ELSEVIER

Energy Efficient Cache Discovery in Wireless Mobile Ad Hoc Networks

Preetha Theresa[1,*] and K. Poulose Jacob[2]

Department of Computer Science, Cochin University of Science and Technology, Kochi, Kerala, India.
e-mail: [1]preetha@mec.ac.in

Abstract. Data caching is an attractive solution for reducing bandwidth demands and network latency in mobile ad hoc networks. Deploying caches in mobile nodes can reduce the overall traffic considerably. Cache hits eliminate the need to contact the data source frequently, which avoids additional network overhead. In this paper we propose a data discovery and cache management policy for cooperative caching, which reduces the power usage, caching overhead and delay by reducing the number of control messages flooded in to the network. A cache discovery process based on position coordinates of neighboring nodes is developed for this. The simulation results gives a promising result based on the metrics of studies.

Keywords: cache discovery, cache placement, cache replacement, cooperative caching, data dissemination.

1. Introduction

As the communication devices become more intelligent and detached from wired networks, mobile networks have emerged as an important component of networking. Among the mobile networks, ad hoc networks have a prominent place. Ad hoc networks are infrastructure less self organizing, peer to peer, and rapidly deployable networks with wireless nodes. The mobile nodes cooperate with each other to dynamically establish communication using limited network management and administration. Ad hoc networks have multiple applications in the areas where wired infrastructure may be unavailable such as battle fields and rescue areas. In these types of networks new hosts can appear and old ones can disappear at any time. The topology of this network is very fragile, it can change at any moment and disconnections are frequent due to mobility or activity status changes. Battery is the major source of energy for these devices and the development of battery technology has not been able to match the power requirements of increasing resource demand. Since battery capacity is fixed, a wireless mobile node is extremely energy constrained. Hence all network related transactions should be power aware to be able to make efficient use of the overall energy resources of the network. In a data-management point of view, these restrictions introduce several issues that

*Corresponding author

K. R. Venugopal and L. M. Patnaik (Eds.) ICCN 2013, pp. 233–242.

need to be addressed. Data transfers must be reduced and mechanisms must be deployed to confront the frequent disconnections and low bandwidth constraints. Therefore it is a challenging task to present the data efficiently by reducing the delay or waiting time to the end user.

Data caching is recognized as feasible approach to improve the performance in many traditional systems. Caching is the process of pre-fetching the needed data and storing it closer to the source. In mobile ad hoc networks, caching can lead to significant bandwidth savings, perceived network latency reduction and ensures higher data availability. Data caching in ad hoc networks are mainly proposed as cooperative caching [6]. Cooperative caching aims to reduce the redundant data transfer using a mechanism which enables the local cache of different mobile clients to be shared in a cooperative manner. In cooperative caching the mobile clients are configured to request the data object from its local set of data items, if not found queries its neighboring nodes. When there is a cache miss in the neighboring nodes queried, the data item is retrieved directly from the server and this procedure continues recursively.

In mobile ad hoc networks, caching can improve mobile client perception in three ways. First, retrieving data from a remote data center involves wireless media network transfers and there is a chance of data loss due to the wireless link characteristics. Second, when the data is served locally, the network latency is hidden from the user. The data center processes the data request only when there is a local and cooperative cache miss. By doing this the server load is balanced, which consequently reduces the latency in serving client request. Thirdly, frequent disconnections which occur in ad hoc networks can be hidden from users, making the network more reliable.

Cooperative caches have different functions: Cache discovery, placement and replacement, consistency maintenance and data dissemination. Discovery refers to how a mobile node locates the cached data. Placement is the processes of selecting appropriate nodes as cache locations and replacement is the strategy used for evicting data when the cache is full. Consistency maintenance is maintaining consistency among the source data and cached copies. Towards the goal of improving the performance in cooperative cache, we propose a cache discovery scheme in which each node is able to dynamically adjust its transmission range to reach its neighbouring nodes, thus saving power whenever possible. Moreover, due to simplicity, this approach does not have any additional overhead in implementation.

1.1 *Motivation and contribution*

Two major techniques used for information discovery in cooperative cache include broadcasting and cluster based approach. Although first method is the simpler version, it relays on flooding to broadcast the data request. Flooding increases network contention and overhead when the network density is high. In contrast, cluster based approach has been best owed with the features of reduced overhead by having a coordinator node which manages the discovery process. The neighbouring node which possesses the data can be easily found out by checking the lookup table maintained by the cluster head. The disadvantage of this approach is that group maintenance is difficult due to the mobility of nodes. The control node may get disconnected which causes excessive overhead [4]. The number of entries in the look up table increases when the network density is high. To maintain the correct status of the network, these tables must be frequently updated. This involves information exchange between the nodes which in turn increases the traffic overload in a dense network.

To circumvent these drawbacks we designed an energy efficient cache discovery protocol that minimizes the energy consumption for cache discovery and maximizes the life time of nodes.

The remainder of the paper is structured as follows. Section 2. reviews the related works in cache resolution, section 3. presents the system architecture of the proposed cooperative caching scheme, section 4. describes simulation model and metrics, section 5. gives the experimental results and section 6. concludes the paper.

2. Previous Work

Caching data in mobile nodes is an effective technique to improve performance in a mobile environment. Recently, several schemes for cooperative caching in mobile ad hoc networks have been presented in the literature. The algorithms proposed in [7,8,15], focuses more on cache placement and discovery while[6,9], concentrates more on cache management protocols. Cache management includes cache admission and replacement. The work [2,4,5,11,10] considers both aspects in cooperative caching. Below, we describe some representative cache discovery for cooperative caching.

S. Lim et al., [5] proposed an aggregate caching scheme to increase the data accessibility in Internet based mobile network. They used a broadcast based information search algorithm called simple search to locate the required data item. Whenever a mobile node needs data, the request is broadcasted to its adjacent nodes. Upon receiving the broadcast request, the adjacent nodes replies to the request if it has already cached the data otherwise the request is forwarded to its neighbors until it is acknowledged by an access point or some other nodes which have the requested data. Flooding is the technique used for broadcasting. This algorithm sets a hop limit for the request packet to reduce the traffic in the network. A caching technique which uses cluster based approach can be seen in [9], in which a coordinator node maintains the cluster cache state information of different nodes within its cluster domain. If there is a local cache miss, the coordinator node will find whether the data item is cached in other clients within its home cluster. Another approach for data discovery other than the mentioned schemes can be seen in [4] and [10]. In [4] a distributed group based approach is proposed, in which each node maintains two tables, a group table and self table. Whenever a data request occurs in a node, it first checks in its self table, if data is not found it will search in the group table which stores the data id and the node which posses the data. Here the mobile nodes interchange information about each cache event occurring in the network.

In [10], a cache discovery technique based on adaptive flooding broadcast is used for searching data in the network. According to this scheme a mobile node uses three schemes; adaptive flooding, profile-based resolution and road side resolution. In adaptive flooding, a node uses constrained flooding to search for items within the neighbourhood. In profile-based resolution, a node uses the past history of received requests. In road side resolution, forwarding nodes caching the requested item, reply to the requests instead of forwarding them to the remote data source.

3. System Architecture

3.1 *Network model*

A mobile ad hoc network is abstracted as a graph $G(V, E)$, where V is the set of nodes and $E \subseteq V^2$ is the set of links which gives the available communication. An edge (u, v) belongs to E means that

there is direct communication between two nodes u and v. The elements of E depend on the position and the communication range of nodes. All links in the graph are bidirectional i.e., if u is in the transmission range of v, v is also in the transmission range of u. The maximum communication range is assumed to be same for all nodes and is represented as R, which is given by the Euclidean distance $d(u, v)$ between nodes u and v. The set of neighborhood nodes in the range R are represented as $N_R(U)$ and the set of neighborhood nodes in the range R_i, is given as $N_{R_i}(U)$.

3.2 *System model*

We assume a mobile ad hoc network with a set of nodes which are able to communicate with each other. The transmission radius R determines the maximum communication range of each node and is equal for all nodes in the network. Two nodes in the network are neighbors if the Euclidean distance between their coordinates in the network is at most R. The Euclidean distance between the nodes are estimated based on the relative position of nodes. We assume that each node knows its current location precisely with the availability of Global Positioning System (GPS).

Initially, to find the neighbor node set in the transmission range R for node $AN_R(A)$, a short neighbor request control message is disseminated in to the network. The request control message contains the following fields: the *source id*, *current location* and a *request id*. The *request id* is used to identify the neighbor request control message. When a node receives the request control message, it sends back a reply control message which includes the *node id* and *current location* coordinates. Upon receiving the location coordinates of neighboring nodes, the source node measures the distance D_{ij} between the source node n_i and neighbor n_j using the formula,

$$D_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

When a node is farther away from the source the Euclidean distance will be large. Each node will arrange the neighboring node list in the descending order of their distance to determine the order of neighboring nodes to receive the data request. To reduce the number of messages, the neighboring node set is divided into different zones, based on the transmission range to maintain the neighbor node set accurately, each node periodically sends a request control message to its neighbors.

Each node maintains a list which stores the cached data item. The list contains the following fields: cached data id, cached data item, ttl, time difference between the current access and previous access. This table is updated when a new data is placed in to the cache. The cache space for each node is limited and when it is full, a replacement strategy evicts the unwanted data. The contents of the local cache are shared by its neighboring nodes.

The data center is assumed to be a fixed location. The data center maintains a set of data items uniquely identified by means of data item id D_i for $1 \leq i \leq n$ where 'n' is the size of the data base. The size of each data item varies from $S_{\min}$ to $S_{\max}$. Each node has a local cache, with certain data items. Each mobile nodes is identified by a distinct ⟨Host id, Name⟩ for $1 \leq i \leq N$, where N is the density of the network. Nodes in the network retrieve data items either from the local cache or from the neighboring cache if there is a local miss. When a node fails to find data in neighboring nodes, data is retrieved from the data center. When a node receives a fresh data directly from the server, it caches a copy of it in the local cache and becomes a provider for that cached content for the neighboring nodes.

When a node wants to access data, it checks in its own local cache. If the requested data is not cached, the node checks whether the data is present in the neighboring nodes. If we are not able to find the data from the neighbor list the request is given to the data center.

3.3 *Cache Discovery*

The cache discovery protocol we propose is based on minimizing the power or energy per bit required to transmit a packet from source to destination. The goal of this scheme is to reduce the average number of messages among the cooperative caches while maintaining high cache hit ratio. The link coast for this transmission can be defined for two cases, (a) when the transmit power is fixed and (b) when the transmit power is varied dynamically as a function of the distance between transmitter and the intended receiver. For the first case, the power needed to transmit and receive a message depends on the message size. For the latter case, the power consumed $P(d)$ by a node in transmitting a request for a distance d is given by $P(d) = d^{\alpha} + c$ for some constants α and c. If we ignore the constant we can see that the power consumption is directly proportional to distance. If the nodes can adjust their transmission power for each node based on distance, the power consumption can be reduced considerably, which leads to increased battery life. So by making use of the position coordinates we can transmit packets with minimum required transmit energy. The basic requirement of this scheme is that each node should know its relative position.

In the proposed algorithm, the decision to forward the data request is based solely on the location of itself and its neighboring nodes. We divide the maximum transmission range of a node in to different zones with transmission radius $R/2$, $R/4$ and $R/6$ and find the nodes present in this transmission radius excluding the one already present in the lower range. This can be found from the neighbor node list formed when a node is active.

When the requested data is not found in the local cache the request is forwarded to the nodes in the lowest transmission radius zone. After sending the request the node waits for the reply. If the node does not receive a positive reply after a period of time t_1, which is a predefined threshold, the node searches the data in the next zone and this process continues until it gets the needed data or when it reaches the maximum transmission range. If we are not able to find the required data within the transmission radius R the request is directly given to the server. The time out interval set for each zone is different to minimize the waiting time. The power needed for each transmission is assumed to be different. The process of cache discovery is fully distributed and runs in all the nodes in the network.

3.4 *Cache placement and replacement*

When there is a local miss the data item is fetched either from the neighboring nodes or from the server. The cache placement module is triggered when the data item is brought in, to decide whether to cache or not the incoming data. In order to cache more distinct data the caching decision is done based on two parameters, size and distance. We set a threshold value T, which is 50% of the cache size for a data item to be admitted to cache. The data coming from the neighboring nodes are also not cached in order to increase the data accessibility.

In cooperative caching if data replacement decision is made by individual nodes by considering only their local cache, the performance is degraded because the data may be present in the neighboring nodes. In order to cache more distinct data, new data item fetched from adjacent nodes are not

cached. When the cache is full, appropriate data from the cache have to be evicted to make room for the incoming data. The replacement policy proposed here considers the number of references for a particular data item and gives more emphasis data items that are referenced more than once. If we have data items referenced only once then that set is given priority for replacement. For this LRU policy is used. If an item is referenced more than once the inter arrival time between the recent two references is considered for eviction.

Let t_c be the current reference time and t_r be the previous reference time then T_{int}, the inter arrival time is given by (1)

$$T_{\text{int}} = t_c - t_r \tag{1}$$

If $t_c - t_r = 0$, an item whose last reference time is smaller is replaced.

If $T_{\text{int}} > 0$, then the replacement decision is made on the value of $K(i)$ which is given as (2)

$$K(i) = \max \sum_{i=1}^{n} t_c - t_r \tag{2}$$

The data items with maximum inter arrival time is considered for replacement. In both cases if more than one data item have the same value, the TTL parameter is taken and the one with lower TTL value is removed as the data with lower TTL will be outdated soon.

4. Simulation Study

We have developed a simulation model in JAVA. The proposed scheme is a fully distributed scheme, with each node runs an application to request, retrieve and cache data items from other neighboring nodes. Each node caches part of the requested items temporarily. The simulated mobile environment consists of a number of mobile nodes which are randomly placed in an area of $1000 \times 1000\,\text{m}^2$. Each node is identified by a node id and a host name. The position of each node is given by the x and y coordinates. The data centre is implemented in a fixed position in the simulation area. The data center contains all the data items requested by the mobile nodes. The size of each data item is uniformly distributed between $s_{\min}$ and $s_{\max}$. The database in the data center contains 1000 data items, with each item identified using a data id. The nodes that generate data request are selected randomly and uniformly. The time interval between two consecutive queries generated from each node follows an exponential distribution with mean of 10 sec. Each mobile node generates a single stream of read only queries. The queries generated follows a Zipf distribution [14], which is frequently used to model non uniform distribution. In Zipf distribution, an item ranked i ($1 \leq i \leq N$) is accessed with a probability,

$$P_N(i) = \frac{1}{i^{\theta} \sum_{k=1}^{N} \frac{1}{k^{\theta}}}, \quad 0 < \theta \leq 1$$

The value of θ ranges between zero for uniform distribution and one for strict zipf distribution. In our study θ is set to 0.8. The data request is processed in FCFS manner at the server. An infinite queue is used to buffer the request when the data center is busy. Each miss in the cooperate cache will incur a delay of 8 ms to retrieve data from the data center.

Initially, the mobile nodes are randomly distributed in the simulation area. After that each node randomly chooses its destination with a speed s which is uniformly distributed $U(V_{\min}, V_{\max})$ and

travels with that constant speed s. When the node reaches destination, it pause for 200 seconds. After that it moves to the new destination with speed s'. For the time out mechanism each node maintains two time out period for the primary and secondary zone. The details of the simulation pattern is given in table 1.

4.1 *Simulation parameters*

Table 1. Simulation Parameters.

Parameter	Value
Simulation Time	3600 sec
Simulation Area	$1000 \times 1000\,m^2$
Database	1000 items
Cache size	20–70 % of total database size
S_{min}	1
S_{max}	10
Nodes Density	20–70
Mobility model	Random waypoint
Transmission Range	500 m
Speed of the mobile host	1–10
Pause time	200 sec
Mean query generation time	10 s

4.2 *Metrices*

The performance metrics evaluated includes cache hit ratio, message overhead and power savings ratio. The evaluation of these parameters are done by varying the number of cache locations with respect to number of nodes and the behavior of cache hit ratio for different cache sizes. The hit ratio is defined as the percentage of requests that can be served from previously cached data. Since the replacement algorithm decides whether to cache the data or not, it affects the cache hits of future requests. The percentage of power saved is calculated as power usage of cooperative caching scheme compared—power usage of the proposed scheme $/$ power usage of the proposed scheme. Message overhead is the overhead messages needed to manage the cache discovery process in cooperative cache.

5. Experimental Results

To evaluate the performance of the proposed cache discovery scheme, we compared the performance of our new cooperative caching scheme (CCN), with Neighbor Caching (NC), a caching scheme which uses broadcasting and LRU for cache replacement. Figure 1 shows the performance comparison of two schemes, as a function message overhead for different node densities.

The figure shows that CCN outperforms NC at all node densities. As the node density increases, the difference become more significant, this implies that CCN can benefit from larger node densities. Figure 2 depicts the power savings ratio of the proposed cache discovery scheme compared to neighbor caching. From Figure 3 we can see that the cache hit ratio for CCN is greater than NC for different cache sizes. The relative performance of cache hit ratio remains relatively stable for higher cache sizes.

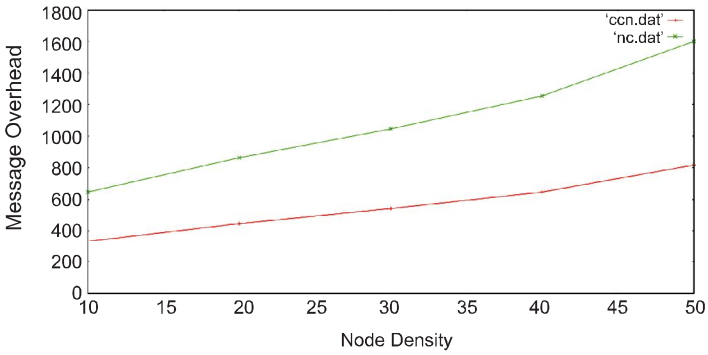

Figure 1. Message overhead for different node densities.

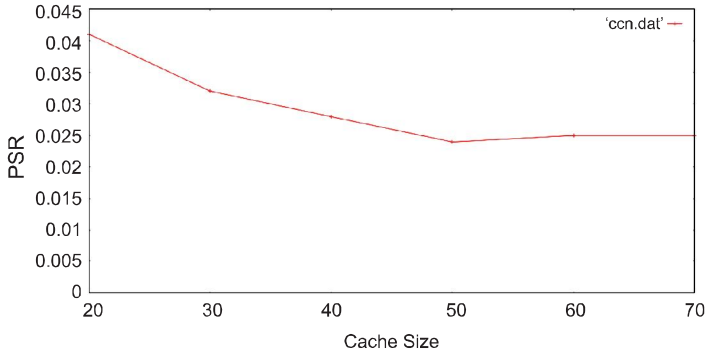

Figure 2. Power savings ratio for different cache sizes.

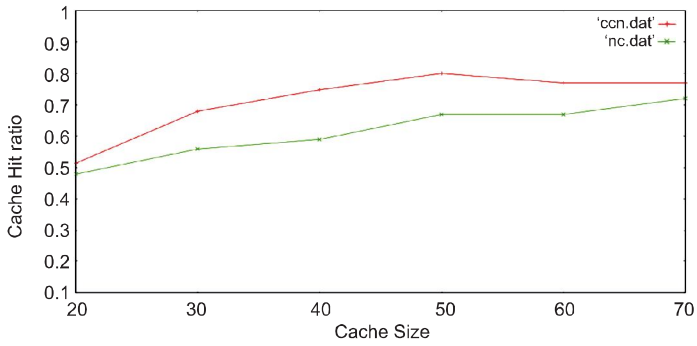

Figure 3. Cache hit ratio for different cache sizes.

6. Conclusion

In this paper we addressed the problem of data discovery and cache management policies for cooperative caching in ad hoc networks. The objective of our problem was to reduce the power consumption for data discovery by minimizing the number of messages flooded in to the network. We designed a data discovery process based on the position of the neighboring node. Experimental results show that the proposed approach can significantly improve the performance in terms of message over head, cache hit ratio and power savings.

References

[1] Zhao, Zhang, P., Cao, G. and Das, C. R.: Cooperative Caching in Wireless P2P Networks: Design, Implementation and Evaluation. In *IEEE Trans Parallel Distributed Syst.*, 21(2), 229 (2010).
[2] Chand, N., Joshi, R. C. and Misra, M.: Cooperative Caching Strategy in Mobile Ad Hoc Networks Based on Clusters. In *Wireless Personal Communications*, 43, 41–63, (2007).
[3] Joonho Cho, Seungtaek Oh, Jaemyoung Kim, Hyeong Ho Lee and Joonwon Lee: Neighbor Caching in Multi-hop Wireless Ad Hoc Networks. In *IEEE Communications Letters*, 7(11), 525–527 (2003).
[4] Yi-Wei Ting and Yeim-Kuan 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).
[5] Lim, S., Lee, W. C., Cao, G. and Das, C. R.: A Novel Caching Scheme for Improving Internet-Based Mobile Ad Hoc Networks Performance. In *Ad Hoc Netw.*, 4(2), 225 (2006).
[6] Yin, L. and Cao, G.: Supporting Cooperative Caching in Ad Hoc Networks. In *IEEE Trans Mobile Comput.* 5(1), 77 (2006).
[7] Fiore, M., Mininni, F., Casetti, C. and Chiasserini, D. F.: To Cache or Not to Cache? In *Proceedings of the IEEE Conference on Computer and Communications (INFOCOM 2009)*, Rio de Janeiro, Brazil, 235 (2009).
[8] Tang, B., Gupta, H. and Das S. R.: Benefit-Based Data Caching in Ad Hoc Networks. In *IEEE Trans Mobile Comput.*, 7(3), 289 (2008).
[9] Denko, M. K. and Tian, J.: Cross-Layer Design for Cooperative Caching in Mobile Ad Hoc Networks. In *Proc. of IEEE Consumer Communications and Networking Conf.* (2008).
[10] Du, Y. and Gupta, S. K. S.: COOP-A Cooperative Caching Service in MANETs. In *Proc. Joint Int. Conf Autonomic and Autonomous Systems and Int. Conf. Networking and Services*, 58–63 (2005).
[11] Dan Hirsch and Sanjay Madria: A Resource-Efficient Adaptive Caching Scheme for Mobile Ad-Hoc Networks. In *29th IEEE International Symposium on Reliable Distributed Systems* (2010).
[12] González-Cañete, et al.: A Cross Layer Interception and Redirection Cooperative Caching Scheme for MANETs. In *EURASIP Journal on Wireless Communications and Networking* (2012).
[13] Niels Sluijs, Frédéric Iterbeke, Tim Wauters, Filip De Turck, Bart Dhoedt and Piet Demeester: Cooperative Caching versus Proactive Replication for Location Dependent Request Patterns. In *Journal of Network and Computer Applications*, 34(2), 562–574 (2011).
[14] Tiance Wang, Pan Hui, Sanjeev R. Kulkarni and Paul Cuff: Cooperative Caching based on File Popularity Ranking in Delay Tolerant Networks. In *Proc. of Extreme Com.*, Zürich, Switzerland (2012).
[15] Breslau, L., Cao, P., Fan, L., Phillips, G. and Sheker, S.: Web Caching and Zipf-Like Distributions: Evidence and Implications. In *Proceedings of IEEE INFOCOM*, 126–134, March (1999).
[16] Hassan Artail, et al.: COACS: A Cooperative and Adaptive Caching System for MANETs. In *IEEE Transactions on Mobile Computing*, 7(8), August (2008).
[17] Chi-Yin Chow, Hong Va Leong and Alvin Chan: Peer-to-Peer Cooperative Caching in Mobile Environments. In *Proc. of 24th International Conference on Distributed Computing Systems Workshops (ICDCSW'04)* (2004).
[18] Preetha Theresa Joy and Polouse Jacob, K.: Cooperative Caching Techniques for Mobile Ad Hoc Networks. In *Proc. of the Intl. Conference on Data Science & Engineering (ICDSE)*, Kochi, India, 175–180 (2012).

[19] Jin, S. and Bestavros, A.: Greedy Dual Web Caching Algorithm: Exploiting the Two Sources of Temporal Locality In Web Request Streams. In *Computer Communications*, 24(2), 174–83, February (2001).
[20] Joonho Cho, Seungtaek Oh, Jaemyoung Kim, Hyeong Ho Lee and Joonwon Lee: Neighbor Caching in Multi-hop Wireless Ad Hoc Networks. In *IEEE Communications Letters*, 7(11), 525–527 (2003).