost
Pastry layered to DSR [33]	$O(N * (\log_{2^b} N))$	$\log_{2^b} N$+ DSR routing table size	$O(N * \log_{2^b} N)$
Chord layered to MANET routing protocols [37]	$O(N$*log $N)$	log (N) + MANET protocol routing table size	$O(N_*$logN)
Bamboo layered to AODV-UU [38]	$O(N$*log $N)$	$\log_{2^b} N$ + AODV-UU routing table size	$O(N_*$logN)
T-DHT [26]	O(0.5*$\sqrt{N}$)	4	4
VCP [40]	$\sqrt{(N)} - 1$	m	m
ScatterPastry [28]	$o(\log_{2^b} N)$	$o(\log_{2^b} N)$	$o(\log_{2^b} N)$
ScatterDHT [29]	$O(\log\ N)$	$o(\log_{2^b} N)$	$o(\log_{2^b} N)$
VRR [43]	O (Diam.log N)	O $(rp + k)$	O (Diam.log N)
SSR [45]	$O(\log N)$	$O(\log\ N)$	$O(\log\ N)$
GHT [48]	$O(\sqrt{N})$	m	m
TChord [50]	$O(\log N)$	$O(\log\ N)$	$O(\log\ N)$
SNBDT [52]	$O(\log \log N)$	O $(2^{2^{i-1}})$	O (1)
Coral-based VRR [54]	O(Diam.log M)	O $((r1 + r2)p + k)$	O (Diam.log N/log M)
C2WSN [55]	$O(\log M\ + \log \lambda i)$	$O(\log M + \log \lambda i)$	$O(\log^2 \lambda_i)$
CSN [56]	Robust mode $O(\log N)$ EE mode $O(M$*log $N)$	$O(\log N)$	$O(\log N)$

7.4 Energy awareness

Energy optimization is also treated by some DHT-based routing protocols. The hierarchical topology of CSN [56], TChord [50], SNBDT [52] and Coral-based VRR [54] optimizes energy since they condensate data traffic in some nodes while avoiding weak nodes and they offer also efficient routing. However, in TChord [50] and SNBDT [52] slave nodes should be placed near to master nodes. Due to these constraints, these routing protocols can not be applied in auto deployment cases. Also, the cluster head rotation method in C2WSN [55] and CSN [56] does not matter of the remaining energy of the new cluster head which can cause its quick failure.

7.5 Bootstrapping

Bootstrapping is rarely considered in DHT- based protocols. T-DHT [26] proposes a recursive procedure. The first reference node maintains all the DHT network, than it announces its membership to its neighbors. Recursively, all nodes join the network. Such technique does not suite the case of network auto-deployment where there is a need to deal with concurrent node joining.

Iterative Successor Pointer Rewiring Protocol (ISPR) [65] is a simple bootstrap protocol for SSR [45] or VRR [43]. The main principal of ISPR is that each node maintains exactly one and only one successor and predecessor by exchanging Successor Solicitation messages. This leads to multiple consistent local rings. To ensure global consistency, a specified node in each ring floods the network. VRR uses also a similar procedure to ISPR.

The linear method [66] shares with ISPR the same steps to reach local ring consistency. However, it avoids its flooding step. In order to ensure global consistency, the linear method assumes that the address space is linear and not a ring. The edges in the virtual graph are undirected. Total ordering of nodes addresses is used in order to distinguish right and left neighbors.To form a global ring, the leftmost node establishes an edge to the rightmost node.

Beside VRR and SSR, this bootstrap procedure is also suitable to protocols having the structure of a virtual ring such as ScatterPastry [28], ScatterDHT [29] and Coral-based VRR [54]. Since the structure of VCP [40] is a cord and not a ring, it can apply the linear method but without applying the last step in the linear method that consists to link the leftmost and rightmost neighbors to form a ring.

7.6 Design options effects on network performance

The way DHT is applied over WSN affects deeply the network performance. As seen from table 1 and table 2, layered approaches have longer path length because they separate the use of DHT and routing protocols. Integrated approaches offer generally best performance in point of vue path length. Cross layered approaches have minimized routing table, since they only need information about neighbors to route messages. However their average path

length are in general higher than those of integrated approaches. This can be explained by the fact that most of cross layered approaches use greedy routing to route messages, whereas integrated approaches try to minimize the hops number by using shortcuts and/or greedy hops.

8 Conclusion

In this paper, we have investigated the opportunities as well as challenges of the DHT applicability in WSN. A classification and a comparison of the existing DHT-based routing and data management protocols have been reported.

There is no consensus on the best protocol since each protocol has its advantages and its limits. Each category of protocols is suitable for a specific applicability domain. Flat DHT-based protocols are mainly used in simple systems made of homogeneous sensors. These protocols can be applied in indoor surveillance for domotic applications. Hierarchical DHT-based protocols are more accommodated to complex systems composed of a great number of heterogeneous sensors. The applicability domains of these protocols are dependent to the relationship between master and slave nodes.

If slaves are totally dependent to master nodes, the WSN should be deployed manually. These protocols can be suitable to intelligent agriculture applications, by helping growers to monitor and control watering and growing plants.

The applicability domain of the DHT-based protocols in which slave nodes are partially dependent to masters is larger since these protocols support WSN auto-deployment. They can be a suitable solution to disaster monitoring applications.

Existing DHT-based routing and data management protocols present good performance in terms of data lookup and storage in WSN. However, sensors dynamism as well as asymmetric link detection and bootstrapping are rarely considered. Further researches are needed to address these issues.

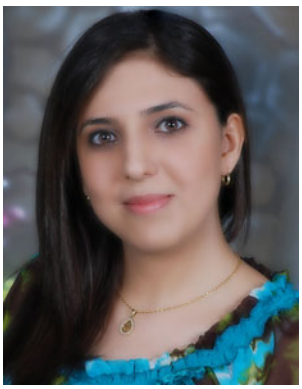
References

1. Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: A survey. *Computer Networks, 38*(4), 393–422.
2. Yick, J., Mukherjee, B., & Ghosal, D. (2008). Wireless sensor network survey. *Computer Networks, 52*(12), 2292–2330.
3. Lewis, F. L. (2004). *Wireless sensor networks. Smart environments: Technologies, protocols, and applications*. New York: Wiley.
4. Yingshu, L., & My, T. (Eds.) (2008). *Wireless sensor networks and applications*. Springer series on signals and communication technology.
5. Kuorilehto, M., Hännikäinen, M., & Hämäläinen, T. D. (2005). A survey of application distribution in wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking, 38*(5), 774–788.
6. Neves, P., Stachyra, M., & Rodrigues, J. (2008). Application of wireless sensor networks to healthcare promotion. *Journal of Communications Software and Systems, 4*(3), 181–190.
7. Chong, C. Y., & Kumar, S. (2003). Sensor networks: Evolution, opportunities, and challenges. *Proceedings of the IEEE, 91*(8), 1247–1256.
8. Puccinelli, D., & Haenggi, M. (2005). Wireless sensor networks: Applications and challenges of ubiquitous sensing. *IEEE Circuits and Systems Magazine, 5*(3), 19–31.
9. Akyildiz, I. F., & Kasimoglu, I. H. (2004). Wireless sensor and actor networks: Research challenges. *Ad Hoc Networks, 2*(4), 351–367.
10. Wehrle, K., Gtz, S., & Rieche, S. (2005). Distributed Hash tables. In R. Steinmetz, K. Wehrle (Eds.), *Peer-to-Peer systems and applications* (Chapter 7, pp. 79–93). Berlin, Heidelberg: Springer.
11. Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., & Balakrishnan, H. (2001). Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM conference of the Special Interest Group on Data Communication*, San Diego, CA.
12. Rowstron, A., & Druschel, P. (2001). Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proceedings of IFIP/ACM international conference in distributed systems platforms*, Heidelberg, Germany.
13. Ratnasamy, S., Francis, P., Handley, M., Richard, K., & Schenker, S. (2001). A scalable content addressable network. In *Proceedings of the ACM conference of the Special Interest Group on Data Communication*, San Diego, CA.
14. Heinzelman, W., Kulik, J., & Balakrishnan, H. (1999). Adaptive protocols for information dissemination in wireless sensor networks. In *Proceedings of the 5th annual ACM/IEEE international conference on mobile computing and networking*, Seattle, WA.
15. Intanagonwiwat, C., Govindan, R., & Estrin, D. (2000). Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of the 6th annual ACM/IEEE international conference on mobile computing and networking*, Boston, MA.
16. Estrin, D., Govindan, R., John, H., & Satish, K. (1999). Next century challenges: Scalable coordination in sensor networks. In *Proceedings of the 5th annual ACM/IEEE international conference on mobile computing and networking*, Seattle, WA.
17. Braginsky, D., & Estrin, D. (2002). Rumor routing algorithm for sensor networks. In *Proceedings of the first workshop on sensor networks and applications*, Atlanta, GA.
18. Heinzelman, W., Chandrakasan, A., & Balakrishnan, H. (2000). Energy-efficient communication protocol for wireless sensor networks. In *Proceedings of the Hawaii international conference system sciences*, Hawaii.
19. Lindsey, S., & Raghavendra, C. S. (2002). PEGASIS: Power Efficient GAthering in Sensor Information Systems. In *Proceedings of the IEEE aerospace conference*, Big Sky, Montana.
20. Manjeshwar, A., & Agrawal, D. P. (2001). TEEN: A protocol for enhanced efficiency in wireless sensor networks. In *Proceedings of the international workshop on parallel and distributed computing issues in wireless networks and mobile computing*, San Francisco, CA.
21. Manjeshwar, A., & Agrawal, D. P. (2002). APTEEN: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. In *Proceedings of the 2nd international workshop on parallel and distributed computing issues in wireless networks and mobile computing*, Ft. Lauderdale, FL.
22. Younis, M., Youssef, M. & Arisha, K. (2002). Energy-aware routing in cluster-based sensor networks. In *Proceedings of the*

10th IEEE/ACM international symposium on modeling, analysis and simulation of computer and telecommunication systems, Fort Worth, TX.
23. Xu, Y., Heidemann, J., & Estrin, D. (2001). Geography-informed energy conservation for ad hoc routing. In *Proceedings of the 7th annual ACM/IEEE international conference on mobile computing and networking*, Rome, Italy.
24. Yu, Y., Estrin, D., & Govindan, R. (2001). Geographical and energy-aware routing: A recursive data dissemination protocol for wireless sensor networks. UCLA Computer Science Department technical report, UCLA-CSD TR-01-0023.
25. Rao, A., Ratnasamy, S., Papadimitriou, C., Shenker, S., & Stoica, I. (2003). Geographic routing without location information. In *Proceedings of the 9th ACM international conference on mobile computing and networking*, San Diego, CA, USA.
26. Landsiedel, O., Lehmann, K., & Wehrle, K. (2005). T-DHT: Topology-based distributed Hash tables. In *Proceedings of the fifth IEEE international conference on Peer-to-Peer computing*, Konstanz, Germany.
27. Awad, A., Sommer, C., German, R., & Dressler, F. (2008). Virtual cord protocol (VCP): A flexible DHT-like routing service for sensor networks. In *Proceedings of the 5th IEEE international conference on mobile ad hoc and sensor systems*, Atlanta, Georgia.
28. Almamo, A., & Labiod, H. (2007). ScatterPastry: An overlay routing using a DHT over wireless sensor networks. In *Proceedings of the international conference on intelligent pervasive computing*, Jeju Island, Korea.
29. Almamou, A., Schiller, J., Labiod, H., & Mesut, G. (2008). A Case for an overlay routing on top of MAC layer for WSN. In *Proceedings of the second international conference on sensor technologies and applications*, Cap Esterel, France .
30. Stefan, G., Simon, R., & Klaus, W. (2005). Selected DHT algorithms. In *Peer-to-Peer systems and applications* (Chapter 8, pp. 95–117). Berlin, Heidelberg: Springer.
31. Rhea, S., Geels, D., Roscoe, T., & Kubiatowicz, J.(2004). Handling churn in a DHT. In *Proceedings of the annual technical conference USENIX*, Boston, MA.
32. Castro, M. C., Kassler, A. J., Chiasserini, C.-F., Casetti, C., & Korpeoglu, I. (2010). Peer-to-Peer overlay in mobile ad-hoc networks. In *Handbook of Peer-to-Peer networking* (Part 9, pp. 1045–1080).
33. Himabindu, P., Saumitra, M. D., & Y. Charlie, H. (2004). 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*, Low Wood, Lake Windermere.
34. Johnson, D. B., & Maltz, D. A. (1996). Dynamic source routing in ad hoc wireless networks. In T. Imielinski, H. Korth (Eds.), *Mobile computing* (Chapter 5, pp. 153–181). Dordrecht: Kluwer.
35. Jacquet, P., Muhlethaler, P., Clausen, T., Laouiti, A., Qayyum, A., & Viennot L. (2001). Optimized link state routing protocol for ad hoc networks. In *Proceedings of IEEE INMIC*, Lahore, Pakistan.
36. Perkins, C. E., & Royer E. M. (1999). Ad hoc on-demand distance vector routing. In *Proceedings of the second IEEE workshop on mobile computing systems and applications*, New Orleans, LA, USA.
37. Cramer, C., & Fuhrmann, T. (2006). Performance evaluation of chord in mobile ad hoc networks. In *Proceedings of the ACM international workshop on decentralized resource sharing in mobile computing and networking*, CA, USA.
38. Castro, M. C., Villanueva, E., Ruiz, I., Sargento, S., & Kassler, A. J. (2008). Performance evaluation of structured p2p over wireless multi-hop networks. In *Proceedings of the 2nd international conference on sensor technologies and applications*, Cap Esterel, France.
39. Wiberg, B. (2002). Porting aodv-uu implementation to ns2 and enabling tracebased simulation. Masters thesis, Uppsala University.
40. Awad, A., German, R., & Dressler, F. (2011). Exploiting virtual coordinates for improved routing performance in sensor networks. *EEE Transactions on Mobile Computing, 10*(9), 1214–1226.
41. ScatterWeb Homepage, Freie Universitt Berlin, Berlin, http://scatterweb.mi.fu-berlin.de.
42. Charles, E., & Bhagwat, P. (1994). Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proceedings of the international conference on communications architectures, protocols and applications*, London, UK.
43. Caesar, M., Castro, M., Nightingale, E. B., O'Shea, G. & Rowstron, A. (2006). Virtual ring routing: Network routing inspired by DHTs. In *Proceedings of ACM/SIGCOMM*, Pisa, Italy.
44. Malkhi, D., Sen, S., Talwar, K., Werneck, R. F., & Wieder, U. (2009). Virtual ring routing trends. In *Proceedings of the 23rd international conference on distributed computing*, Berlin, Heidelberg.
45. Fuhrmann, T. (2005). The use of scalable source routing for networked sensors. In *Proceedings of the 2nd IEEE workshop on embedded networked sensors*, Sydney, Australia.
46. Fuhrmann, T., Di, P., Kutzner, K., & Cramer, C. (2006). Pushing Chord into the underlay: Scalable routing for hybrid manets. Universitt Karlsruhe (TH), Technical report.
47. Fuhrmann, T. (2005). Scalable routing for networked sensors and actuators. In *Proceedings of the IEEE second annual conference on sensor and ad hoc communications and networks*, Santa Clara, California, USA.
48. Ratnasamy, S., Karp, B., Yin, L., & Yu, F. (2002). GHT: A geographic hash table for data-centric storage. In *Proceedings of the 1st ACM international workshop on wireless sensor networks and applications*, Atlanta, GA, USA.
49. Brad, K., & Kung, H. T. (2000). GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th annual ACM/IEEE international conference on mobile computing and networking*, Boston, USA.
50. Muneeb, A., & Koen, L. (2007). A case for Peer-to-Peer network overlays in sensor networks. In *Proceedings of the international workshop on wireless sensor network architecture*, Cambridge, MA, USA.
51. Gnawali, O., Jang, K.-Y., Paek, J., Vieira, M., Govindan, R., Greenstein, B., et al. (2006). The tenet architecture for tiered sensor networks. In *Proceedings of the 4th ACM conference on embedded networked sensor systems*, Boulder, CO.
52. Sioutas, S., Oikonomou, K., & Papaloukopoulos, G. (2009). Building an efficient P2P overlay for energy-level queries in sensor networks. In *Proceedings of the international conference on management of emergent digital ecosystems*, Lyon, France.
53. Sioutas, S. (2008). NBDT: An efficient p2p indexing scheme for web service discovery. *Journal of Web Engineering and Technologies, 4*(1), 95–113.
54. Fersi, G., Louati, W., & Ben Jemaa, M. (2010). Energy-aware virtual ring routing in wireless sensor networks. *Journal of Network Protocols and Algorithms, 2*(4), 16–29.
55. Yu, J., Liu, W., & Song, J. (2007). C2WSN: A two-tier Chord overlay serving for efficient queries in large-scale wireless sensor networks. In *Proceedings of the 15th international conference on advanced computing and communications*, India.
56. Muneeb, A., & Uzmi, Z. (2004). CSN: A network protocol for serving dynamic queries in large-scale wireless sensor networks. In *Proceedings of 2nd annual conference on communication networks and services research*, Fredericton, NB, Canada.
57. Pengfei, D., Yaser, M., Qing, W., Jorg, W., & Thomas, F. (2007). Application of DHT-inspired routing for object tracking. In *Proceedings of the IEEE international conference on mobile adhoc and sensor systems*, Pisa, Italy.

58. Di, P., & Fuhrmann, T. (2009). Using link-layer broadcast to improve scalable source routing. In *Proceedings of the international conference on wireless communications and mobile computing: Connecting the world wirelessly*, Leipzig, Germany.
59. Birnstill, P., & Fuhrmann, T. (2010). Using asymmetric links to improve SSR's routing performance. In *Proceedings of the 9th IFIP annual Mediterranean ad hoc networking workshop*, Juan-Les-Pins, France.
60. Albano, M., Chessa, S., Nidito, F., & Pelagatti, S. (2007). Q-NiGHT: Adding QoS to data centric storage in non-uniform sensor networks. In *Proceedings of the international conference on mobile data management*, Mannheim, Germany.
61. Neumann, J. V. (1951). Various techniques used in connection with random digits. *National Bureau of Standards Applied Mathematics Series, 12*(1951), 36–38.
62. Awad, A., Shi, L. German, R., & Dressler, F. (2009). Advantages of virtual addressing for efficient and failure tolerant routing in sensor networks. In *Proceedings of the sixth IEEE/IFIP international conference on wireless on-demand network systems and services*, Snowbird, UT.
63. Dressler, F., Awad, A., German, R., & Mario, G. (2009). Enabling inter-domain routing in virtual coordinate based ad hoc and sensor networks. In *Proceedings of ACM MobiCom, poster session*, Beijing, China.
64. Dressler, F., Koch, R., & Gerla, M. (2010). Path Heuristics using ACO for inter-domain routing in mobile ad hoc and sensor networks. In *Proceedings of the ACM/ICST international conference on bio-inspired models of network, information and computing systems*, Boston.
65. Cramer, C., & Fuhrmann, T. (2005). *Self-stabilizing ring networks on connected graphs*. University of Karlsruhe (TH), Fakultaet fuer Informatik, Technical report 2005-5.
66. Kutzner, K., & Fuhrmann, T. (2007). Using linearization for global consistency in SSR. In *Proceedings of 4th international IEEE workshop on hot topics in P2P systems*, Long Beach, CA.

Author Biographies



Ghofrane Fersi graduated in Computer Science engineering from the National School of Engineers of Sfax (ENIS), Tunisia, in 2008. She received her M.S. degree (DEA) in Systems of Information and dedicated New Technologies (NTSID) in 2009, from ENIS. Since December 2010, she has joined the Research Unit of Development and Control of Distributed Applications (ReDCAD) at the National School of Engineering of Sfax (ENIS) as a Ph.D. candidate. Her current research focuses on energy optimization in Distributed Hash Table-based routing protocols in Wireless Sensor Networks.

Wassef Louati graduated in Computer Science engineering from the National School of Engineers of Sfax,Tunisia, in 2003. In 2004, he joined the SAMOVAR (Distributed Services, Architectures, Modelling, Validation and Network Administration) research unit at the National Institute of Telecommunications (GET-INT), France, as a Ph.D. candidate. He received the Ph.D. degree in Computer Science from the Pierre et Marie Curie University (Paris 6) and GET-INT, in 2007. Since 2008, he has been an Associate Professor in the Department of Computer Science at the University of Monastir, Tunisia. His research activities are focused on Distributed Hash Table systems and applications in Wireless Sensor Networks.

Maher Ben Jemaa obtained his diploma of Engineer in Computer Science from the National School of Computer Sciences ENSI (Tunisia) in 1989 and his Ph.D. from the Department of Electrical Engineering at the National Institute of Applied Sciences (INSA) Rennes (France) in 1993. He joined the National School of Engineers of Sfax (ENIS) as Assistant Professor of Computer Science in the Department of Computer Science and Applied Mathematics in 1995. He became an Associate Professor in 1997 and he is a keynote professor since March 2011. His current research areas include Fault Tolerance of distributed systems, Quality of Service in Ad hoc Networks and routing issue in Wireless Sensor Networks.