Scientific Journal Impact Factor (SJIF): 1.711
International Journal of Modern Trends in Engineering and Research
ISSN (ONLINE) : 2349 - 9745 ISSN (PRINT) : 2393 - 8161

A New Coordinates based caching strategy CBC in mobile ad hoc network (MANT)

Aseel F. Marzook [1], Ahmed I. Saleh [2], Mohammed F. Al Rahmawy [3]
[1,2]*Mansoura University Faculty of Engineering*
[3]*Mansoura University Faculty of computer and Information*

Abstract—In this paper, we propose an efficient caching strategy for mobile ad hoc networks (MANET) called coordinator based caching strategy CBC. In this scheme, multiple hosts cooperate to improve data accessibility, reduce the time delay and reduce the number of messages flooded in to the network to find the requester data. In the CBC strategy, a non-overlapping clustering architecture is used for network organization and intra-local zone and inter-local zone requests are defined to increase the overall data accessibility. The effects of cache size, mobility, quarry generate time on the network performance were investigated using the discrete event simulation environment NS2. Simulations show that CBC caching has made considerable improvement in cache hit ratio, massage overhead and average query latency in comparison with other existing caching strategies.

Keywords—mobile ad hoc network, cooperative cache, cluster, local zone, network map

I. INTRODUCTION

MANETs aims to communicate mobile nodes without any support from an existing infrastructure of the network and without a need to any central administration. Therefore it is useful in places where traditional infrastructure does not exist, e.g. military battlefield, emergency services (e.g. firefighting o disaster recovery), in places with non-existing or damaged communication infrastructure and where fast spread of a communication network is required. MANETs are also used in business meetings and conferences to confidentially exchange data and in the library to access the Internet with a laptop. Also, several interesting applications are proposed especially for highly dynamic ad-hoc environments (VANETs)[1]. Although ad hoc networks offers attractive network opportunities, data management in MANET environment suffers from the following limitations [2]: (i) Highly dynamic network topology, where the mobile nodes move very freely, which may cause frequent network partitions and unpredictably and this makes data communication among those partitions difficult. (ii) Frequent failure in the routing paths as the nodes may be disconnected due to shutdown or it goes beyond the transmission range of other nodes (iii) No guarantee for reliable data communication and data consistency. Even when the network is connected and routing path exists between source and destination, there could still be such problems as network congestion and packets dropping, which are caused by limited network bandwidth and limited node processing and storage capability. Due to these reasons, data accessibility is not guaranteed. Therefore, data management in MANET is much more challenging than expected.

A lot of the researches on MANETs suggest dynamic routing protocols for enhancing connectivity strength among clients. Although routing is an important issue in MANET, other issues such as data availability are very significant, as the final goal of using such networks is to supply data to mobile hosts [3]. There are still many challenging issues in data accessing on MANETs due to frequent disconnections, high mobility of clients, limited bandwidth utilization and the poor power resources and small storage capacity. To defeat these challenges on data availability in MANET, data caching is one of the most attractive techniques that can enhance the efficiency of data availability

[4]. Data caching means that clients (nodes) copy some parts of data they need from the data source and cache them in their own memory. Due to caching or holding some pieces of data, the mobile nodes will not need to communicate with the server whenever it receives data requests. In this way, the total arrival delay is reduced with the help of these cache hosts.

Caching only is not sufficient to guarantee high data availability and low communication latency in dynamic systems with limited network resources, as mobile clients in MANET may have similar tasks and share common interest, cooperative caching which allows the data and information to share and collaborate each other among multiple nodes, can be used to enhance the bandwidth utilization, reduce power resource consumption of mobile nodes and reduce access latency. Furthermore, cooperative caching technique allow mobile caching nodes to coordinate the caching data tasks such as the redirection of data requests, cache placement, cache replacement and cache invalidation process [5] .

In this paper, we investigate the data retrieval challenge of MANETs and propose a novel scheme called Coordinate Cased Caching (CBC) strategy. The goal of CBC is to enhance the data availability using a cluster based cooperative caching technique, reduce the query delay, make the number of hops between the data centre and the requester as few as possible and ensuring efficient data dissemination among the group members of the MANET and improving the performance and efficiency of data access. In CBC, we assume cluster and zone based partitioning of the network. In cluster partitioning, the network is divided into non-overlapping clusters with square shapes. Each cluster has a cluster head that is adaptively changed and it maintains two tables namely the Internal Cache Table (ICT) and the External Cache Table (ECT). each mobile node has a network map to handle mobility handling. In zone partitioning, mobile clients are accessible in one hop, two type of nodes in a local zone, nodes in the same cluster and in adjoin clusters but in same zone, and nodes outside of it. To improve the efficiency of CBC caching, a Least Item Priority (LIP) based replacement policy has been developed. NS2 Simulations prove that CBC caching achieves higher performance than other strategy replacement

The rest of the paper is organized as follows. Section 2 gives an overview on cooperative caching is provided. Section 3 presents the related work. Section 4 presents the proposed coordinate based caching (CBC) strategy, section 5 discusses the cooperative cache scheme. Section 6 presents the cache management. Finally, in section 7 the performance of the strategy is evaluated.

II. BACKGROUND

This section overviews the fundamental technologies used in this research. Section 2.1 gives an overview of Cooperative Caching and ad-hoc Caching policies. Section 2.2 is an overview on a clustering in MANETs network

2.1 Cooperative Caching In MANT

Last few years have seen a revolution of mobile computing and wireless communication techniques. However, overcoming constraints such as frequent network failures/disconnections, limited bandwidth and limited local resources (energy) remains a challenge for effective data access in wireless network environments [6]. Data caching in mobile nodes is essential for improving the system performance, i.e. the terminal latency and energy consumption. However, the caching on individual systems cannot be scaled easily. Hence, cooperative caching based on the idea of sharing and coordination of cached data among multiple users can be particularly effective for information access in ad-hoc networks . Cooperative caching aims to assure delivery of data to a large population of users without the need for significant investment in servers. However, an efficient algorithm for distributed caching is important when network nodes have limited memory and energy. In addition, due to the high mobility and limitations in resources constraints in ad-hoc networks, cooperative caching techniques designed for wired networks are not appropriate for MANETs

The aim of this paper is to use cooperative caching mechanism in order to minimize the overhead and delay in MANETs, This mechanism definitely requires efficient management of caching. Caching Management is affected by three main issues: cache replacement, cache pre-fetching and cache consistency . Cache replacement algorithm is required when cache is full and need to delete or replace some data item to give place for the new coming data. The cache replacement is the core of the data caching. Hence, it is very necessary to create an efficient replacement algorithm [17]. On other hand, prefetching is important to improve caching performance. Prefetching fetches in advance the data items likely to be accessed in the near future to increase cache hit ratio and reduce user-perceived query latency.

2.2 MANET Clustering

The aim of clustering is to decrease the traffic during data routing process in the network. Clustering divides the network into various virtual groups based on certain rules to distinguish the nodes allocated to different sub-networks. Clustering enables easier management and enables scalability and enhances the availability in case of using large mobile networks. The nodes in any specific cluster-base network can be classified into four categories: cluster head, member nodes, gateway nodes and guest nodes [7]. These are defined next.

Cluster-head. A Cluster-head node (CH) is one of the most important nodes in the cluster and it work as local coordinator among all the nodes of its containing cluster and other nodes in other clusters or server, so the transmission range of cluster head characterizes the limitations of a cluster.

Gateway Nodes. Gateway nodes are nodes responsible for the transmission and the reception of information between nodes in different clusters, because it located at the boundary of the clusters.

Member Nodes. These nodes are sometimes called ordinary nodes; they are members of a cluster and these nodes have neighbors belonging to their own cluster.

Guest Node. Guest node is a node connected to a specific cluster.

A Cluster head is responsible for routing coordination. So, it contains the topology and routing information and it can pass it to other nodes; if required; member nodes don't have this information. Cluster algorithms are divided into two types: active and passive clustering algorithm. In active clustering algorithms [8], the nodes communicate with each other to exchange information to select the cluster head according to certain criteria, even if there is no data transmission. In passive clustering algorithms [18], clustering occurs only when transmitting data. The process of passive clustering has a main drawback when compared to active clustering, this is that is doesn't have main control on overheads and it needs to longer setup time which is significant for time critical application.

III. RELATED WORK

Many work has been conducted for developing caching strategies and techniques. This section introduces a brief description of some of the recent and the most widely used of those caching techniques.

The authors in [10] assumed three models of caching: cache data, cache path and hybrid cache. For the cache data model; if there are many requests for the same data item from several nodes, then this data item is popular. Hence, the intermediate nodes along routing path cache this popular data item and it serves any future requests directly from the cached data. The node doesn't cache the data if all the requests for the data are from the same node. Cache path model works in a similar way to cache data, but a node caches the path to the required data instead of caching the data itself. In cache path, a node doesn't record the path of all passing data, it only records the path when it is closer to the caching node than the data source. Hybrid Cache model combines the above two schemes to achieve better performance, where cache data is optimal when the data is small because the data needs a small amount of cache space; otherwise, cache path is preferable.

In [11], a zone cooperative scheme was adopted. Where each client has a cache to store the frequently accessed data items; if a data is missing in the local cache, the requester node first searches the data in its zone before forwarding the request to the next node that lies on a path towards server. However, the latency may become longer if the neighbors of the intermediate nodes do not have a copy of the requested data object for the request.

In[12], neighbor caching (NC) was presented. The NC scheme utilizes the available cache space of neighbor nodes to improve the caching performance. If a requester node fetches data from the node that has it, the requester node keeps data in its local cache if it has enough space; otherwise, the requester node calls the replacement algorithm and this data is cached in a neighbor node's storage; when the requester node needs the same data in the future, it is retrieved from the neighbor instead remote source

In [13], a scheme called *Group Caching* (GC) was presented. In this scheme, each node and all its one hop neighbors are consider a group and each node within group is given the same group member ID. A *Hello* message is used by each node to recognize its neighbor. Each node has two tables: a self-table and a group-table. A mobile node updates the group table whenever it receives notification of caching status from the group member. From the group table, the mobile node knows which data object cached in which group member; whenever a request is received to the mobile node, it can search its node table and its group table to find a record of the requested data object.

In [14], a global cluster cooperative caching (GCC) scheme was proposed. In this scheme, the network divided into a grid of equal sized clusters. Each node is given a cluster ID in addition to its unique node ID . When node needs a data, it first checks local cache; then, if it is not found, the client looks up the required data item at the cluster members. Only when the client cannot find the requested data in the cluster members' caches (called cluster cache miss), it requests the cache state node (CSN) which keeps the global cache state (GCS) and maintains the information about the node in the network which has copy of desired data item. If a cluster other than requesting nodes' cluster has the requested data (called remote cache hit), then it can serve the request without forwarding it further towards the server. Otherwise, the request is served by the server.

IV. THE PROPOSED COORDINATES BASED CASHING (CBC) STRATEGY

In this section, the details of the proposed Coordinates Based Caching strategy are discussed, where a set of assumption as well as the basic elements of the proposed CBC are explained. The basic assumptions for the proposed strategy are:

- The network is divided into several non-overlapping clusters, where each cluster contains a cluster head node, a gateway node and number of ordinary node.
- Each node has network map.
- The region of the transmission range of a certain node is called the local zone (LZ) and any node within a local zone is called a neighbored node (NN).
- Each mobile host mh_i, has an ID number, i.e., the set of mobile nodes M={mh_1 , mh_2 ,..., mh_{end} }, where *end* is the total number of mobile devices in the network environment.
- Each data item d_i , has an ID number, i.e the set of the used data D = {d_1, d_2, …… d_{num}}, where num is the total number of data items.
- The data items have different sizes.
- The mobile nodes have different cache spaces.

The basic elements used in the proposed CBC strategy are:
CBC Cluster: Clustering is a way to organizing an ad hoc network (i.e. the ad hoc network topology). In the proposed CBC, the network area is considered as a grid divided into clusters of the same dimensions. Each of these clusters is a square shaped region formed in column wise fashion. Each cluster has a unique number to identify the address of cluster within the network. All nodes within a single cluster can connect to each other in one hop communication, where each cluster has an ID number.
Local zone (LZ): is the last distance of transmission range. A node within LZ could be in one of three roles: data item requester node, neighbor node in the local cluster or in adjacent clusters that located within the transmission range of the requested node and cluster head
CBC map: is a map of coordinators for each cluster in the network. The objective of the map is to save the locations of all the nodes in a clusters.
Gateway node: A gateway node is a node able to identify other nodes in clusters other than its own containing cluster.

4.1 Cluster Formation (CF)

The main purpose of network clustering is to use the network resources more efficiently in order to enhance both the scalability and the availability and reduce communication overheads. Several algorithms have been proposed to cluster nodes. In our proposal, as shown in figure (1), the network is divided into non overlapped clusters. Each cluster has a square shape. Initially the nodes move randomly and distributed in the network area, then a random point (x,y) is selected as an origin, which is the starting point for network configuration, where X axis shifted in M value ,Y shift in M value to form the clusters in the network. Each node records the ID of its containing cluster in its local table. In each cluster, a node could be in one of three roles, cluster head, ordinary node or gateway node. Every cluster head (CH) contains a table of node information called external cache table ECT. The external cache table contains information about a node within different clusters, an example of the ECT is shown in table (1).

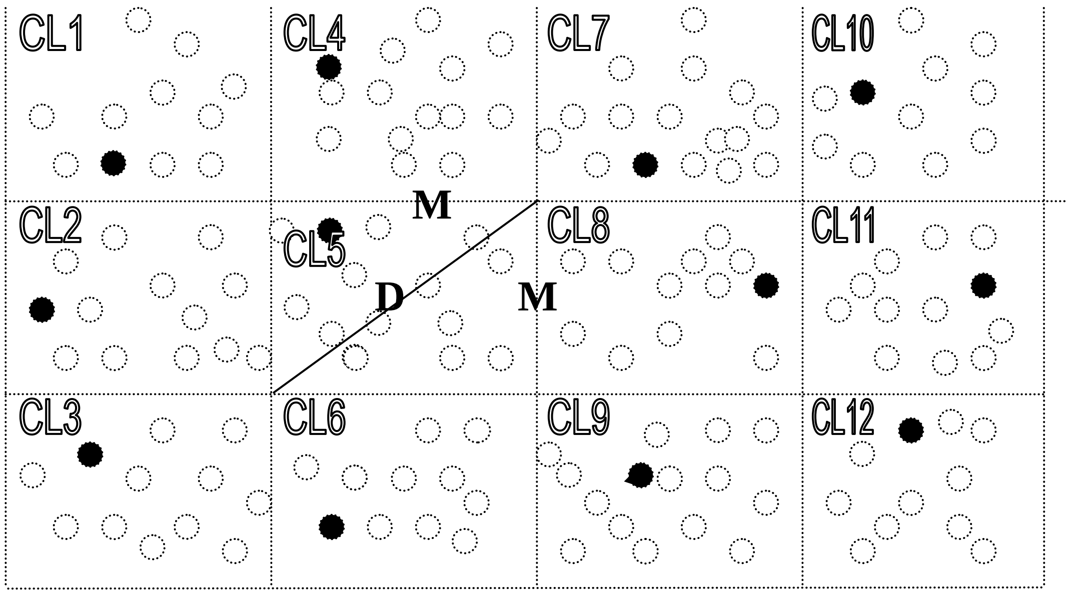

Figure 1. Network model

The clusters are named CL_1, CL_2, CL_3 ....in column wise fashion. All mobile nodes in a cluster can connect to each other in one hop communication. Each square-shaped cluster has a side length M, where M is derived as follows:

The largest possible distance D between any two node in the same cluster would never exceed the diagonal D of the square representing that cluster, where $D=M.\sqrt{2}$ for any square shaped cluster of side length M. To ensure one hop communication among all the nodes within a cluster, D should be less than or equal to the transmission range (R), i.e. $M.\sqrt{2} \leq R$. Hence choosing $M = R / \sqrt{2}$ would ensure that all nodes in a cluster can connect to one another in one hop communication.

Table 1. External cache table

Variable Name	Purpose
cluster id	to identify the address of cluster within cluster a network
Node id	to identify the address of a node within cluster .
status of node	to identify the current status of node (cluster head, ordinary node, gateway node)
Data item id	To identify address of data item that stored different node
TTL	Time-To-Live value, indicating how long an item can exist in cache.

Each cluster has a unique ID number, Each mobile node has a unique cluster ID, the node within transmission range is called neighbor node (NN). A requester node with all its neighbor nodes forms a local zone. As shown in figure (2) local zone covers the area of the transmission range of a mobile requester node. In each LZ, a node could be in one of three roles, requester node, node in the same cluster or a node in another cluster but in the same transmission range. Requester nodes use broadcast massages to request data item(s) in its LZ. If a neighbor node has the data, it replies with the actual data; otherwise, in case it replies with a null message. When the broadcast message reaches to the CH, the CH replies with an acknowledge of the node id that has the data item

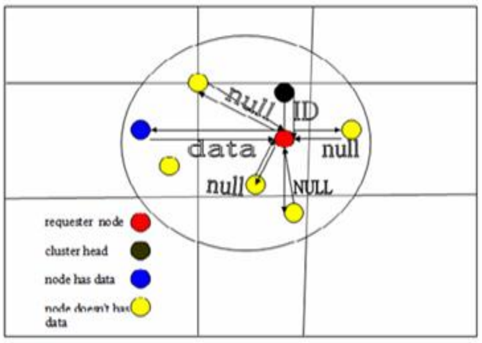

Figure 2. Local zone

4.2 Cluster Head Selection (CHS)

After cluster formation, the cluster head should be selected. In CBC strategy, the proposed **cache space, transmission range, mobility (CTMA) algorithm** is used. This algorithm is, as its name indicates, based on several factors such as the lowest node mobility, the highest cache space and the best transmission range, where a weight *w(n)* is calculated for each node as follows :

$$w(n) = (TR) . (Cs) / m \qquad (1)$$

where,

w(n) : the proportional weight of a mobile node
TR: the node's transmission range
Cs : cache space of the node
m: the rate of the mobility of node

A node that has the highest weight is chosen as a cluster head. After selecting the cluster head, it sends a broadcast message with its ID to all the member nodes in the cluster, when this message reaches the node members, all the members store the received cluster head's ID and send their cache information to the CH. Upon receiving the cache information from its cluster member, the cluster head creates its own table to store these data. Then the CH sends requests to other clusters' heads to send/receive its cache table information. When there is a change (add or delete) of the cache space, a node that has the change sends its update item list to its cluster head immediately. In this way, the table of information in cluster head can reflect the change of cached item within a cluster synchronously. Periodically, each cluster head sends the updates in its cache table information to update the cached data in the nearby clusters' heads hop by hop using AODV until the update reaches all clusters of the network

4.3 Mobility Handling(MH)

Mobility is an important component that should be addressed in our network due to the random mobility of the nodes. In the proposed CBC strategy, the mobility of nodes is classified into two main categories: Intra-cluster mobility and Inter-cluster mobility. In the first category, i.e. the intra-cluster mobility, a node moves within the same cluster; this kind of mobility causes no overhead in the network. In the second category, i.e. the inter-cluster mobility, a node moves out of its own cluster to a new cluster; the nodes of this category have to be handled efficiently to reduce the overhead. The inter-cluster mobility of a node is handled as follows:
Each node in a network has a network map that identifies the bounders of network clusters, as shown in figure (3); the network map stores information about cluster ID and coordinator for each cluster in a network

Cluster id	H1		H2		H3		H4	
	X1	Y1	X2	Y2	X3	Y3	X4	Y4
Cl1	0	4M	M	4M	0	3M	M	3M
Cl2	0	3M	M	3M	0	2M	M	2M
Cl3	0	2M	M	2M	0	0	M	0
Cl4	M	4M	2M	4M	M	3M	2M	3M
Cl5	M	3M	2M	3M	M	2M	2M	2M
Cl6	M	2M	2M	2M	M	0	2M	0
Cl7	2M	4M	3M	4M	2M	3M	3M	3M
Cl8	2M	3M	3M	3M	2M	2M	3M	2M
Cl9	2M	2M	3M	2M	2M	0	3M	0
Cl10	3M	4M	4M	4M	3M	3M	4M	3M
Cl11	3M	3M	4M	3M	3M	2M	4M	2M
Cl12	3M	3M	4M	2M	3M	0	4M	0

Figure 3. Network map

As shown in Figure (4), periodically each node asks the Global Position System (GPS) for its position. When a node checks its private network map (NM) and it finds that it entered a new cluster, it starts the notification procedure by sending broadcast request message with its location information to the one hop neighbors. The neighbors who belong to the same cluster replies with their cluster head ID. The neighbors residing outside the cluster ignore the request message. The first positive reply from a neighbor node makes the client sends its cache information to the cluster head in order to join the cluster.

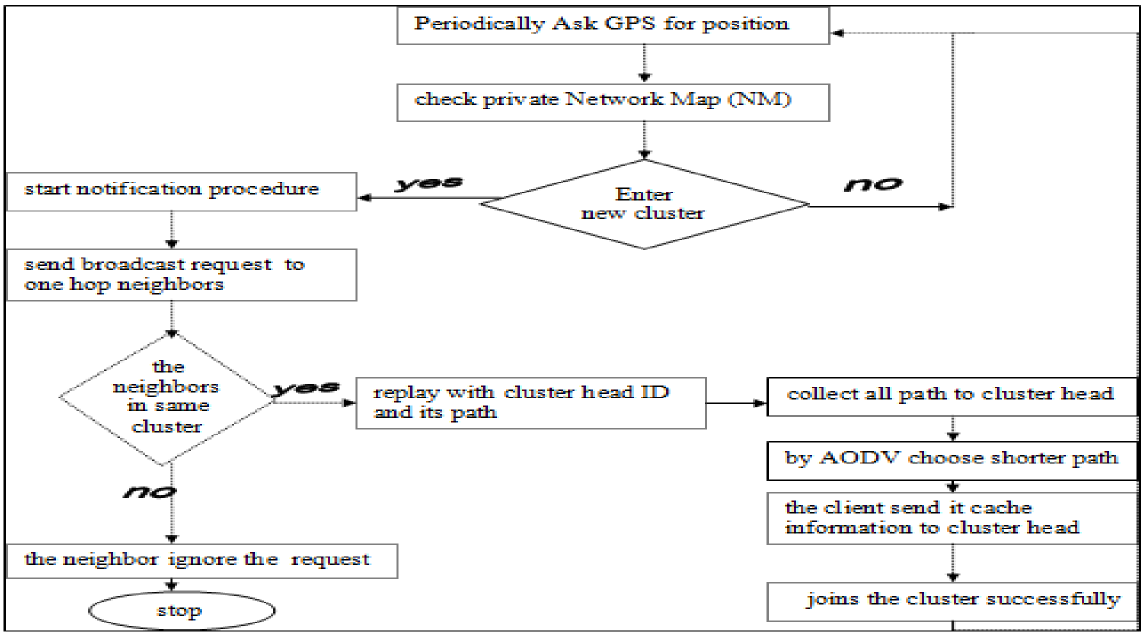
Periodically Ask GPS for position
check private Network Map (NM)
Enter new cluster
yes
no
start notification procedure
send broadcast request to one hop neighbors
the neighbors in same cluster
yes
replay with cluster head ID and its path
collect all path to cluster head
by AODV choose shorter path
the client send it cache information to cluster head
joins the cluster successfully
no
the neighbor ignore the request
stop

Figure 4. Flowchart for mobility management

4.4 Failure Handling (FH)

The failure in MANET can be due to several reasons, e.g. a node leaves the network. In CBC strategy, the failure reaction depends on the type of node failed; if a gateway node or ordinary node is failed, the failure is discovered and handled according to the steps shown in figure (5); first each cluster head periodically sends a "Hello" message to all the nodes within its cluster; if a mobile node doesn't reply to the hello message within a certain specified time, it means that the node either moved away from the network range or it has been shut down, so the cluster head updates its caching table and the record(s) corresponding to this failed node

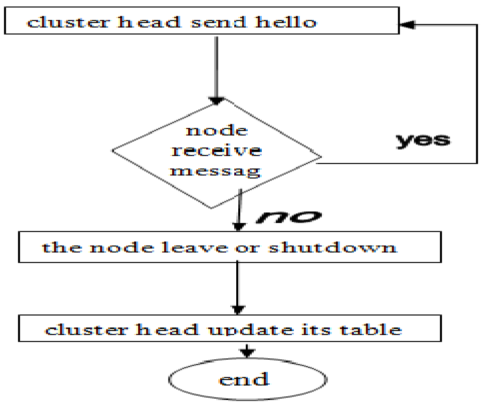

Figure 5. Flowchart for handling failure of ordinary node

In the case of a failure of a cluster head, the steps in figure (6) is applied if a cluster head does not respond to a node request within pre-specified time; the node sends a request to one or more of its neighbors, to confirm if a cluster head exists or no; if the cluster head doesn't respond to neighbors as well, then the requester node becomes assured that the cluster head is not available; hence, it initiates the cluster head election process by broadcasting a selection message to all its neighbors. Each node sends its quality information (as explained in section 3) as a reply to the initiator. The initiator chooses the node with best characteristics as new cluster head. Once chosen, the new cluster head sends a broadcast message to all members of its cluster to denote its new job. All member nodes update their identifier cluster head and send cache information to reflect the newly selected cluster head.

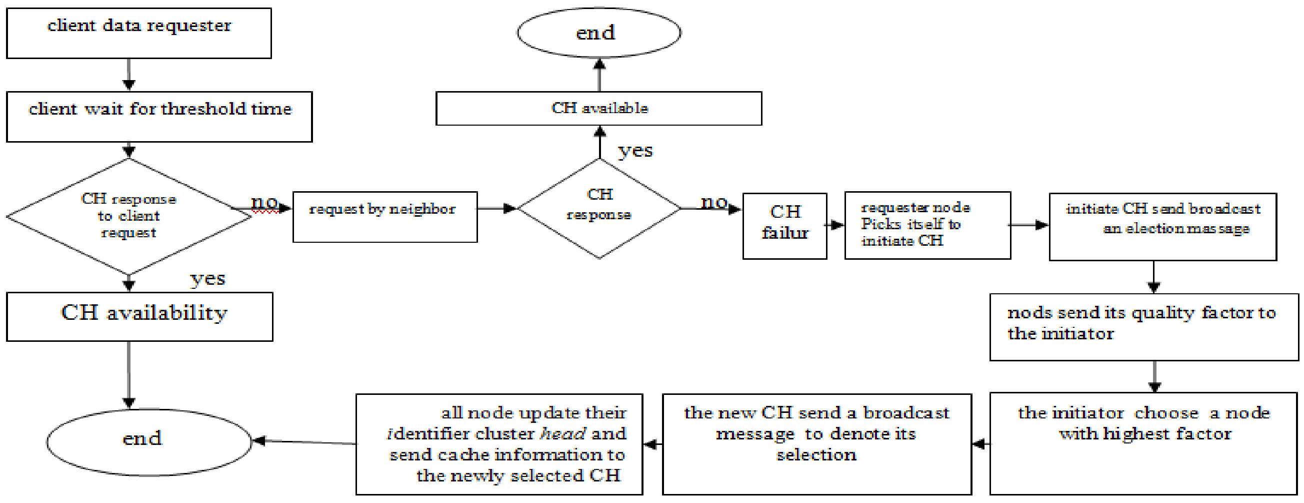
client data requester
client wait for threshold time
CH response to client request
no
request by neighbor
CH response
yes
CH available
end
no
CH failur
requester node Picks itself to initiate CH
initiate CH send broadcast an election massage
nods send its quality factor to the initiator
the initiator choose a node with highest factor
the new CH send a broadcast message to denote its selection
all node update their identifier cluster head and send cache information to the newly selected CH
end
yes
CH availability

Figure 6. Flowchart for handling failure of cluster head

V. COOPERATIVE CACHE SCHEME (CCS)

Cache discovery is the process of searching for and locating the required data item. This section describes our CCS that take advantage of the above described local zone mechanism for data retrieval in mobile ad hoc network. We use two type of requests based on how requested data is represented in the request:
ID-based request, in this types a client that requests a data item sends the ID of the data item and in turn, it gets a reply with an *acknowledgement* packet containing the *ID* of the client node which has the required cached item as well as the requester node sends request to cluster head
Content-based request, in this type a client that requests the data sends the ID of the required data item and in turn, it gets a *reply* packet containing the required item

As shown in Figure (7), CBC strategy defines the following related events:
A Local hit: when a node needs data, it first checks its own cache. If the data item is available in its local cache this event is called local hit.
A Neighbor/Zone hit: if the data item is not available it sends a broadcast request to the nodes within its transmission range (i.e. within the local zone LZ) and it waits till it gets a reply from these neighbor nodes, when the request reaches the neighbor node, it checks in its local cache if the data item is available; if the actual data is found the requester node receives the found item from the node in which it is cached, i.e. (content based request).
Remote hit: if the request reaches the cluster head, then the cluster head checks in its own external cache table (ECT) and it replies with the id of the remote node that has data (i.e. *ID-based request*).
Server hit: if the data is not found in the ECT of the cluster head, the request is processed by the server and that is called Server or global hit.

It is possible that the node sends broadcast request to all the nodes in its local zone; hence, two types of replies could be received: first, from one or more neighbor node(s) that has/have the actual data, second, from cluster head if one or more cluster heads has/have the id of the node that has the required data. In this case, when more than one node has the required data in the network, the requester node chooses the shortest path using the AODV protocol. In case the selected node is a node discovered by the cluster head, the requester node sends a *confirm* packet to the selected node whose *id* is returned by *cluster head* and this node responds with a *reply* packet that contains the requested data item. The *reply* packet contains the item *id,* the actual data and the *TTL, i.e. Content-based request* .

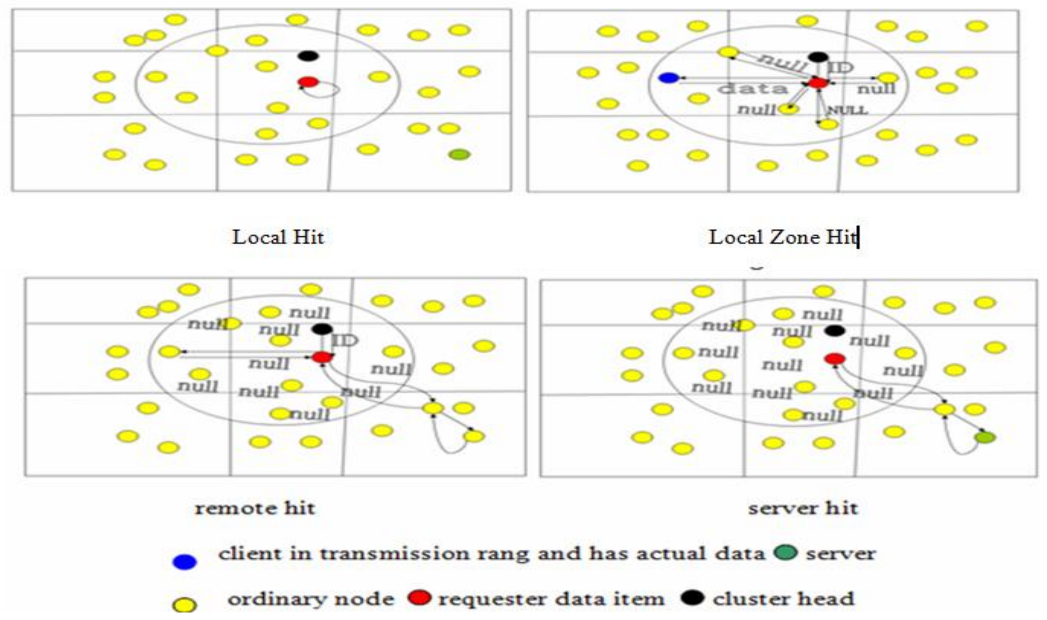

Figure 7. The cases of request data

Each client has a certain amount of cache space for caching the received data items. The efficiency of a cache replacement algorithm determines the probability of finding the requested data item in the cache space. When a client need a data item it check its cache, if the data item is available in its local cache, the requester node checks the consistency, when a data item is valid that means a local cache hit, which has the least communication overhead and the least query latency. Otherwise, the search process continues to the next step, i.e. sending requests to the entire nodes within the local zone. In the local zone, the client node searches the requested data item within its transmission range (i.e. intra local zone) by broadcast massage mechanism; the client sends a broadcast message and waits for a reply from any neighbor node NN, if a neighbor node is an ordinary node, in order to receive a broadcast of the requested data. It checks in its local cache, if the data item is available, it cheeks the consistency of this data item and it replies with the actual data (*Content-based request*) to the requester. Otherwise, the neighbor node replies with null massage.

On the other hand, if a neighbor node is a cluster head, it replies with the id of the client that has the required data. Hence, the requester node sends a unicast request message to the remote node of this client and the client responds with a *reply* packet (*Content-based request*) that contains the requested data item. The *reply* packet contains the items of *id data*, actual data and *TTL*; the requester node checks if a sufficient cache space is available, if it doesn't have sufficient cache space, the requester node calls the replacement algorithm. If the data is not found in ECT, the request is satisfied by the server.

Figure (8) shows an example of the CBC strategy; when (A) needs a data an item and this item is not found in the local cache; then, (A) sends a broadcast message in its local zone. A cluster head replies with an acknowledgement of the id of (B) that has the requested data; hence, (A) sends a request message directed to (B), finally (B) replies with a packet that has the actual data

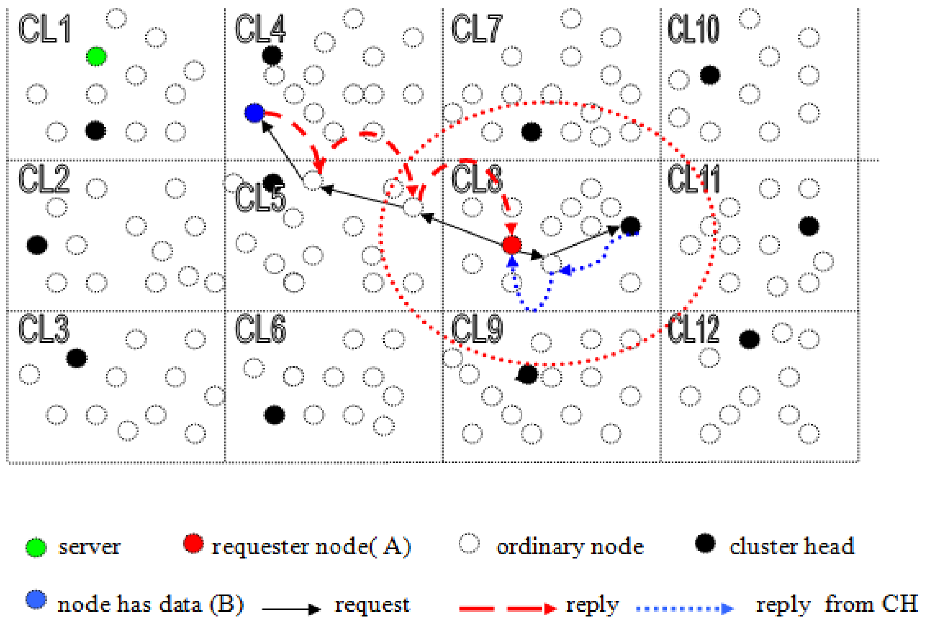

Figure 8. Example of cache data

VI. CACHE MANAGEMENT

The cache management process is divided into three sub-processes as follows.
Cache Admission Control(CAC). It is responsible for determining whether a received data item should be cached or no.
Cache Replacement(CR). It determines which cached item in the cache space should be removed when the cache is full and a new data item has to be cached;
Cache Consistency Strategy(C2S). It keeps the cached data items synchronized with the original data items in the data source. Most approaches adopt TTL (Time-To-Live)-based cache consistency strategy, in which every data item is assigned a TTL value representing how long this data item will exist in cache space, and the cached data item is considered valid until the TTL elapses.

6.1 Cache Admission Control (CAC)

Cache admission control is defined as a process that allows the mobile host to cache a data item based on the location of the data source or determine if there is another client that has the requested data to increase data accessibility. In the proposed CBC strategy, to reduce the query delay, the node caches all the received data items until its cache space becomes full to avoid fetching the requested item from the neighbor nodes as much as possible. After the cache space becomes full, the received data item is not cached if the data item has a copy within local zone (i.e. within the transmission range); otherwise, the received data item is cached. The next algorithm summarizes these steps.

Algorithm(1) Admission control (d)

```
1. A reply (d, TTL) is obtained from the data source
2- Check if the cache is sufficient or not
3- IF there is sufficient cache space THEN
        Cache the information into the cache space
4- ELSE
        Calculate the location of the data source (LDS)
        IF (LDS is within the zone of the requester node)
            Do not cache data
```

```
                ELSE IF (LDS is within a remote clusters)
                Call the replacement algorithm
                Cache the new item
                END IF
      END IF
```

6.2 Cache Replacement (CR)
In the proposed CBC strategy, a proposed priority based cache replacement policy is used; it is called *least item priority* (LIP). In LIP, one or more items are evicted out the local cache, where four factors are consider when computing the priority of a data item at a client.
1- Number of copies (N). Number of copies of the item in mobile host within transmission range, whenever the number of copies increases, this means that the item has a lower priority, i.e. the probability of replacing it is high.
2- Number of usage (Popularity). Number of times the item is used either internally from the same Node or externally from a neighbor Node. An item with lower access probability should be chosen for replacement

$$\text{Popularity} = \sum_{i=1}^{n} U_i$$

where U_i is the mean number of the data items used and N is the total number of data items in the cache table
3- Size of item (s). If the data size is large, then the item should be chosen for replacement. Because the cache can be replaced by more than one item
4- TTL. This is the time during which the data item is valid. An item valid for shorter time has always the highest probability of replacement.

Based on the above factors, an item priority function for data item is calculated as follows:

$$\text{Item-Priority} = Popularity.TTL/N.S$$

The item priority mechanism. When a requester node retrieves a data item from a client in remote cluster it inserts the data item in its own cache. In case the cache space is full, then there is a need to evict the least valuable item from cache based on the priority value defined by the above equation. The item with least priority is evicted if there isn't sufficient cache space and insert newly arrived information item into cache space the data item evicted may be cached in other node of the zone based on space availability . later if any mobile host of cluster need the data is request the data from one hop (neighbor) local zone member that keeps the copy of data instead requires it from remote client or server.

6.3 Cache Consistency Strategy(C2S)
Cache consistency issue must be addressed in the proposed caching strategy to ensure that clients only access valid states of the data. In general, Cache consistency is classified into the following two strategies:
- **Strong cache strategy**: it doesn't allow a stale data item to return to the user (client)
- **Weak cache strategy:** a stale data item may be returned to the user (client)

In our proposal, we used a weak strategy model based on the TTL. In this model, the client considers a cached copy up-to-date if its TTL has not expired; otherwise, the client removes the cached data when the TTL expires. A client forward the request to other clients or to the data center to retrieve the required data; when received, this data is cached as a fresh copy with an updated fresh TTL. [16].

VII. PERFORMANCE EVALUATION

7.1 Simulation Model

We developed a simulation model using NS2 to evaluate the proposed strategy. In the simulation model, the AODV is used as the ad-hoc network routing protocol. The total number of client is set to 70 in a fixed area of 1500m**x**1500m. The wireless bandwidth is assumed to be 2MB/s and the transmission range is 250m

In the simulation, like the simulation used in [16], groups of nodes are assumed to be distributed randomly in equal square areas (i.e. clusters). The clusters are arranged in columns named as CL1 ,CL2 ...., where each node selects a random destination and moves toward that destination with a random speed in the range(2m/s - 20m/s). Table (2) illustrates the details of the used simulation parameters.

Table 2 Simulation Parameters

Parameter	Default value	Range
Database size (*N*)	1000 items	
s min	1kB	
s max	10kB	
Number of clients (*M)*	70	50–100
Client cache size (*C)*	800kB	200–1400kB
Client speed (*v* min - v max*)*	2m/s	2–20 m/s
Bandwidth (*b)*	2Mbps	
TTL	5000 s	
Pause time	300 s	
Mean query generate time (*Tq)*	5 s	2–100 s
Transmission range (*r)*	250m	

7.2 Performance Metrics

In order to evaluate the proposed strategy, we use the *average query latency*, *cache hit ratio* and *message overhead* as performance metrics to compare the efficiency of the proposed strategy (CBC) against three well-known strategies; *non cooperative (NC) caching*, GCC[15] and *cache Data*[10]. These metrics are defined as follows:

Average Query Latency. It is the average of the *query latency* values measured of all the queries made in the network, where the *query latency* of a certain query is measured from the moment of sending the query from a requesting node to the moment of the return of the requested data from the data source.

Cache Hit Ratio. This is the percentage of the ratio of the number of the requested data that is found and retrieved from within local nodes (N_{local}) or zones (N_{zone}) or from a remote cache (N_{local}) to the total number of requested data (N_{total}), i.e.

Cache Hit Ratio = (Local Cache Hit+Zone Cache Hit +Remote Cache Hit), where

Local cache hit = N_{local} / N_{total},

Zone cache hit = N_{zone} / N_{total}

Remote cache hit = N_{remote} / N_{total}

Message overhead. It is the number of massages that Spread into the network to process a data request query from the moment of sending the request to the moment of receiving the data.

7.3 Results

In our experiments, the same data access pattern and mobility model are applied to all the three schemes and the performance measures are made to compare the proposed strategy with other strategies as stated above.

The results obtained are discussed in the following:

7.3.1 Impact of of Varying client Cache Size

To study the impact of cache size on both the average query latency and the message overhead, the cache size was assigned values in the range from 200 KB to 1400 KB and the measurements are taken and drawn in Figures9,10. In Figure 9, it is clear that the proposed CBC strategy performs the best among all the simulated strategies. This is expected when comparing the high byte hit ratio due to cluster cooperation. This is normal as there is no caching at all in case of the NC, which means. Also, the CBC strategy performs much better than Cache Data strategy as more required data could be found in both the local nodes and the local zones cache in case of using the CBC as compared to Cache Data which employs only the local cache. CBC strategy also performs much better than the GCC strategy as more required data could retrieved from the local nodes and the local zones, where the local zone covers nodes in the same cluster and in different clusters, i.e. the accessible caching area in case of the CBC covers more nodes than the GCC, which reduces the hop count, which in turn lowers the average query latency. Also, when the cache size is large enough, the nodes would be able to access most of the required data items from either the local node cache or the local zone cache which minimizes the query latency. From the figure, it is noted that CBC reaches its best performance when the cache size is 800 KB.

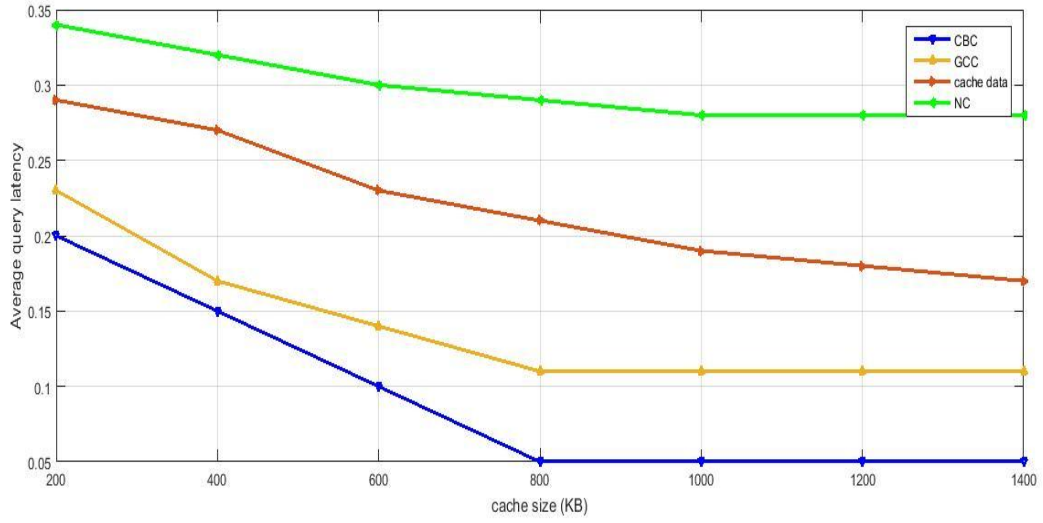

Figure 9. Impact of cache size on average query latency

Figure 10 shows that CBC has the minimum number of overhead messages among all the tested strategies. This is because CBC gets data from nearby clusters or zones caching the required data instead of remote data sources, which makes the data requests and replies messages move among smaller number of hops. Also, it is noted in the same figure that increasing the cache size reduces the message overhead further as the increase in the cache size increases the chance of finding the required data at nearby nodes and reducing the number of processed messages in turn.

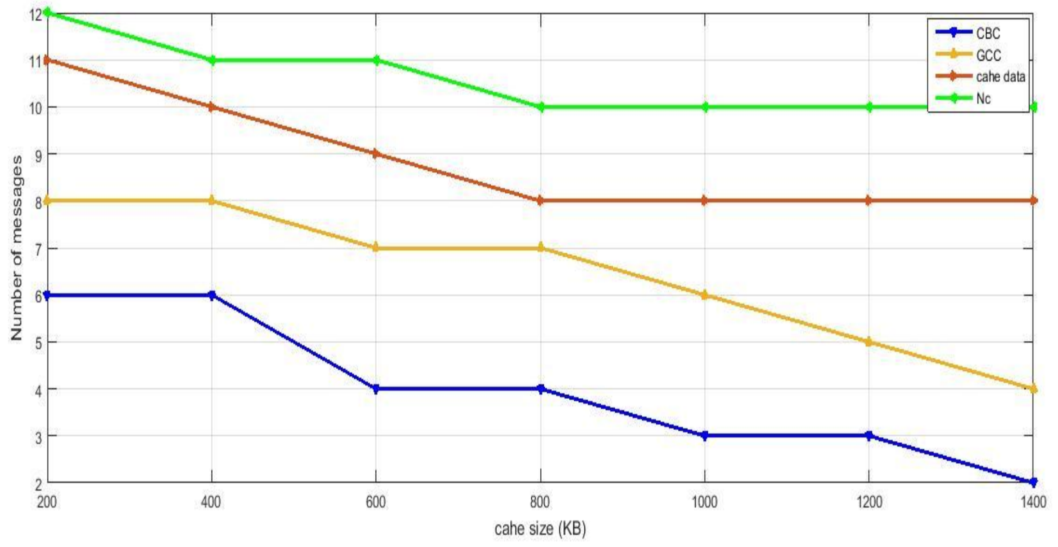

Figure 10. Impact of cache size on message overhead

7.3.2 Impact of different query generation times

The average query latency is measured in the simulation as a function of the mean generate time T_q and the results are drawn in Figure 11. The figure shows that, for all values of T_q, CBC strategy has better results than No Cache, Cache Data and GCC strategies. In general, small values of T_q leads to high rate of query generation, which increases value of the average query latency due to the increased amount of work. The figure shows that increasing the value of T_q leads to a decrease of the generated queries, which reduces the average query latency. The figure shows that if T_q keeps increasing, the average query latency continues to decrease; however sometimes it increases a little due to decrease in the cache byte hit ratio. .

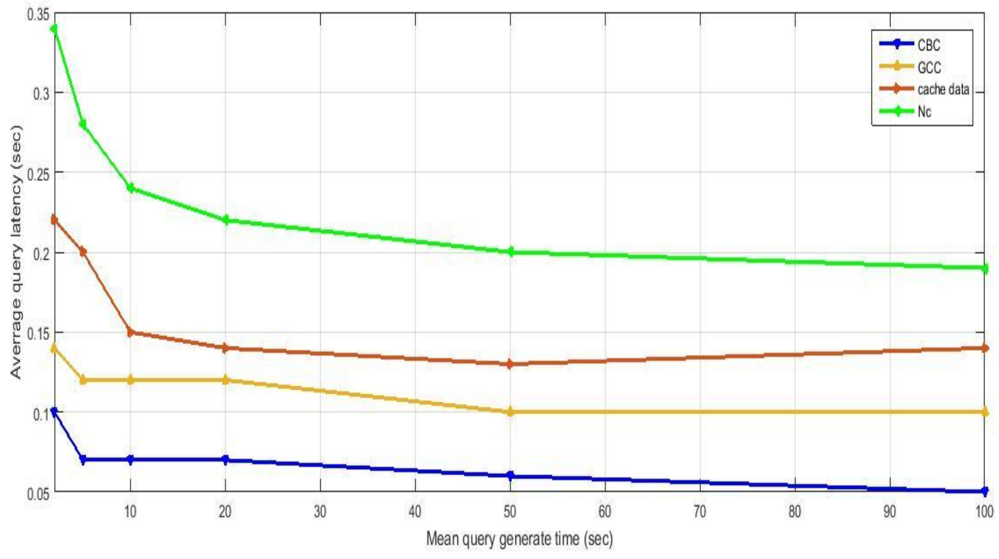

Figure 11. Impact of different query generation times on average query latency

Figure 12 shows that when the mean query generate time increases, CBC strategy has better results than NC, Cache Data and GCC strategies.

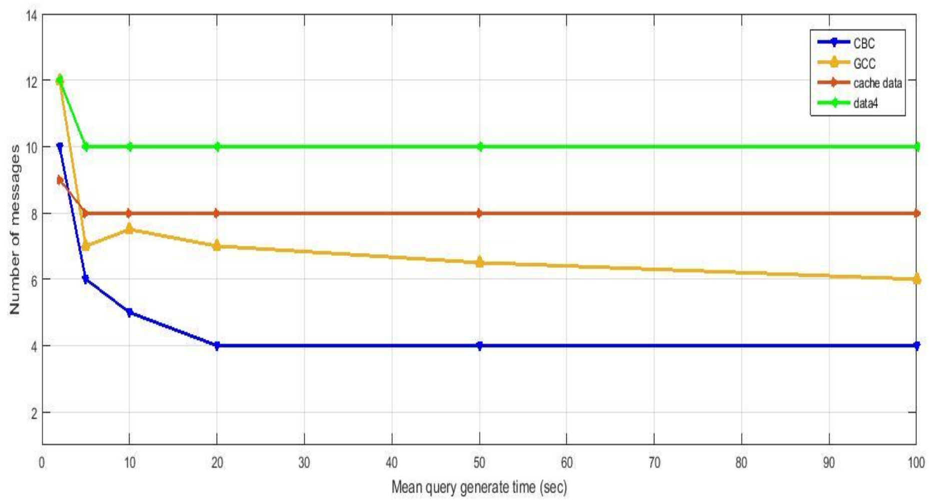

Figure 12. Impact of different query generate times on message over head.

7.3.3 Impact of Nodes' Mobility Speed

The impact of nodes' mobility speed on the average query latency is illustrated in Figures 13, 14, where each node is moving with a uniformly distributed in a range of speed values between 0 and a maximum value. The results are drawn for different assigned maximum values of the nodes' speed in the range from 2 to 20 m/sec. Figure 13 shows that the performance reduces with increasing the mobility regardless of the used caching strategy, this is due to the induced route failures and route re-computations resulting from the increase in the frequency of nodes with different data affinity leaving/joining a cluster.

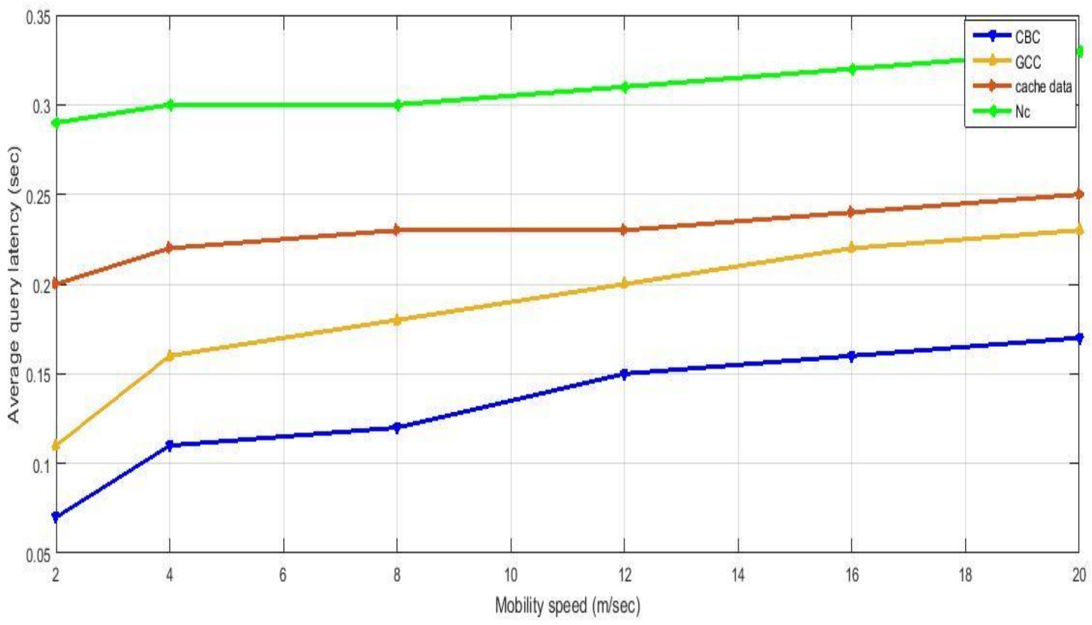

Figure 13. Impact of nodes'mobility on average query latency

For the same reasons, as shown in Figure 14, the message overhead increases with increasing mobility for all the caching strategies. Moreover, in case of the CBC, the number of messages increases due to the increased rate of the change/election of new cluster heads and the increases in the rate of updating cache states.

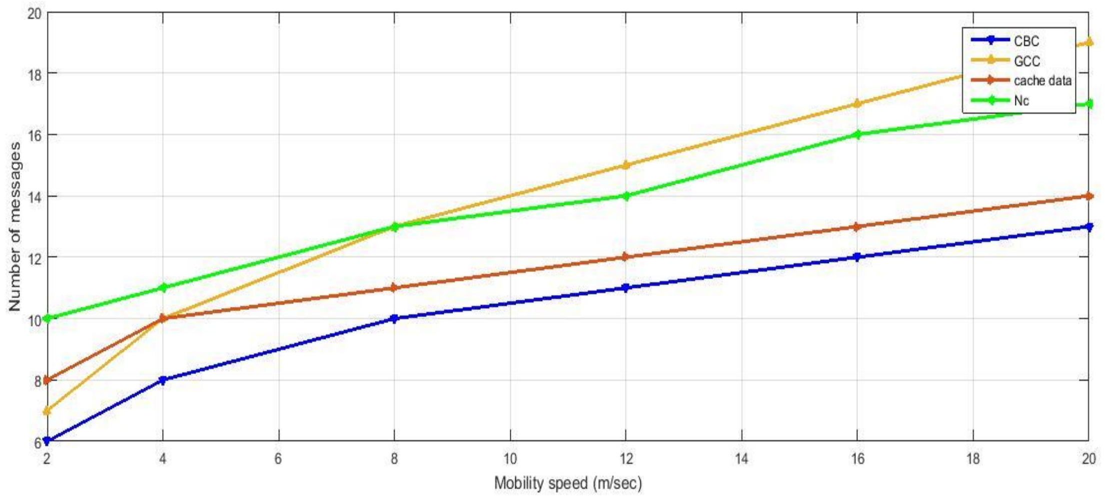

Figure 14.Impact of nods'mobility on message overhead

7.3.4 Impact of Transmission Range

Figure 15 shows that for all the caching strategies used, increasing the transmission range leads to reducing the message overhead. This is expected as increasing the transmission range leads to an increase in the predictable arrival of the packets to its final destinations as the packets are transmitted along few hops to reach their destinations. However, this leads to higher energy consumption per transmission.

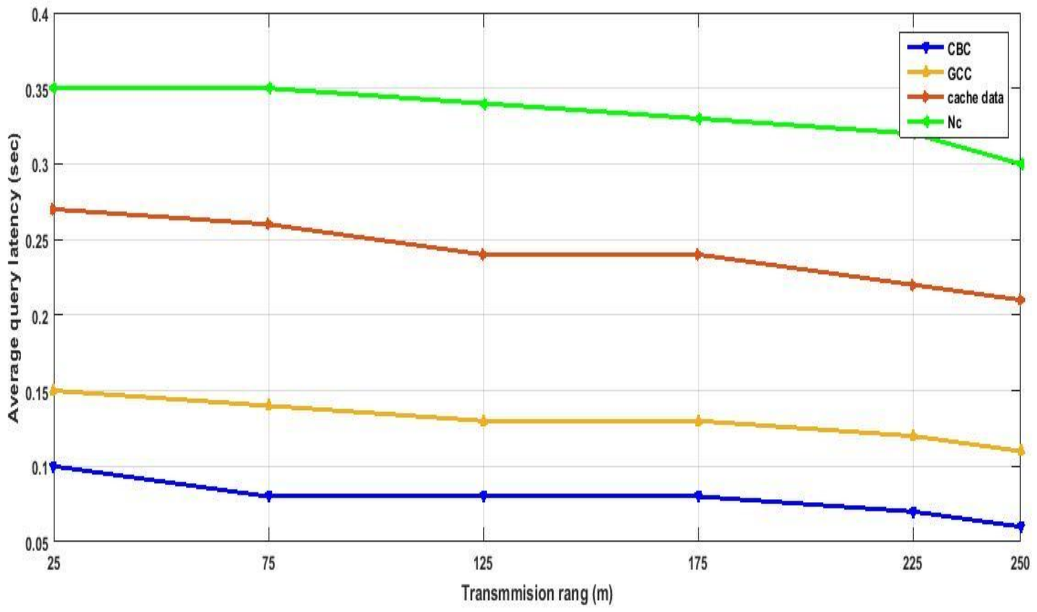

Figure 15. Impact of transmission range on quarry latency

For the same reasons, Figure 16 shows that, for all the caching strategies tested, increasing the transmission range decreases the number of overhead messages because the packets need smaller number of hops to reach their destinations.

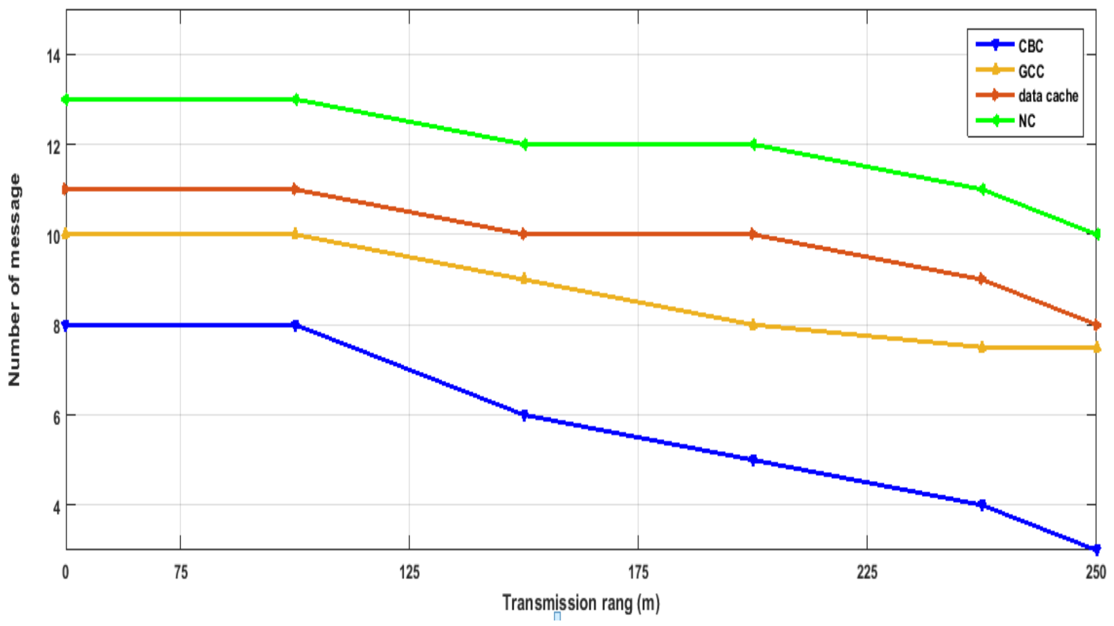

Figure 16. Impact of transmission range on message overhead

VIII. CONCLUSIONS

This paper proposed a new strategy, CBC, for caching in MANET by providing efficient data caching mechanisms in the in the clusters and local zones (transmission rang). The proposed caching strategy is scalable and affords lower overhead even with the increase of the number of nodes. In this strategy, the network is divided virtually into clusters and zones, where the each zone covers a cluster and an extended area out of it up to the transmission range, which makes it possible for the clients in a cluster to share their data with other clients in the transmission range, which helps relieve the longer average query latency and limited data accessibility problems in MANET. In the proposed CBC caching strategy, a cache discovery process and a cache management technique are used, where the cache management includes cache admission control, *LIP* replacement policy and cache consistency technique. The proposed strategy was simulated using NS2 to evaluate its performance, where the results shows that CBC caching strategy performs better than other caching strategies .We aim to extend our work of cooperative cache based data access to enhance QoS of MANET.

REFERENCES

[1] H. M. Abu-Thuraia and I. I. Abuhaibasecure ,"Zone Routing Protocol in Ad-Hoc _networks " Islamic University – Gaza, Faculty of Engineering Computer Engineering Department.pp.3,2010.
[2] H. LU "REPLICA ALLOCATION AND DATA UPDATE STRATEGIES IN MOBILE AD HOC NETWORKS", A Thesis in master,The Faculty of Graduate Studies of the University of Guelph April ,2005
[3] Y. L. Zhong and C.G. Gong, “Supporting Cooperative Caching in Ad Hoc Networks”, IEEE Infocom, 2004, pp. 2537-2547
[4] N. Chand, R.C. Joshi and M. Misra, Cooperative Caching in Mobile Ad Hoc Networks Based on Data Utility, International Journal of Mobile Information Systems Vol. 3, No. 1, pp.19-37,2007.
[5] N. H. WIN "Tightly Cooperative Caching Approach in Mobile Ad Hoc Network", WSEAS TRANSACTIONS ON COMPUTERS, Volume 13, 2014.
[6] H. R. K. Nejad " Resouce- Aware Cooperative Caching on Mobile Ad - hoc peer to peer Networks " Ryerson University, May 2012.
[7] C. Uikey " Node Based Cluster Routing Algorithm for Mobile Ad-Hoc Network" International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 7, July 2011.

[8] D. Gavalas, G. Pantziou, C. Konstantopoulos, Basilis Mamalis. “Clustering of Mobile Ad Hoc Networks: An Adaptive Broadcast Period Approach”, IEEE ICC, 2006, pp. 4034-4039.
[9] T.J.Kwon, M.Gerla, V.K.Varma, T.Russel Hsing.”Efficient Flooding with Passive clustering-An Overhead-Free Selective Forward Mechanism for Ad Hoc/Sensor Networks”, roceedings of the IEEE, Vol.91, No.8, pp.1210-1220, 2003.
[10] N. CHAND, R.C. JOSHI and M. MISRA “Efficient Cooperative Caching in Ad Hoc Networks Communication System Software and Middleware," 2006. Comsware 2006. First International Conference on 08-12 Jan. 2006.
[11] J. Cho, S. Oh, J. Kim, J. Lee, "Neighbor caching in multi-hop wireless ad hoc networks," IEEE Communications Letters, Volume 7, Issue 11, Nov. 2003 Page(s):525 – 527.
[12] Y. Ting ,Y. Chang. A novel cooperative caching scheme for wireless ad hoc networks: Groupcaching. In NAS ’07: Proceedings of the International Conference on Networking, Architecture,and Storage, 2007.
[13] J. Tian, "Cross-Layer Design for Cooperative Caching in Mobile Ad Hoc Networks", Proc .of IEEE Consumer Communications and Networking Conf, 2008.
[14] N. CHAND "Global cluster cooperation strategy in mobile ad hoc network " International Journal on Computer Science and Engineering, vol. 2, no. 7, pp. 2268-2273, 2010.
[15] N. Chand and N. Chauhan, “Cooperative Caching in Mobile Ad Hoc Networks Through Clustering”, ACM Recent Researches in Software Engineering, Parallel and Distributed Systems, 2011.
[16] N. Chand, R.C. Joshi and M. Misra, “Cooperative Caching Strategy in Mobile Ad Hoc Networks Based on Clusters,” International Journal of Wireless Personal Communication, Vol. 43, Issue 1, pp. 41- 63, Oct 2007.
[17] Kuppusamyp "Certain improvements in cooperative caching for mant using evolutionary algorithm " Faculty of information and communication engineering Anna university chennal ,600 025 APRIL 2014.
[18] T.J.Kwon, M.Gerla, V.K.Varma, T.Russel Hsing.”Efficient Flooding with Passive clustering-An Overhead-Free Selective Forward Mechanism for Ad Hoc/Sensor Networks”, roceedings of the IEEE, Vol.91, No.8, 2003, pp.1210-1220.