Research Journal of Applied Sciences, Engineering and Technology 7(7): 1351-1363, 2014
DOI:10.19026/rjaset.7.401
ISSN: 2040-7459; e-ISSN: 2040-7467
© 2014 Maxwell Scientific Publication Corp.
Submitted: April 24, 2013 Accepted: June 19, 2013 Published: February 20, 2014

Research Article

Scalable Resource Discovery Architecture for Large Scale MANETs

Saad Al-Ahmadi and Abdullah Al-Dhelaan
Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia, P. O. Box: 51187, Riyadh 11543, Saudi Arabia

Abstract: The study conducted a primary investigation into using the Gray cube structure, clustering and Distributed Hash Tables (DHTs) to build an efficient virtual network backbone for Resource Discovery (RD) tasks in large scale Mobile Ad hoc NET works (MANETs). MANET is an autonomous system of mobile nodes characterized by wireless links. One of the major challenges in MANET is RD protocols responsible for advertising and searching network services. We propose an efficient and scalable RD architecture to meet the challenging requirements of reliable, scalable and power-efficient RD protocol suitable for MANETs with potentially thousands of wireless mobile devices. Our RD is based on virtual network backbone created by dividing the network into several non overlapping localities using multi-hop clustering. In every locality we build a Gray cube with locally adapted dimension. All the Gray cubes are connected through gateways and access points to form virtual backbone used as substrate for DHT operations to distribute, register and locate network resources efficiently. The Gray cube is characterized by low network diameter, low average distance and strong connectivity. We evaluated the proposed RD performance and compared it to some of the well known RD schemes in the literature based on modeling and simulation. The results show the superiority of the proposed RD in terms of delay, load balancing, overloading avoidance, scalability and fault-tolerance.

Keywords: Distributed hash table, gray cube, mobile ad hoc networks, multi-hop clustering, resource discovery

INTRODUCTION

The proliferation of mobile computing devices capable of wireless communication causes a revolutionary change in our information society. These devices such as cell phones, tablets, pads, note books, PDA, laptops and wearable computers are the basis of new generation of computer networks and ubiquitous computing environments. MANETs are multi-hop wireless networks that are self configured, rapidly deployable and has neither fixed infrastructure nor centralized control. In MANETs, nodes are heterogeneous mobile nodes with limited energy sources and short wireless transmission range. MANET topology may change rapidly and unpredictably due to node mobility, depleted energy and intermittent wireless communication. Every mobile device is required to act as router for communication throughout the network despite transmission failures and limited available power (Sailhan and Issarny, 2005). Applications of MANETs include many rapidly deployed networks for disaster relief operations, military battle fields and public events. There are many issues and challenges that need to be considered in designing effective MANETs. Some of these issues are routing, deployment, self organization, security, scalability, energy management and Resource Discovery (RD). MANETs have been the subject of rigorous research to create fully decentralized multi-hop wireless network, but little research has been given to RD in large-scale MANETs (Bettstetter and Renner, 2000).

RD protocol is an integral part of any computer network to facilitate resource utilization. The two terms resource and service are used interchangeably in the literature. A resource can be a tangible (hardware) or intangible (software) entity. Hardware resources include processors, memories, printers and communication links. Software resources include files, database management systems, hyperlinks and security authentication. RD is defined as the process that allows network entities to publish their resources and look for other entity's resources. Resources are registered by full descriptive information tags which include resource name, host node location, input and output parameters with their complete format and description and various additional resource related attributes (Meshkova *et al*., 2008). Resources are categorized based on several objectives like functionality, content, semantic, characteristics, performance utilization and QoS. RD

Corresponding Author: Saad Al-Ahmadi, Computer Science Department, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia, P.O. Box: 51187, Riyadh 11543, Saudi Arabia

architecture design is influenced by network characteristics such as number of nodes, transmission speed, topology dynamics and type of service. RD performance is measured by network scalability, load balancing, query efficiency and fault tolerance (Richard III, 2000). The increased number of mobile heterogeneous nodes with various types of resources increases the size of MANETs and imposes great challenges for effective exploitation of the available resources with scalable capabilities. RD process should be tolerant to communication faults between the wireless nodes due to depleted energy or mobility reasons (Eriksson *et al*., 2004).

There are various protocols for RD in MANETs particularly tailored to specific sets of objectives. RD two main tasks are: registration (advertisement or publishing) and search (lookup). Every RD protocol consists of at least two basic participating elements: client (user) and server. The client is the entity interested in finding and using the resource offered by the server entity. A third entity called the directory or broker may or may not be used. RD protocols can be classified into two categories: directory-less (client-server) protocols and directory-based (resource brokers) protocols. In directory-less structure, servers broadcast their advertisements and clients broadcast their requests. Both broadcasting processes may take place at the same time in the network. The server listens to client's requests at predetermined network interface and send reply message to the requester if the resource is available. The client-server mode is simple, does not require physical or logical infrastructure and can be implemented using broadcasting, multicasting, or any casting techniques. One major problem with client-server protocols is how to determine the frequency of resource advertisements in order to reduce network load and avoid redundant transmissions (Oliveira *et al*., 2005). The client-server mode is not suitable for large-scale MANETs, while the resource broker (directory-based) mode is more suitable because it is scalable, faster in resource locating, prevent high demand server flooding, achieve load balancing between nodes and improves the overall RD performance. In directory-based structures, resources are registered using central or distributed resource repositories to minimize RD queries. All client's requests and server's registrations are directed to the resource broker (directory) which in return send back resource reply messages to the clients and registration acknowledgments to the servers. The resource broker mode requires assigning dynamically one or more nodes the functionality of the directory which presents an extra load to the network. Centralized directory approaches eliminate the need for extensive resource lookup, however, it is not good strategy for MANETs, because the central directory: not always reachable, presents a single point of failure, not scalable and exhaust the limited node's resources. Thus, centralized directories are not well suited for MANETs and distributed directories are more suitable where few nodes act as resource directories. Sometimes, the distributed directories may have coordination between them to relief the resource provider from publishing to more than one directory node. Distributed directories may be implemented using a backbone of directory-enabled nodes technique, set of clusters technique, or Distributed Hash Table (DHT) technique. Also, there are some hybrid structure that consists of directory-based and directory-less architectures where the directory is used to hold information about existing resources only in the node's vicinity. RD task can be placed in the application layer of the network structure or coupled with the network layer. In the latter case, the overhead of message traffic is reduced and limited energy resources are saved at the expense of flattening the route discovery task.

In this study, we propose a new RD architecture based on virtual backbone constructed using Gray cubes in different network localities. The first step is to divide the network into several non-overlapping multi-hop clusters using limited flooding-based technique. This type of clustering generates clusters of maximum diameter 2 *d*, where, *d* is local adaptable integer parameter and designated node as cluster head. This clustering makes every cluster head aware of all the current cluster members and their roles as ordinary nodes or gateway nodes. Next, the cluster head creates local Gray cube from all or some of the existing cluster members. The created Gray cube dimension is adaptable to the cluster size and the current load of the local RD task. Finally, all Gray cubes are connected to form a virtual backbone through cluster's gateways and access points. The network resources are distributed among all the nodes of the network backbone which serve as distributed directories. The local and global addressing of the Gray cubes facilitates applying Distributed Hashing Table (DHT) structure for efficient resource distribution and lookup requests. The proposed RD architecture allows the creation of completely scalable MANETs by decoupling resource's identifier from the node's topology-dependent address. It is fault-tolerant by providing multiple paths between any couple of nodes (client and server) and gives an excellent logical-topological mapping. It allows load-balancing between backbone nodes in contrast to what occurs in clustering, where the head is heavily loaded.

LITERATURE REVIEW

RD is an important component of MANET to achieve stand-alone and self-configurable communication network. It allows mobile devices to automatically discover network resources with their attributes and advertise their own capabilities to the rest of the network (Chakraborty *et al*., 2006). Several standards exist for RD protocols (Bettstetter and Renner, 2000), examples: Salutation (Salutation Consortium, 1999), Service Location Protocol (SLP) (Guttman, 1999) and Jini (Arnold *et al*., 1999). All these standards are designed basically for wired networks or fixed infrastructure networks. Several RD protocols have been proposed for MANETs in the literature. In this paper we consider 6 of these previous protocols: Simple Flooding Protocol (SFP) (Oliveira *et al*., 2005), Distributed Service Discovery Protocol (DSDP) (Kozat and Tassiulas, 2003), Max-Min Clustering Algorithm (Max-Min) (Amis *et al*., 2000), Adaptive Multi-Hop Clustering protocol (AMHC) (Al-Ahmadi and Al-Dhelaan, 2012), DHT-based Functionalities Using Hyper cubes (DHT) (Alvarez-Hamelin *et al*., 2006) and Resource Discovery Scheme for Large Scale Ad Hoc Networks Using a Hypercube-Based Backbone (HYPER) (Dekar and Kheddouci, 2009).

SFP is a directory-less RD protocol while the remaining methods are distributed directory-based RD protocols using virtual backbone. In SFP, the discovery process starts with a source node broadcasting a packet to all neighbors. Each of those neighbors in turn rebroadcast the packet exactly one time and this continues until all reachable network nodes have received the packet. Several versions of reliable broadcast and multicast in MANETs exists (Peng and Lu, 2000). Broadcasting is used extensively in MANETs and considered an important operation in many on-demand (reactive) routing protocols; examples include: DSR, AODV and ODMRP. Flooding is used in the simple protocol of the IETF Internet Draft protocol for broadcasting and multicasting in low densities and/or high mobility ad hoc networks. Also, flooding is a viable candidate for reliable multicast in MANETs with high mobility (Williams and Camp, 2002). Although flooding is simple, it has poor scalability problems and consumes high amount of network resources because of tremendous amount of duplicated messages. Also, flooding leads to serious redundancy, contention and collision in MANETs known as the broadcast storm problem.

DSDP is distributed RD architecture based on virtual backbone for locating and registering resources within dynamic network topology. DSDP consists of two phases: Back Bone Management (BBM) Phase and Distributed Service Discovery (DSD) Phase. BBM is responsible of network backbone constructs while DSD distributes resources registrations, requests and replies. In DSDP, the network is represented by connected graph with where the maximum distance (radius) is greater than or equal to 3 hops only. The virtual backbone is a dominating set created from a subset of nodes acting as service brokers and subset of paths (virtual links) connecting these brokers. All the nodes in the network are either in the dominating set (service brokers) or only one-hop away from at least one service broker. BBM is adaptable to network topology changes

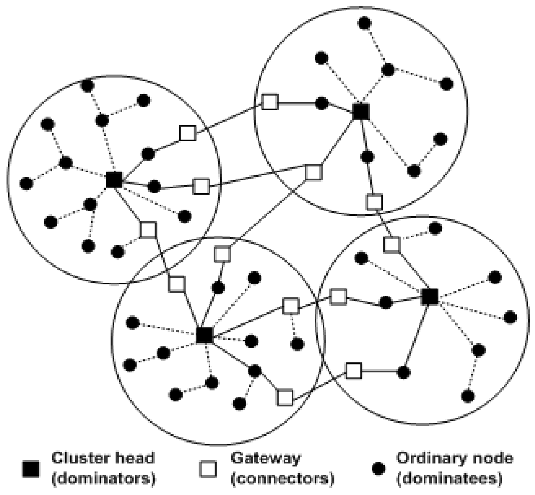

Fig. 1: Backbone of four clusters each of 2-hops

by adding or removing nodes to or from the created backbone. The second phase (DSD) distributes the request and register messages from the service agents to the backbone nodes (service brokers). These messages assist in forming multicast trees rooted at client and server nodes on top of the backbone mesh. DSDP generates backbone with large number of nodes which causes significant communication responsibility on individual nodes. Extensive communication drains battery, limits network life and decreases the overall network performance among many other side effects. When the network size increases, number of messages exchanged to maintain resource tables may cause congestion and generate significant delays in message propagation from one node to another.

Some RD protocols are based on clustering. Clustering divides the network into *d*-hop clusters where $d \geq 1$. When $d = 1$, the generated one-hop clusters have the property that every node is no more than one-hop away from their elected cluster head. One-hop clustering generates large number of clusters especially in dense networks; for this reason cluster maintenance phase is both resource and time consuming. As rule of thumb, it is desirable to have control over number of created clusters, thus multi-hop clustering is used with $d \geq t$, where t is clusters number threshold that varies with time. In multi-hop clustering, nodes are classified as: cluster head (exactly one node per cluster), gateway nodes and ordinary node. A cluster head is responsible for managing RD tasks within its proximity; i.e., the *d*-neighborhood of the cluster head. A gateway node has neighbor node in another cluster (connecting two or more clusters). All the other nodes that are non cluster head and non gateways are ordinary members. In multi-hop clustering, the cluster heads form a dominating set used as a virtual backbone to register or lookup network resources (Wang and Olariu, 2004).

The Max-Min protocol is a multi-hop clustering where any node is at most *d* hops away from its elected cluster head (Fig. 1). Max-Min can be used easily as underlying structure for RD task. Nodes participate in the leader election based on their node id only. The algorithm consists of 2 *d* rounds of flooding where nodes exchange their ids with their neighborhood. Max-Min has 2 phases; each phase consist of *d* rounds; called Flood max and Flood min. Each node has two arrays of size 2 *d* named WINNER and SENDER records the winner's id and the sender id (the node that sent the winner id) during a particular round of 2 *d* rounds of flooding. The WINNER array helps in identifying the elected cluster head while the SENDER array helps in establishing a path to the elected head. During every Flood max round, each node exchanges its WINNER id with the received node id from its one-hop neighboring nodes. If the received node id is greater than the currently stored WINNER node, then it replaces it. The Flood min is similar to Flood max except it chooses the smallest node id in order to gives chance to small node ids to regain back some of their lost territories. After the flooding process terminates, leader election process follows a set of four rules consecutively to elect the leader. This heuristic does not take into account mobility and assumes the network is static during the algorithm execution. It also does not take into consideration energy or neighborhood density. The algorithm is executed in O (*d*) but the hidden order constants are high because of the large number of exchanged messages.

AMHC is a cross-layer RD protocol based on multi-hop clustering. It constructs virtual backbone in a timely fashion to capture changes in network topology due to nodes mobility and/or limited resources. The backbone consists from the cluster heads and their associated gateways. AMHC falls in the same category with Max-Min with enhanced features. AMHC strategy favors reelecting cluster heads in future rounds, thereby reducing transition overheads. This type of overhead occur when old cluster heads give way all the resources details to the new cluster heads. AMHC utilizes all the designated gateways to generate network backbone with multiple paths between cluster heads. AMHC optimizes load-balance and stability which are usually mutually conflicting goals. High stability causes heavy load on cluster heads and drain their scarce energy resources quickly. Low stability causes excessive communication in transferring cluster resource tables from the old cluster head to the new cluster head causing performance degradation and battery consumption. It runs for *d* rounds of information exchange, in every round it elects a node with the highest weight. Node's weight is based on: current available energy, connectivity degree and node-to-head path length. A node with the highest weight in its *d*-neighborhood becomes a cluster head. The algorithm is executed in O (*d*) and very efficient in terms of RD performance, network life time and fault-tolerance. For large values of the parameter *d*, AMHC face the problem of heavy loaded cluster heads with large resource tables. In this case, higher degree of resource distribution is necessary to prevent partial bottle necks and congestion in dense resources areas.

Some RD protocols use hyper cubes. The hypercube is an interconnection network with attractive properties of a small diameter, multi-paths and symmetry. The hypercube has been used in various applications on massively parallel computers and networking. Lately, hypercube has been used in Peer-to-Peer (P2P) networks (Schlosser *et al*., 2003). Also, hyper cubes have been used in ad hoc networks for routing (Alvarez-Hamelin *et al*., 2006; Manoharan and Thambidurai, 2006), RD (Dekar and Kheddouci, 2009) and WSN data gathering (Chuang *et al*., 2007). The hypercube is symmetric and strongly connected structure with small diameter and small link complexity. A hypercube of dimension n is a connected graph $G = (V, E)$ where $|V| = 2^n$ nodes labeled with n-bit binary string $(b_{n-1} \dots b_0) \in \{0,1\}^*$and $|E| = n2^{n-1}$. Two nodes $(b_{n-1} \dots b_k \dots b_0)$ and $(b_{n-1} \dots b'_k \dots b_0)$ where $b_k \neq b'_k, k \in \{0..n-1\}$ are connected if they differ by one bit position only. Let $\|x, y\|$ defined as the distance d between two nodes x and y be the length of the shortest path from x to$y$. The hypercube has a diameter $(\max_{x,y \in V} \|x, y\|)$ equal to n and average distance $\bar{d} = n/2$. The hypercube structure can be used to link all the nodes and provides multi-hop fault-tolerant multi-path network structure.

Some RD protocols use Distributed Hash Tables (DHTs). DHT uses algorithms to assign and locate resources to network nodes based on some node's characteristics. From infrastructure aspect, DHTs rely on large variety of different structures, such as rings, multidimensional spaces, or other types of graphs especially hyper cubes for example Chord (Stoica *et al*., 2001), Pastry (Rowstron and Druschel, 2001), Tapestry (Zhao *et al*., 2004), eQuss (Locher *et al*., 2006), HyCube (Olszak, 2010), Hyper Cup (Schlosser *et al*., 2003) and HYPEER (Serbu *et al*., 2011). One notable difference between these structures is the node's degree which can be constant, logarithmic, linear, or other equations. Every node maintains continuous contact with its neighbors to exchange routing information. In P2P systems, there are several DHTs arrange the peers over hypercube structure to allow efficient search and form resilient structure against attacks and frequent network changes. Pastry and Tapestry are scalable, decentralized resource location and routing for large-scale P2P systems based on address prefixes, which can be viewed as a generalization of hypercube routing. Unlike traditional routing algorithms (like distance vector and link-state protocols) which globally propagate information about routes to each destination, Pastry nodes only use local information which is more scalable and decentralized. The eQuus system has a topology of a partial hypercube where each vertex represents a clique, i.e., a group of nodes that are close in terms of proximity metric. HyCube is DHT routing system based on hierarchical hypercube geometry. HyperCuP is DHT system with hypercube structure where every node keeps its neighbors on a per-dimension basis. If neighbor's count is less than the cube dimension, one or more neighbors are replicated to substitute this shortage. HYPEER is a P2P DHT overlay design loosely based on a hypercube structure providing redundant paths. It approximates a hypercube by extending a ring-based DHT overlay and controlling the placement of nodes when they join the network.

Some routing protocols based on P2P networks have been introduced for MANET's routing. MADP as try (Zahn and Schiller, 2005) is a DHT substrate that considers physical locality and integrates the functionality of DHT and AODV at the network layer to provide an efficient indirect routing primitives in MANETs. The DHT protocol proposed in Alvarez-Hamelin *et al.* (2006) utilizes DHT as a scalable network substrate to provide location-independent node identification. It creates a partial hypercube in the network that allow establishment of multiple paths between any two nodes, which increases the robustness of the topology to mobility. Decoupling the permanent identifier of a node from the node's topology-dependent address is very important for completely scalable self-organizing networks. In (Dekar and Kheddouci, 2009) a hybrid RD scheme (HYPER) for ad hoc networks is proposed based on hyper cubes and DHT. DHT-based systems allow fast locating of resources with minimum overhead given its key. The dynamic topology of MANETs and multi-hop nature of communication increase the cost of constructing and maintaining the underlying overlay network and decreases the DHT-based system's efficiency.

Our RD architecture creates virtual network backbone that handles resource requests efficiently. The backbone is constructed in three phases. The first phase is to divide the network into several non-overlapping localities using multi-hop clustering technique where the cluster head is aware about all the current cluster members. The second phase is to create in every locality a Gray cube of certain dimension based on several parameters such as number of nodes, current load, etc. The cluster head handles the job of creating the Gray cube in every locality with close physical to logical mapping to realize the great benefits of the Gray cube structure. The final phase is to link all these Gray cubes through gateways to form virtual network backbone. This backbone allows the application of DHT operations efficiently for resource advertisements and lookup queries.

MATERIALS AND METHODS

In this study, we use the Gray cube structure (Al-Ahmadi *et al.*, 2011) as the main component of the proposed RD architecture for large scale MANETs. The n-dimensional Gray cube GQ_n (Fig. 2) is a connected undirected graph $GQ_n = (V, E)$, $|V| = 2^n$, $|E| = n2^{n-1}$. The Gray cube is a hypercube-like regular structure that lacks symmetry but has lower diameter and shorter average path length. In ad hoc networks, when we minimize the average path length (number of hops) will lead to minimizing number of exchanged messages which save scarce node's resources especially power. Limiting number of generated data packets per resource publishing or lookup query enhances RD efficiency and increase MANET lifetime.

GQ_n is built recursively using the Gray code (reflected binary Gray code). The Gray code is a binary numeral where two successive values differ in only one bit. Let $B_n = b_{n-1} \dots b_1 b_0$ to be n-bit binary code and $G_n = g_{n-1} \dots g_1 g_0$ to be n-bit Gray code, G_n can be obtained by $G_n = B_n \oplus B'_n$ where $B'_n = 0b_{n-1} \dots b_1$ and $\oplus$ is XOR operation. Another method to obtain G_n:

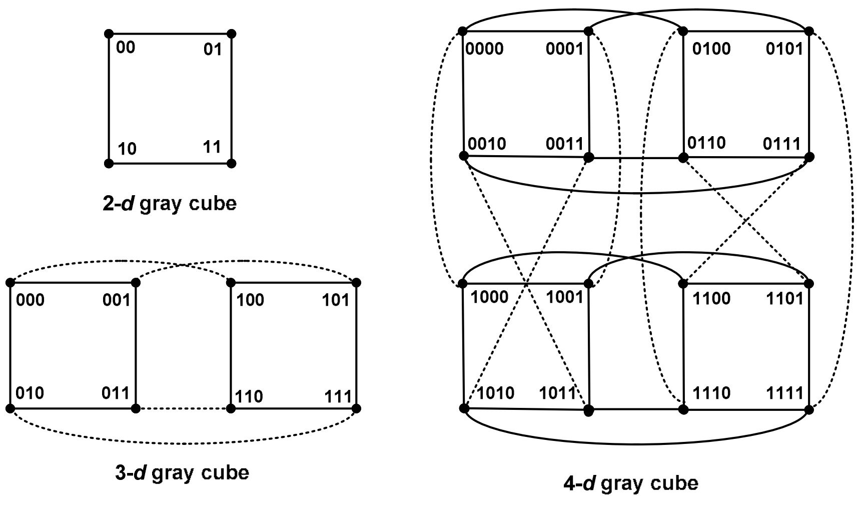

Fig. 2: Gray cubes of different dimensions

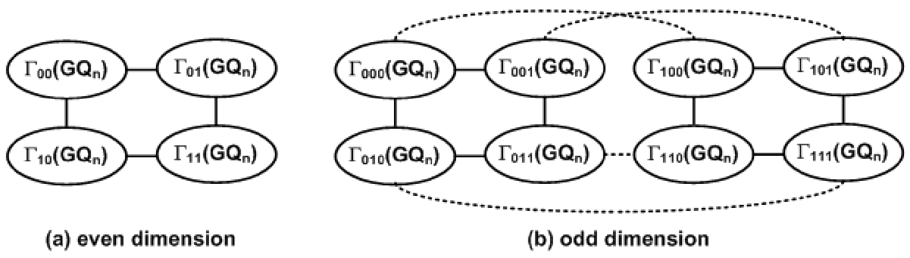

Fig. 3: The generalized case of Gray cube dimensions

$$g_i = \begin{cases} b_i \oplus b_{i+1} & , \quad 0 \le i \le n-2 \\ b_i & , \quad i = n-1 \end{cases}$$

GQ_{n+1} is defined inductively as $GQ_{n+1} = Inter(GQ_n, GQ_n^*, \pi^n)$, $n \ge 1$, where π^n is a permutation defined with relation to G_n as bijective function on $Z_n = \{0,1,\dots,2^{n-1}\}$. Both GQ_n and GQ_n^* are Gray cubes of equal dimension. The function $Inter$ represent one-to-one mapping between the nodes of GQ_n and GQ_n^* and consists of the following two steps:

- Node $v_i \in V(GQ_n)$ is connected with a node $\pi^n(v_i) \in V(GQ_n^*)$ by an edge
- The subset of vertices of GQ_{n+1} that are in GQ_n^* is relabeled by adding 2^n to their existing labels

GQ_n consists of two GQ_{n-1} denoted by GQ_{n-1}^0 and GQ_{n-1}^1, where GQ_{n-1}^0 is the subgraph of GQ_n that has all the nodes with prefix 0 in their label, while GQ_{n-1}^1 has those nodes having prefix 1. Let $\Gamma_\alpha(G)$ be the subgraph of all the nodes with prefix α. $\Gamma_\alpha(G)$and $\Gamma_\beta(G)$; for two binary strings α and β of equal length; are adjacent subgraphs of G if there exists at least one node $\mu \in \Gamma_\alpha(G)$ adjacent to some $v \in \Gamma_\beta(G)$. The diameter D of GQ_n is $\lceil (n+1)/2 \rceil$, which is almost half the diameter of the hypercube. Also, the average distance $\bar{d}$ is about 3/4 the average distance of the hypercube.

The Gray cube advantages come when we move from the lower odd dimension to the upper even dimension or vice versa (from the lower even dimension to the higher odd dimension. Let $n = 2k$, then $D(GQ_{2k}) = D(GQ_{2k+1}) = k+1$. By induction on the cube dimension, n, the base case is when $k = 1$, $D(GQ_2) = GQ_3 = 2$. The induction hypothesis is when $k > 1$ is divided into two cases based on the dimension: even dimension $(2k)$ and odd dimension $(2k+1)$. From Fig. 3, we have $\Gamma_{00}(GQ_{2k})$ and $\Gamma_{10}(GQ_{2k})$ are adjacent by direct link. Similar case $\Gamma_{01}(GQ_{2k})$ and $\Gamma_{11}(GQ_{2k})$. Also, $\Gamma_{00}(GQ_{2k})$ and $\Gamma_{01}(GQ_{2k})$ are adjacent by isomorphism to GQ_{2k-1}. Similar case $\Gamma_{10}(GQ_{2k})$ and $\Gamma_{11}(GQ_{2k})$. The only non symmetric case comes from the nonadjacent subgraphs pairs $(\Gamma_{01}(GQ_{2k}), \Gamma_{10}(GQ_{2k}))$ and $(\Gamma_{00}(GQ_{2k}), \Gamma_{11}(GQ_{2k}))$. So, the diameter is increased only by one for every two consecutive dimensions.

We evaluated the average path length $(\bar{d}_n)$ of GQ_n by induction on n. The base cases are $\bar{d}_1 = 0.5$, $\bar{d}_2 = 1$ and $\bar{d}_3 = 11/8$. Let $\Gamma_\alpha(GQ_n)$ and $\Gamma_\beta(GQ_n)$ be two isomorphic adjacent subgraphs to GQ_{2k-2}. We define $\bar{\gamma}_n$ as the average path length from vertex μ in $\Gamma_\alpha(GQ_n)$ to all adjacent vertices in $\Gamma_\beta(GQ_n)$. Given that $\bar{\gamma}_2 = ((1 \times 1 + 3 \times 2))/4 = 7/4$, we calculate $\bar{\gamma}_n$ by the recurrence relation $\bar{\gamma}_n = \bar{\gamma}_{n-2} + (3/4)$. The average path length starting from the source vertex μ in an even dimension GQ_n ($n = 2k$ and $\delta_n = 2^n$) is computed by adding 3 terms:

- Path length between Γ_α and Γ_β
- Path length of Γ_γ that is not adjacent to μ with every vertex having at least one neighbor to the adjacent subgraphs to Γ_α
- Path length of Γ_α where $\mu \in \Gamma_\alpha$ which has $\bar{d}_{n-2}$

Thus, for $n = 2k$, we have $\bar{d}_n$ equal to:

$$= \frac{(2\times\delta_{n-2}\times\bar{\gamma}_{n-2}) + (\delta_{n-2}\times(1+\bar{\gamma}_{n-2})) + (\delta_{n-2}\times\bar{d}_{n-2})}{\delta_n} = \frac{(2^{n-1}\times\bar{\gamma}_{n-2}) + (2^{n-2}\times(1+\bar{\gamma}_{n-2})) + (2^{n-2}\times\bar{d}_{n-2})}{2^n} = \frac{3\bar{\gamma}_{n-2}+\bar{d}_{n-2}+1}{4} \quad for\ (n = 2k\ and\ k \in Z_n)$$

Similarly, we calculated $\bar{d}_n$ for odd dimension $(n = 2k+1)$ and we came with this equation:

$$\bar{d}_n = \frac{7\bar{\gamma}_{n-3}+\bar{d}_{n-3}+4}{8} for\ (n = 2k+1\ and\ k \in Z_n)$$

As part of our study, we built model that creates for GQ_n from its original recursive definition. The model generates one of the corresponding minimum spanning tree $MST(GQ_n)$ rooted at selected network node. The average distance of GQ_n is computed empirically by traversing the minimum spanning tree in breadth first order. Let C_l be number of nodes in level l, the simple cases are $C_2 = n$ and $C_3 = (n-1)^2$. We have$\bar{d} = \frac{\sum_{l=1}^{m} l\times C_l}{2^n}, wherem = \left\lceil \frac{n+1}{2} \right\rceil$. The generated results match those results from our mathematical derivation.

The hyper cube can be embedded into a Gray cube with dilation of tow. This feature allows the Gray cube to simulate the hyper cube efficiently and make it suitable structure for any proposed RD scheme based on the hyper cube. It allows backward compatibility with existing hypercube schemes with enhanced performance.

Gray cube-based backbone construction: Our RD architecture starts by dividing the network into several clusters using multi-hop clustering algorithm. After the completion of this step, a Gray cube is created in every cluster. It is not necessary that all Gray cubes of the same dimension and not necessary that all local cluster nodes participate in the Gray cube construction process. The final step is to link all these separate Gray cubes to form a virtual network backbone that carry all the RD advertisements and lookup messages. Several multi-hops clustering algorithms are available in the literature, but our RD architecture require distributed asynchronous clustering algorithm that generates non-overlapping multi-hop clusters where any node is at most d-hops away from its elected cluster head. It also requires the cluster head has explicit knowledge of all the cluster's members with maximum number of gateways. Two candidate clustering algorithms are available: Max-Min and AMCH. We used AMHC for the following reasons: steady algorithm, generate clusters with balanced density, adaptable to MANET dynamics and more efficient than Max-Min (Al-Ahmadi and Al-Dhelaan, 2012).

For some MANET with n nodes and average cluster sizeS_c, on average there are $n_c = n/S_c$clusters. In every cluster C_i, $1 \leq i \leq n_c$, the elected cluster head selects subset of the local members GQ_i ($Q_i \leq C_i$) to create Gray cube of dimension k_i where $|GQ_i| = 2^{k_i}$ (the operation | | is the set size). The parameter k_i is controlled by many parameters such as local cluster size, nodes failure, nodes mobility rate, resource usage and physical topology layout. As RD load increases, k_i is increased and vice versa. The Gray cube dimension is used to control load degree and prevent overload or under load that occur when lookup queries are not uniformly distributed overall the resources or when the resources are not uniformly distributed over the backbone nodes. Those nodes in the Gray cube will play the role of directory agents in the RD process and form the underlying structure for DHT operations. The remaining $C_i - Q_i$ nodes are automatically connected to Q_i nodes as direct outcome from AMHC.

In AMHC, the cluster head assign the Gray cube addresses to the selected subset of nodes GQ_i. The cluster head itself holds address 0 and for every local node $x_i \in GG_i$, the cluster head assigns it the most adequate virtual address that have the maximum mapping between physical topology layout and the constructed virtual Gray cube topology. An adequate mapping is necessary to ensure minimum distances between adjacent nodes in physical and logical space. Communication with nearby neighbor is efficient in terms of wireless communication and energy more than communication with faraway neighbor. The concept of node closeness is related to location awareness. Using GPS or distributed localization methods to calculate node's location limits the proposed RD architecture application areas. We utilize combined neighborhood lists collected by the cluster head from all the cluster members. This step is already part from AMHC to identify the cluster's gateways. The cluster head can build excellent local Gray cube from these reported proximity lists sets. We follow three criteria to ensure maximum symmetry between physical and logical structures:

- **Nodes separation:** Two different nodes in the virtual Gray cube should correspond as much as possible to two different physical nodes. In some rare cases when there is no suitable neighbor in the nearby area, a virtual node with virtual address is created. In this case a physical node may have more than one virtual Gray cube address which may increase its responsibilities and lower Gray cube efficiency.
- **Nodes neighbors:** Two adjacent nodes in the virtual Gray cube should correspond as much as possible to two adjacent physical nodes. Physical closeness enables efficient wireless transmission.
- **Paths separation:** two paths in the virtual Gray cube should correspond as much as possible to two separate paths in the physical topology. This criterion enhances the Gray cube efficiency in providing multi-transmission paths for fault tolerance and load balance. If multiple virtual paths mapped onto the same physical path, then the usefulness of the Gray cube is degraded.

The problem of creating optimal multidimensional cube from 2 dimensions (or 3 dimensions) network topology is NP-complete problem. The Gray cube $GQ_n$is built on stages from dimension 1 to n. If pure greedy construction algorithm is used in every stage i that selects the nearest physical neighbor node as the logical cube neighbor at dimension i, we will get cubes with lower dimensions have excellent physical-logical mapping properties on the expense that the cubes with higher dimensions have less matching properties. For this reason, our RD protocol involves some controlled randomness in selecting the corresponding logical cube neighbor to alleviate the greediness of the algorithm. In the second phase, in every cluster generated from the first phase, GQ_i is built as follow: 2^{i-1} of GQ_1 are created, then every two cubes are linked to create 2^{m-2} of GQ_2 and so on. The building process continues until the building of the local GQ_i is finished. This building process is decentralized and asynchronous that occurs in every cluster independent of the remaining clusters.

We alleviate the problem of closeness and minimize necessary computation: every constructed

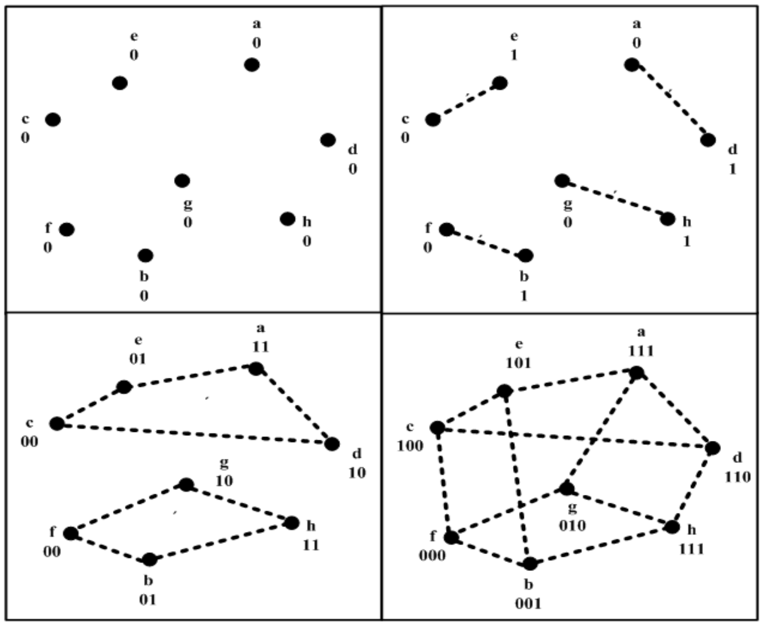

Fig. 4: Gray cube construction

cube GQ_ihas location, L, seen as center of gravity, defined as the middle point between the two lower dimensional cube subcomponents and calculated as: $L(GQ_i) = (L(GQ_{i-1}) + L(GQ'_{i-1})/2$. This formula is simple and efficient because only two numbers calculated from the previous step are involved. Number of distance comparisons is reverse linear formula rather 2^{i+1} comparisons as in the direct approach. The direct approach calculates the distances between all n nodes where, $i = \lceil(\log(n) + 1)/2\rceil$. Figure 4 shows simple example of GQ_3 creation from 8 nodes. It starts by creating 4 GQ_1 through linking every node to (the most) nearby neighbor, the cube center of gravity is marked by ×. Next, 2 GQ_2 are created by pairing the nearest 2 GQ_1. Finally, 1 GQ_3 is shown by linking 2 GQ_2.

When the local GQ_i is constructed, the cluster head attach the remaining non GQ_i nodes to the cube. MANET is a dynamic network and the requirement of static period to execute the algorithm limits our RD architecture applicability. Thus to cope to dynamic changes in network topology, every GQ_i node broadcast periodically *Hello* message and every non GQ_i node that receives for the first time this *Hello* message, chooses the sender as its *access node* to GQ_i and forward the message to the nearby non GQ_i neighbors. The access node is responsible of receiving resource advertisements and lookup from every non GQ_i node attached to it. The identified gateway nodes from clustering are used to connect all local Gray cubes to form virtual network backbone that consist of Gray cubes nodes, access nodes and gateway nodes. We do not attempt to minimize number of nodes in the network backbone; all the available gateways nodes are identified and used to generate fault-tolerant backbone.

In the previous example, only 3 cubes of similar dimensions GQ_3 are shown for simplicity reasons only, but this is not the general case. The dimension i of GQ_i is selected locally based on several parameters to avoid overload. Overloaded nodes deplete their energy resources quickly and cause severe performance degradation. The local GQ_i load is measured by many parameters that include resources usage, cube dimension and local topology changes. These parameters change with time, thus the dimension i is adjusted locally with any global network assumptions to cope with load variations. The resource distribution degree is controlled by i; when the load increases, i increased to provide higher degree of resource distribution. When the load decreased, lower degree of resource distribution is needed to minimize the average route length and hence i decreased. Instead of considering the load of all the local backbone nodes, we consider only γ nodes out of 2^i nodes to measure the local GQ_i load. The integer constant γ represents number of nodes of GQ_i with highest load. Let α and β be the highest and lowest load threshold respectively, when the load of γ nodes exceeds α, the current dimension is increased and when the load of γ nodes drops below β, the current dimension of is decreased.

The local GQ_i is reconstructed by the cluster head node which always holds the local address 0. In the case of increasing the cube dimension from GQ_i to GQ_{i+1} two options are available. The first option: the cluster head selects 2^i nodes from the local members not already members of GQ_i and connect them to get new GQ_{i+1}. The newly selected nodes are linked through adding new $k + 1$ edges and increasing the virtual address length by one. The address of $v \in GQ_i$ is increased by appending 0 to the left, while the address of newly selected node will get 1 at the leftmost bit of its address. This option and readdressing technique is simple but usually generate ill mapped Gray cube to the current physical layout. The second option: reconstruct new GQ_{i+1} from scratch by disregarding the old cube and considering all cluster's nodes. The reconstruction process is similar to the construction process previously explained in this section. Decreasing the dimension of GQ_i to GQ_{i-1} require the cluster head to select and remove 2^i nodes from GQ_i. Similar options to the case of increasing the cube dimensions are applied here also.

Proposed RD architecture: In our RD architecture, every Gray cube node is a directory agent capable of handling resource registration and lookup queries. We use DHT to assign and locate resources to these distributed directory agents of the network backbone. DHT is an efficient search structure based on resource's name. All the directory agent holds one distributed hash table that takes an input key k, calculates the hash function H on the key H (k) and uses the result as an index into the table. For simplicity, we consider only the resource name from the resource attributes. Our H function is simple and hash the name's characters (ASCII characters) to create binary string. Further enhancements to the architecture can be done through considering other attributes and their semantics. The hash function H takes the published or looked resource name as input and produce the virtual address of the directory agent (*H (resource name)→node address*). In resource publishing, the identified agent will register the resource in its local table by storing resource attributes and location. In resource lookup, the identified agent will reply by resource's location. In summary, the client calculates H (k) to get the agent's address, communicate with the agent to find the server location and finally starts communicating with the server.

Since the Gray cubes dimensions may vary from one region to another, we use an upper bound constant (Ψ) on the maximum Gray cue dimension (i.e., virtual address) created in the network. We relate the parameter Ψ to the maximum created cluster diameter ($2d$) used in the underlying cluster algorithm AMHC, where any node is at most d hops away from its cluster head, as $\Psi = \omega d$ where ω is an integer constant with value based on the current network density. The parameter ω is adjusted dynamically to adapt to changes in the topology. The hash function H generates Gray cube directory agent address of Ψ bits exactly. In any local GQ_i, only the rightmost i bits of Ψ address are used. As an example, for $\Psi = 4$ and resource registration (name = *Printer*), we have H(*Printer*) → 0101 where 0101 is a directory agent address.

Figure 5 shows the case when node x wants to register its resource R. First the node x applies the hash function H on R to get the directory agent's address on the network backbone. Then, the node x sends

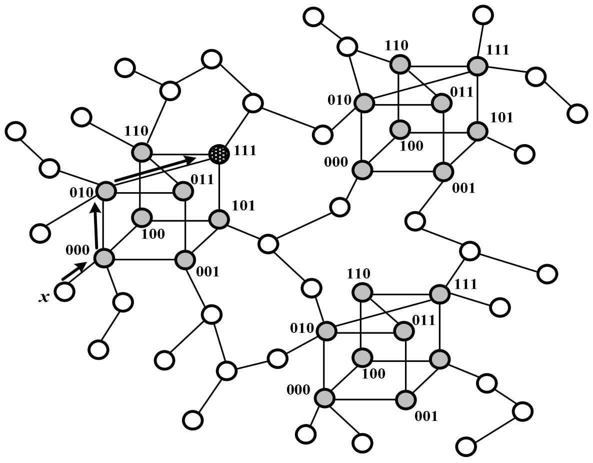

Fig. 5: Resource registration example

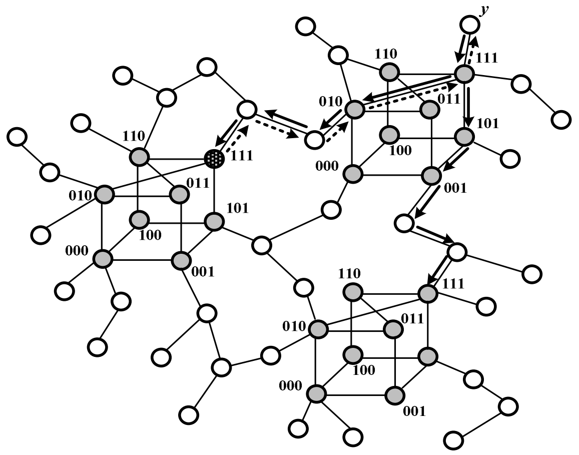

Fig. 6: Resource lookup example

registration query of R to the node with address $H(R)$ through the network backbone. In this example, $H(R) = 111$, so x forwards the registration query to the associated access node (which is also a local Gray cube node) and the query is forwarded in two hops to the directory agent with Gray cube address 111.

Example of resource lookup is shown in Fig. 6. When node y search for a resource R, it uses $H(R) = 111$ to get the directory agent holding the resource. In this example, node y sends query to the associated access node. Since the local node with address 111 does not have the required resource in its directory, it responds negatively to y and forward the query to the adjacent Gray cubes through the network backbone. The query is propagated through the network where every directory node with local address 111 responds positively or negatively to the query. Query propagation is limited by Time-to-Live (TTL) and Query Count to prevent queries loop-back based on the clustering parameter d.

The proposed RD network backbone is based on the Gray cube structure which allows congestion control, load balance, multipath routing and fault-tolerance. Congestion control is necessary to avoid network performance degradation due to increased packet loss and higher end-to-end delay generated mainly from node's mobility. Congestion in infrastructure-based networks generated mainly from buffer overflow or longer queuing delay. Load balanced is achieved by redirecting data packets from congested routes to non congested routes. The Gray cube offers multiple paths between every two nodes and thus when a client notices increased packet loss or higher end-to-end delay, the path is considered congested and packets are switched to another non congested route. Fault-tolerance and robustness are important requirements to alleviate the side-effects of node failures due to mobility or energy depletion. Our RD architecture uses distributed directory agents where the failure of some agents does not cause local or global system failure.

RESULTS AND DISCUSSION

The simulation model conducted using NS-2 (Network Simulator 2) to represent 100 to 2000 MANET nodes with mobility range between 10 and 20 m/s on area of 3000 m×3000 m. We used AMHC multi-hop clustering to divided the network into several non-overlapping clusters with parameter d = 5. Every simulation scenario of an RD protocol takes 10 min. We compared our RD architecture (GRAY) with the following five RD model:

- **SFP (Oliveira *et al.*, 2005):** A flooding-based RD protocol that has no resource registration queries. Only resource lookup messages are used to find required network resources. SFP floods the entire network.
- **DSDP (Kozat and Tassiulas, 2003):** RD is protocol based on connected dominating set S that represent a subset of network nodes where for each node $i \notin S$has at least one direct neighbor that belongs to S. Members of S are distributed directory agents.
- **AMHC (Al-Ahmadi and Al-Dhelaan, 2012):** A multi-hop is clustering algorithm that divides the network into non-overlapping areas where every node is at most d-hops away from its cluster head. The set of cluster heads forms d-hop connected dominating set. Local RD queries are carried by the cluster head as local centralized resource directory agent. All the clusters are connected by gateways to form network backbone where non-local resource queries are forwarded to other areas through the backbone.
- **DHT (Alvarez-Hamelin *et al.*, 2006):** RD is protocol that creates single overall incomplete hypercube in the network. The created hypercube is seen as distributed directory agents that handle resource registration and discovery messages. DHT uses hashing to distribute resources over this single hypercube.

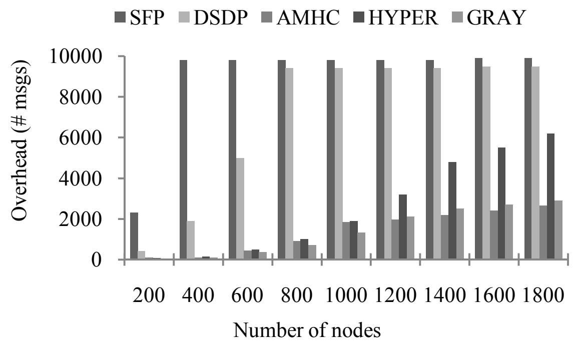

Fig. 7: Message overhead versus number of nodes

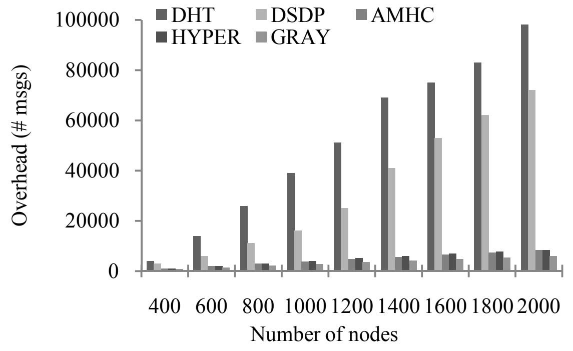

Fig. 8: Global message overhead

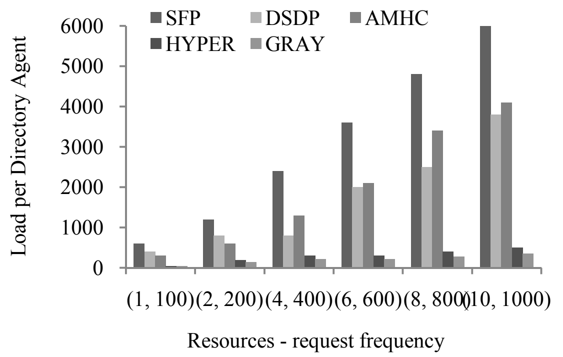

Fig. 9: Load per directory agent

- **HYPER (Dekar and Kheddouci, 2009):** This RD model is based on distributed hypercubes linked together to form network backbone. The clustering algorithm Max-Min is used to divide the network. HYPER uses hashing to distribute resources over these multiple hypercubes.

Among these RD protocols, SFP has no construction cost because it does not use resource directories. On the other hand, SFP uses flooding which creates many problems such as: slow resource lookup, congestion, non scalable. DSDP uses connected dominating set which generates large set of distributed directories for even small MANETs. AMHC clustering generates lower set of distributed directories compared to DSDP. In DHT, the construction phase of the general hypercube is very expensive in terms of communication cost and time latency. DHT provides very fast RD operations but the embedded cost in global network communication to build the underlying structure is very high. HYPER avoids the high construction cost of DHT by dividing the single hypercube to several hypercubes through clustering.

The overhead of RD protocol is measured by the number of generated messages (advertisement and search messages) for certain network size. Figure 7 shows the overhead involved in GRAY along with the selected protocols. SFP and DSDP generate large number of RD messages as the network size increase until the network is saturated and number of RD messages becomes stable. The generated overhead of HYPER is higher than AMHC and GRAY because the created hypercube has higher average path length and does not choose always the shortest path. GRAY overhead is almost comparable to AMHC but with the advantage of providing multi-paths property. When we increased the parameter *d* from 6 to 7, GRAY has lower overhead than AMHC. We did not include DHT in the comparison because it uses single general hypercube to cover the entire network which makes the overhead minimal compared to other methods but on the expense of extreme construction phase overhead. When we calculate the average overhead of the construction phase and discovery phase, GRAY has less overhead than DHT. There is no congestion or saturation in GRAY and HYPER due to their multi-paths property.

The previous overhead measures only the overhead involved in RD operations (advertisements and lookup), it does not show the global overhead involved by forming the underlying structure. Figure 8 shows the global overhead generated by topology construction and maintenance messages. SFP is not involved in this comparison because it does not use any underlying structure. It can be seen that DHT has the highest global overhead because building the overall hypercube require large number of messages exchange. DSDP has the second largest global overhead because of the large number of generated one-hop clusters (dominating sets) which require high construction and maintenance cost. GRAY and AMHC outperforms HYPER in topology construction and maintenance efficiency.

Overload avoidance is very important characteristic of resource discovery protocol. Figure 9 shows the load per directory agent participating in resource discovery process. The load is measured by a pair of two numbers (*x*, *y*) where *x* is the rate of client lookup messages per minute and *y* is the total number of network resources. During our simulation, every node with a resource (server) advertises at the rate of 5 messages per minute. During the simulation the client search rate (*x*) is varied from 1 search message per min to 10 search messages per minute. Number of resource in the network (*y*) is chosen to be 10 times the client search rate which is normal value in comparing RD protocols. SFP has the

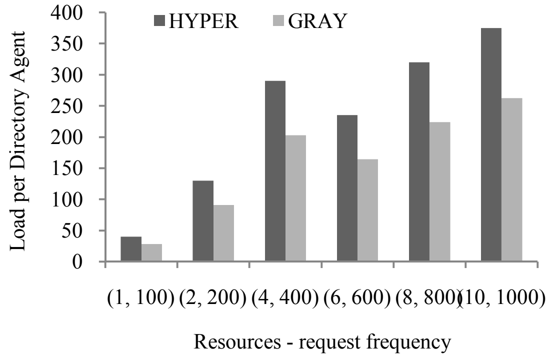

Fig. 10: Load per directory agent

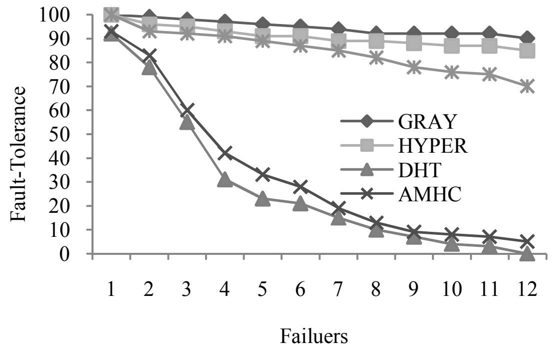

Fig. 11: Fault-tolerance versus failures

largest load per node because every node receives the same lookup requests many times as the requests replicated from node to node. In AMHC, all local resource publishing and lookup queries are directed to the cluster head. The cluster head load is very high compared to other node because it serves as local central directory for all nodes that exists within a diameter of 2*d*. There are differences between clustering and dominating sets protocols, but (in general) we can say that: the performance of DSDP protocol is comparable to the performance of AMHC protocol with *d* = 1. AMHC has the second largest load per directory node (cluster head) while DSDP has lower load per directory agent (dominating node). AMHC cluster head manages all resources that are within d hops away, while DSDP dominating node manages all resources that are within 1 hop only.

The remaining two protocols from Fig. 9, GRAY and HYPER, are shown in detail in Fig. 10. GRAY has the lowest load per directory agent compared to the HYPER. The strange reduction in the case of (6,600) is caused by the cube maintenance phase where the cube dimension is increased or decreased. When the load per directory agent exceeds certain threshold, the cube dimension is increased. Cube enlargement distributes the load among larger set of directory agents and hence the load per directory agent decreased. We excluded the protocol DHT from the comparison of load per directory agent because it considers every node in the network is considered as client, server and directory agent at the same time. The remaining methods consider only subset of the network nodes as directory agents. The incomplete global hypercube allows direct resource requests to be forwarded to the servers without the need of intermediate directory agents.

Fault tolerance is an important characteristic of RD protocols. MANETs suffer from nodes failures due to energy and/or mobility reasons. If the failed node is a directory agent holding information about some resources, then the resource discovery protocol is affected and its performance is degraded. In MANET, variations in resource discovery performance occur frequently and should be measured. Figure 11 shows the fault tolerance of GRAY and the selected protocols. GRAY, HYPER and DSDP have good load distribution of resources and hence good fault tolerance degree. GRAY maintains good performance and more fault tolerant compared to DSDP and HYPER. The performance of AMHC and DHT protocols decrease significantly when the rate of node failures increases. This performance decrease is caused by limited directory distribution in AMHC and the single structure used in DHT protocol.

CONCLUSION

In this study, we proposed new resource discovery architecture (GRAY) designed for dense and large scale ad hoc networks. It creates virtual network backbone based on connected Gray cubes created in several non overlapping network localities. The Gray cube provides multiple communication paths with low average distance. GRAY uses distributed hashing tables for efficient resource registration and search requests. At this stage we used only the resource name as hashing function parameter leaving the door open for further enhancements in GRAY resource distribution. GRAY is efficient, scalable, load balance and fault tolerant. It has low RD overhead because resources are distributed over the network backbone with fast DHT location identification. The generated overhead is on average one message per resource in every network locality. As more nodes join the network, the virtual backbone structure scales well and keeps its excellent behavior as efficient RD structure. The GRAY multi-paths feature allows load balance by redirecting data packets away from congested routes. The overall resource discovery load is spread over the set of distributed directory agents which allow load management. Overload avoidance increases network life time and fault tolerance despite the frequent failure of some MANET nodes. We evaluated the performance of GRAY as well as some of the well known protocols in the literature. Our modeling and simulation results demonstrated the superiority of the GRAY in terms of load balancing, scalability and fault-tolerance.

ACKNOWLEDGMENT

The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through research group no. RGP-VPP-264.

REFERENCES

Al-Ahmadi, S., A. Al-Dhelaan and N. Al-Hosini, 2011. New data gathering scheme for large scale wireless sensor networks. Proceeding of the 15th WSEAS International Conference on Computers. Corfu, Greece, pp: 111-117.

Al-Ahmadi, S. and A. Al-Dhelaan, 2012. An adaptive clustering-based resource discovery scheme for large scale MANETs. Proceeding of the 11th WSEAS International Conference on Telecom and Informatics. Saint Malo, France, pp: 88-95.

Alvarez-Hamelin, J., A. Viana and M. de Amorim, 2006. DHT-based Functionalities using Hypercubes. Ad-Hoc Networking, Springer, US, pp: 157-176.

Amis, A.D., R. Prakash, T.H.P. Vuong and D.T. Huynh, 2000. Max-mind-cluster formation in wireless ad hoc networks. Proceeding of the 19th Annual Joint Conference of the IEEE Computer and Comm. Societies (INFOCOM 2000), 1: 32-41.

Arnold, K., R. Scheifler, J. Waldo, B. O'Sullivan and A. Wollrath, 1999. Jini Specification. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.

Bettstetter, C. and C. Renner, 2000. A comparison of service discovery protocols and implementation of the service location protocol. Proceeding of the 6th EUNICE Open European Summer School: Innovative Internet Applications.

Chakraborty, D., A. Joshi, Y. Yesha and T. Finin, 2006. Toward distributed service discovery in pervasive computing environments. IEEE T. Mobile Comput., 5(2): 97-112.

Chuang, P.J., B.Y. Li and T.H. Chao, 2007. Hypercube-based data gathering in wireless sensor networks. J. Inform. Sci. Eng., 23(4): 1155.

Dekar, L. and H. Kheddouci, 2009. A resource discovery scheme for large scale ad hoc networks using a hypercube-based backbone. Proceeding of the IEEE 23rd International Conference on Advanced Information Networking and Applications. Bradford, UK, pp: 293-300.

Eriksson, J., M. Faloutsos and S. Krishnamurthy, 2004. Scalable ad hoc routing: The case for dynamic addressing. Proceeding of the 23rd Annual Joint Conference of the IEEE Computer and Comm. Societies (INFOCOM 2004), pp: 1108-1119.

Guttman, E., 1999. Service location protocol: Automatic discovery of IP network services. IEEE Internet Comput., 3(4): 71-80.

Kozat, U.C. and L. Tassiulas, 2003. Network layer support for service discovery in mobile ad hoc networks. Proceeding of the 22nd Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies, 3: 1965-1975.

Locher, T., S. Schmid and R. Wattenhofer, 2006. Equus: A provably robust and locality-aware peer-to-peer system. Proceeding of the 6th IEEE International Conference on Peer-to-Peer Computing, pp: 3-11.

Manoharan, R. and P. Thambidurai, 2006. Hypercube based team multicast routing protocol for mobile ad hoc networks. Proceeding of the 9th International Conference on Information Technology (ICIT'06), pp: 60-63.

Meshkova, E., J. Riihijarvi, M. Petrova and P. Mahonen, 2008. A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. Comput. Netw., 52(11): 2097-2128.

Oliveira, R., L. Bernardo and P. Pinto, 2005. Flooding techniques for resource discovery on high mobility MANETs. Proceeding of the Workshop on Wireless Ad-hoc Networks.

Olszak, A., 2010. HyCube: A DHT routing system based on a hierarchical hypercube geometry. Proceeding of the 8th International Conference on Parallel Processing and Applied Mathematics (PPAM'09). Springer-Verlag, Berlin, Heidelberg, pp: 260-269.

Peng, W. and X.C. Lu, 2000. On the reduction of broadcast redundancy in mobile ad hoc networks. Proceeding of the 1st ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp: 129-130.

Richard III, G.G., 2000. Service advertisement and discovery. IEEE Internet Comput., 4(5): 18-26.

Rowstron, A. and P. Druschel, 2001. Pastry: Scalable, Decentralized Object Location and Routing for Large-Scale Peer-to-Peer Systems. In: Guerraoui, R. (Ed.), Middleware 2001. LNCS Vol. 2218, Springer, Berlin, Heidelberg, pp: 329-350.

Sailhan, F. and V. Issarny, 2005. Scalable service discovery for MANET. Proceeding of the 3rd IEEE International Conference on Pervasive Computing and Comm. (PerCom 2005), pp: 235-244.

Salutation Consortium, 1999. Salutation architecture specification. The Salutation Consortium Inc. Retrieved from: http://ftp.salutation.org/salute/sa20e1a21. ps.

Schlosser, M., M. Sintek, S. Decker and W. Nejdl, 2003. Hypercup: Hypercubes, Ontologies and Efficient Search on Peer-to-Peer Networks. Lect. Notes Comput. Sc., Vol. 2530, Springer, Berlin, Heidelberg, pp: 112-124.

Serbu, S., P. Felber and P. Kropf, 2011. HyPeer: Structured overlay with flexible-choice routing. Comput. Netw., 55(1): 300-313.

Stoica, I., R. Morris, D. Karger, M.F. Kaashoek and H. Balakrishnan, 2001. Chord: A scalable peer-to-peer lookup service for internet applications. ACM SIGCOMM Comput. Commun. Rev., 31(4): 149-160.

Wang, L. and S. Olariu, 2004. A unifying look at clustering in mobile ad hoc networks. Wirel. Commun. Mob. Comp., 4(6): 623-637.

Williams, B. and T. Camp, 2002. Comparison of broadcasting techniques for mobile ad hoc networks. Proceeding of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp: 194-205.

Zahn, T. and J. Schiller, 2005. MADPastry: A DHT substrate for practicably sized MANETs. Proceedingof the 5th Workshop on Applications and Services in Wireless Networks (ASWN).

Zhao, B.Y., L. Huang, J. Stribling, S.C. Rhea, A.D. Joseph and J.D. Kubiatowicz, 2004. Tapestry: A resilient global-scale overlay for service deployment. IEEE J. Sel. Area. Comm., 22(1): 41-53.