International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:7, No:5, 2013

Energy Efficient Cooperative Caching in WSN

Narottam Chand

***Abstract*—**Wireless sensor networks (WSNs) consist of number of tiny, low cost and low power sensor nodes to monitor some physical phenomenon. The major limitation in these networks is the use of non-rechargeable battery having limited power supply. The main cause of energy consumption in such networks is communication subsystem. This paper presents an energy efficient Cluster Cooperative Caching at Sensor (C3S) based upon grid type clustering. Sensor nodes belonging to the same cluster/grid form a cooperative cache system for the node since the cost for communication with them is low both in terms of energy consumption and message exchanges. The proposed scheme uses cache admission control and utility based data replacement policy to ensure that more useful data is retained in the local cache of a node. Simulation results demonstrate that C3S scheme performs better in various performance metrics than NICoCa which is existing cooperative caching protocol for WSNs.

***Keywords*—**Cooperative caching, cache replacement, admission control, WSN, clustering.

I. Introduction

CACHING is a potential technique where frequently accessed data items are generally stored closer to the requester than to the original source of the information. This technique is employed in traditional areas such as operating systems, virtual memory, distributed systems, and Web environments to enhance the system performance by improving the data availability and query response time. Queries can be answered faster if the requested information is in the cache rather than sending the request to the original source, which may be remotely located. Cooperative caching is a technique where a group of caches at different nodes work in coordination to achieve better performance.

Due to deployment of wireless sensors in unattended harsh environment, it is not possible to charge or replace their batteries. Therefore, energy efficient operation of wireless sensors to prolong the lifetime of overall WSN is of utmost importance. Energy consumption mainly occurs due to three types of operations (i) sensing, (ii) data processing, and (iii) data communication. Various applications in WSNs demand reduction in the number of communications among the sensors to serve the requested data with lower latency and minimum energy consumption. Network lifetime of WSN can be enhanced if the rate of nodes' energy depletion is reduced, which is possible if the amount of communication is reduced. This can be achieved by caching useful data for each sensor either in its local flash memory or in the nearby neighborhood.

A lot of research in data routing [1], [2], data compression [3], and in-network aggregation [4] has been carried out in WSNs during recent years. This paper targets the problem of efficient data dissemination and tries to solve it by utilizing the memory of sensor nodes by caching the data items in it. Caching if implemented optimally can reduce network traffic and enhance data availability to the users through sink.

In this paper we have proposed a technique called Cluster Cooperative Caching at Sensor (C3S) in wireless sensor network. The sensor field is divided into equal size clusters and each cluster is monitored and controlled by a node called cluster head (CH). These CHs can communicate directly with the sink or base station (BS) in a single hop or through other CHs to form multi hop routing of data to the sink/BS. The sensor nodes (SN) within a cluster sense the environment and forward the sensed data to their CH. The CH first aggregates the data received from all the sensor nodes within the cluster and then finally forwards it directly or through other CH to the sink. Since the CH has to collect the data, perform data aggregation and has to involve in long distance transmission as compared to normal sensor nodes, therefore its energy depletes at faster rate than the other nodes within the cluster. The role of the CH, therefore, is changed whenever its energy falls below threshold value and can be assigned to other node within a cluster which is more powerful.

Rest of the paper is organized as follows. Section II describes the related work. System model has been explained in Section III. Section IV describes proposed C3S scheme. Section V defines various simulation parameters, performance metrics and explains simulation results. Section VI gives concluding remarks of the paper.

II. Related Work

Various researches have been carried out by exploiting data caching either in some intermediate nodes or at a location nearer to the sink in the wireless sensor networks. Jinbao Li et al. [6] proposes a caching scheme for the multi-sink sensor network. The sensor network forms a network tree for particular sink. A common subtree is formed out of such trees and the root of the common subtree is selected as the data caching node to reduce the communication cost.

Md. A. Rahman et al. [7] propose effective caching by data negotiation between base station and the sensors, developing expectancy of data change and data vanishing. J. Xu et al. [5] proposed a waiting cache scheme which waits for the data of same cluster until it becomes available within a threshold, aggregating it with the packet from the lower cluster and then sending it to the sink, thus reducing number of packets travelling in the network. K.S. Prabh et al. [8] consider the whole network to be a Steiner Data Caching Tree which actually is a binary tree and buffers data at some intermediate

Narottam Chand is Associate Professor at the Department of Computer Science & Engineering, National Institute of Technology, Hamirpur, 177 005 India (e-mail: nar.chand@gmail.com).

node (data cache) such that it reduces the network traffic during multicast. In [9], M.N. Al-Ameen et al. exploit caching for faulty nodes in WSNs and propose a mechanism to handle the packets when node fails. T.P. Sharma et al. [4] proposed a cooperative caching scheme which exploits cooperation among various sensor nodes in a defined region. Apart from its own local storage, a node utilizes memory of nodes from certain region around it to form larger cache storage known as cumulative cache. A token based cache admission control scheme is devised where node holding the token can cache or replace data item. Disadvantage of proposed model is that, there are overheads to maintain and rotate the token. N. Dimokas et al. [10], [11] have identified various goals which are required to be optimized such as energy consumption, access latency, and number of copies of data items to be placed at different locations. Disadvantage of schemes is that node importance (NI) considers neighborhood of a particular node. So, overhead to find NI for all the nodes consumes energy which in turn reduces the lifetime of sensor network.

III. System Model

We assume a wireless sensor network (WSN) consisting of sensor nodes (SNs) that interact with the environment and sense the physical data. A SN that senses and holds the original copy of a data item is called source for that particular data item. A data request initiated by a sink is forwarded hop-by-hop along the routing path until it reaches the source and then the source sends back the requested data. Sensor nodes frequently access the data, and cache some data locally to reduce network traffic and data access delay. As sensor nodes do not have sufficient cache storage e.g. for multimedia data, cooperative caching may be more useful where cached data at sensor node may also be shared by the neighboring nodes.

WSN comprises a group of sensor nodes communicating through omni-directional antennas with the same transmission range. The WSN topology is thus represented by an undirected graph G = (V, E), where V is the set of sensor nodes SN_1, SN_2, ..., and $E \subseteq V \times V$ is the set of links between nodes. The existence of a link $(SN_i, SN_j) \in E$ also means $(SN_j, SN_i) \in E$, and that nodes SN_i and SN_j are within the transmission range of each other, and are called one-hop neighbors of each other. The set of one hop neighbors of a node SN_i belonging to the same cluster/grid is denoted by SN_i^1 and forms a cooperative region. The combination of nodes and transitive closure of their cluster neighbors forms a wireless sensor network. The sensor nodes might be turned off/on at any time, so the set of live nodes varies with time. We make the following assumptions in this system environment:

- Sensor nodes are static, the communication links are bidirectional, and the nodes communicate using multi hop.
- The WSN is homogeneous i.e. the computation, communication and energy capabilities are the same for all sensor nodes.
- Each sensor node is aware of its geographical coordinates (x, y) through some localization method [12].
- Unique node identification is assigned to each sensor node in the system. The system has total of M nodes and SN_i ($1 \leq i \leq M$) is node identification.
- The set of data items is denoted by $D = \{d_1, d_2, \ldots\ldots d_N\}$, where N is the total number of data items and d_j ($1 \leq j \leq N$) is a data identifier. D_i denotes the actual data for item d_i.
- All the data items have same size.
- The original of each data item is at particular source.
- Each sensor node has a cache space of C bytes and can cache a number of data items depending upon the size of items.
- Data value sensed at a source may change with time. After a data item is updated, its cached copy maintained on one or more nodes may become invalid.

IV. C3S Cooperative Caching

This section describes our C3S caching scheme that uses the above described model. C3S exploits cooperation among various sensor nodes inside a cluster. The design rationale of C3S is that, for a sensor node, all other nodes within its cluster domain form a cooperative cache system for the sensor node since local caches of the nodes virtually form a cumulative cache. In each cluster, the cluster head (CH) is selected to act as the Cache Index Node (CIN), which is responsible for recording the information about cached items by all the nodes within its cluster. When a node in any cluster stores/deletes some data item into/from its cache, it sends the information to its CIN so that the corresponding index value can be updated. For each cached item its Time To Live (TTL) information is also maintained at the CIN. Whenever, cluster head is rotated, the responsibility of CIN is transferred to new CH.

In C3S, when a node experiences cache miss (called local cache miss), the node will look up the required data item from the cluster members by sending a request to the CIN. Only when the node cannot find the data item in the cluster members' caches (called cluster cache miss), it will request the data from the CIN that lies on the routing path towards source. If a cluster along the path to the source has the requested data (called remote cache hit), then it can serve the request without forwarding it further towards the source. Otherwise, the request will be satisfied by the source.

Fig. 1 shows the behavior of C3S caching strategy for a data request. For each request, one of the following four cases holds:

Local hit: when copy of the requested data item is stored in the cache of the requester. If the data item is valid, it is retrieved to serve the query and no cooperation is necessary.

Cluster hit: when the requested data item is stored by a node within the cluster of the requester. The requester sends a request to the CIN and the CIN returns the address of the node that has cached the data item.

Remote hit: when the data is found with a node belonging to a cluster (other than home cluster of the requester) along the routing path to the data source.

Global hit: data item is retrieved from the source.

A. Cache Discovery

C3S uses a cache discovery algorithm to find the node who has cached the requested data item. When a data request is initiated at a node, it first looks for the data item in its own cache. If there is a local cache miss, the node confirms with the CIN if the data item is cached in other nodes within its home cluster. In case of a cluster cache miss, the request is forwarded to the next hop node along the routing path. Before forwarding a request, each node along the path searches the item in its local/cluster cache. If the data item is not found on the clusters along the routing path, the request finally reaches the data source and the data source sends back the requested data.

Using this strategy, we describe the cache discovery steps to determine the data access path to the node having the requested cached data or to the data source. In Fig. 2, let us assume SN_i needs to access a data item d_x. It sends a request for data item d_x to SN_k located along the path through which the request travels to the data source SN_s, where $k \in \{a, b, c\}$.

The discovery process is described as follows:

1. When SN_i needs d_x, it first checks its own cache. If the data item is not available in its local cache, it sends a lookup packet to the CIN SN_a in its cluster. Upon receiving the lookup message, the CIN searches the index for the requested data item. If the item is found, the CIN replies with an ack packet containing id of the node who has cached the item. SN_i sends a confirm packet to the node whose id is returned by SN_a and the node responds with reply packet that contains the requested data item. If no node has cached the item, the request message is forwarded towards next CIN SN_b that lies along the routing path towards data source SN_s.
2. When SN_j ($j \in \{b, c\}$, in the order b then c) receives a request packet, it searches its CIN whether some node has cached the item d_x within the cluster. When SN_j locates an entry for d_x in its index, it sends confirm packet to the node within cluster or request packet to next hop node as described in Step 1 above.
3. When a node/SN_s receives a confirm packet, it sends the reply packet to the requester.

The reply packet containing item id d_x, actual data D_x and TTL_x, is forwarded hop-by-hop along the routing path until it reaches the original requester. Once a node receives the requested data, it triggers the cache admission control procedure to determine whether it should cache the data item.

B. Cache Admission Control

When a sensor node receives the requested data or a data item passes through it, a cache admission control is triggered to decide whether it should be stored into the cache of the node or not. Inserting a data item into cache might not always be favorable because incorrect decision can lower the probability of cache hits and also makes poor utilization of the limited storage. In C3S, the cache admission decision at a node SN_i is based on two distance parameters (i) number of hops H_i between SN_i and sensor/source of the data item from where the cached copy is shared, and (ii) number of hops H_s between SN_i and sink.

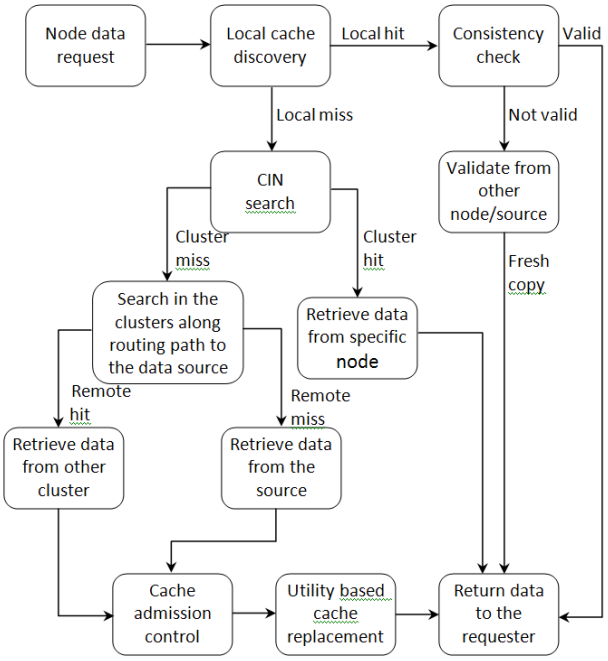

Fig. 1 Working of C3S cooperative caching strategy

If the sensor/source is less than Δ hops away from the requesting node SN_i i.e. $H_i < \Delta$, then it does not cache the data; otherwise it caches the data item. In general, same data items are cached at least Δ hops away. A tradeoff exists between query latency and data accessibility. With a small Δ, the number of replicas for each data item is high and access delay for this data item is low. On the other hand, with a larger Δ, each data item has a small number of replicas, and the access delay can be little longer. Advantage is that sensor nodes can cache more distinct data items and still serve requests when the data source is not accessible. In C3S, we have used $\Delta = 2$, i.e. if the cache/source of the data resides in the same cluster of the requesting node, then the item is not cached, because it is unnecessary to replicate data item in the same cluster since cached data can be used by closely located sensors. So in C3S, the same data item is replicated at least two hops away.

A node SN_i is allowed to cache a data item only if H_s is having a value lower than the specified threshold Ω i.e. $H_s < \Omega$. To increase proximity of the data items nearer to sink, it is better to start caching data items at the sink. Initially all the data items are cached at sink and the next level for cache storage will be a SN along the routing path towards source. Initially we assume $\Omega = 1$ and value is gradually increased with the decreasing free cache space on the sensor nodes located nearer to the sink.

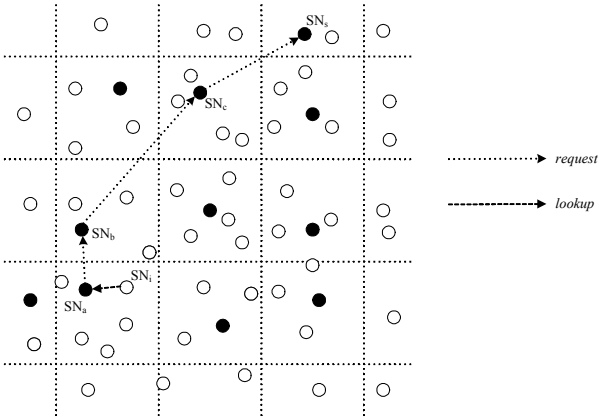

Fig. 2 Request packet from SN_i is forwarded to the data source SN_s

C. Cache Consistency

Cache consistency ensures that sensor nodes only access valid data and no stale data is used to serve the queries. Due to multi hop environment, limited bandwidth and energy constraints in wireless sensor networks, the weak consistency model is more attractive [13]. The C3S caching uses a simple weak consistency model based on Time To Live (TTL), in which a SN considers a cached copy up-to-date if its TTL has not expired. The node considers a data item as victim for replacement if its TTL expires. A SN refreshes a cached data item and updates its TTL if a fresh copy of the same data passes by.

D. Cache Replacement Policy

A cache replacement policy determines which data item should be deleted from the cache when the cache does not have enough free space to store a new item. Such policies apply a value function to each of the cached items, and select as victims, those items which satisfy some criteria. We have developed utility based cache replacement policy, where data item with the lowest utility is removed from the cache. Three factors are considered while computing utility value of a data item at a node:

1. Popularity

The access probability reflects the popularity of a data item for a node. An item with lower access probability should be chosen for replacement. At a node, the access probability P_i for data item d_i is given as

$$P_i = a_i / \sum_{k=1}^{N} a_k$$

where a_i is the mean access rate to data item d_i.

2. Distance

Distance (δ) is measured as the number of hops between the requesting node and the responding node (data source or cache). This policy incorporates the distance as an important parameter in selecting a victim for replacement. The greater the distance, the greater is the utility value of the data item. This is because caching data items which are further away, saves bandwidth and reduces latency for subsequent requests.

3. Consistency

A data item d_i is valid for a limited lifetime, which is known using the TTL_i field. An item which is valid for shorter period should be preferred for replacement.

Based on the above factors, the $utility_i$ for a data item d_i is computed using the following expression

$$utility_i = P_i.\delta_i.TTL_i$$

The idea is to maximize the total utility value for the data items kept in the cache. Therefore remove the cached data item d_i having minimum $utility_i$ value until the free cache space is sufficient to accommodate the incoming data.

V. Performance Evaluation

In this section, we evaluate the performance of proposed C3S protocol through simulating. We compare the performance of our protocol with NICoCa [10] which is cooperative caching protocol for wireless sensor networks.

A. Simulation Parameters

We consider a sensor field of size 100 × 100m^2 in which SNs are randomly deployed. All nodes are homogeneous. Various simulation parameters are listed in Table I.

TABLE I
Simulation Parameters

Parameter	Default Value	Range
Network size	(100×100)m^2	(50×50)~(400×400)m^2
Number of nodes	400	100~500
Transmission range (r)	100m	20~140m
Sink location	(0, 0)	
Initial Energy of node	2 Joule	
Data packet size	100 byte	
Mean query generate time (T_q)	5sec	2~100sec
Cache size (C)	800KB	200~1400KB
TTL	300sec	100~300sec
Skewness parameter (θ)	0.8	0~1
E_{elect}	50nJ/bit	
ε_{fs}	10pJ/bit/m^2	
ε_{amp}	0.00134pJ/bit/m^4	
Data aggregation (E_{DA})	5nJ/bit/signal	

B. Performance Metrics

1. Network Lifetime

Network lifetime of wireless sensor network is the time span from the deployment to the instant the network works and is able to achieve its objectives. During our simulation, we have used the following parameters to measure network lifetime:

- FND: number of rounds after which first node dies.
- HND: number of rounds after which 50% nodes die.

2. Average Query Latency (T_a)

The query latency is the time elapsed between the query is sent and the data is transmitted back to the sink, and average query latency (T_a) is the query latency averaged over all the

queries.

3. Byte Hit Ratio (B)

Byte hit ratio is defined as the ratio of the number of data bytes retrieved from the cache to the total number of requested data bytes. It is used as a measure of the efficiency of the cache management. Here byte hit ratio (B) includes local byte hit (B_{local}), cluster byte hit ($B_{cluster}$) and remote byte hit (B_{remote}).

C. Results

Here we study the effect of cache size on various performance metrics on our proposed protocol C3S and compare the results with existing cooperative caching protocol NICoCa [10] for WSN.

1. Effect of Cache Size on Network Lifetime

Figs. 3 and 4 show the results of network lifetime as a function of cache size of sensor node. To see the effect of cache size on the network lifetime, the number of nodes and the network size is kept fixed at its default values. Both the schemes exhibit better network lifetime with increasing cache size. This is because more required data items can be found in the local cache as the cache gets larger. Due to cooperation within a cluster, the remote byte hit ratio of C3S increases with increasing cache size because each node shares caches of its neighbors within the cluster. When the cache size is small, the contribution due to cluster hit and remote hit is more significant. Due to increase in local, cluster and remote byte hit ratio with increasing cache size, the overall byte hit ratio increases in the proposed protocol C3S. As byte hit ratio increases, more data may be shared from the nearby sensor nodes, thus reducing the number of transmissions and hence prolonging the network lifetime.

C3S always has better lifetime than NICoCa. This is due to the fact that to compute node importance (NI) in NICoCa, the message overhead is large and energy of a node depletes in exchanging these messages.

Also, due to cumulative caching within a cluster and better replacement policy, the C3S scheme outperforms NICoCa scheme under different cache size settings. We deploy utility based replacement in C3S which retains more useful data in the caches of nodes and thus query may be satisfied from local cache or nearby nodes.

2. Effect of Cache Size on Average Query Latency

Fig. 5 shows the effect of cache size on average query latency for proposed protocol C3S and the NICoCa [10] protocol. With increasing cache size, the average query latency decreases as more number of requests is satisfied from the cache. This is because more required data items can be found in the local cache as the cache gets larger. As cache size is small, the local byte hit ratio is low, and the requests need to travel large number of hops thus average query latency is quite high.

The proposed protocol C3S demonstrates lower query latency than NICoCa because of large size of cumulative cache due to cooperation within a cluster. Also, due to utility based replacement, the C3S scheme outperforms NICoCa scheme under different cache sizes.

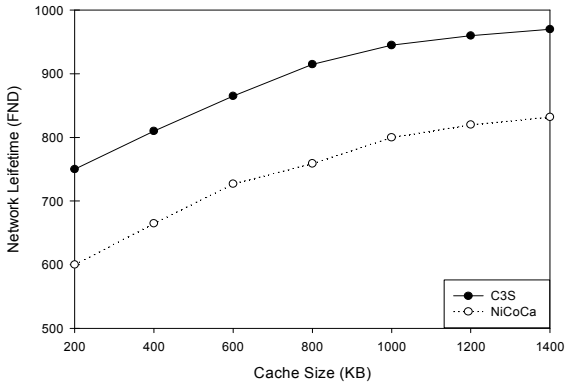

Fig. 3 Effect of cache size on FND

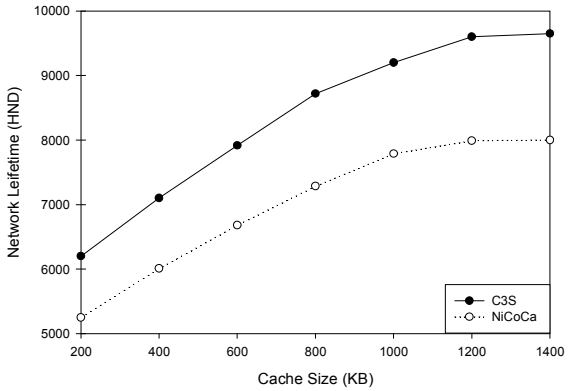

Fig. 4 Effect of cache size on HND

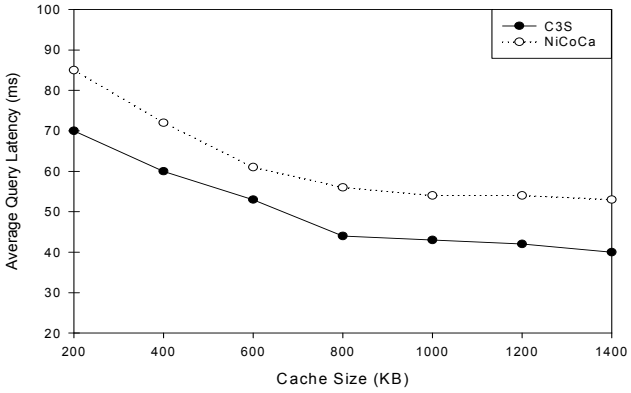

Fig. 5 Effect of cache size on average query latency

3. Effect of Cache Size on Byte Hit Ratio

Fig. 6 shows effect of cache size on byte hit ratio. Both the schemes exhibit better byte hit ratio with increasing cache size. This is because more required data items can be found in the local cache as the cache gets larger. Local byte hit ratio increases with the increasing cache size because with large

cache size more data can be shared locally. Due to cooperation within a cluster, the remote byte hit ratio of C3S increases with increasing cache size because each node shares caches of its neighbors within the cluster. When the cache size is small, the contribution due to cluster hit and remote hit is more significant. Due to increase in local, cluster and remote byte hit ratio with increasing cache size, the overall byte hit ratio increases in the proposed protocol C3S. Due to cumulative caching within a cluster and better replacement policy, the C3S scheme outperforms NICoCa scheme under different cache size settings. We deploy utility based replacement in C3S which retains more useful data in the caches of nodes and thus increasing the overall byte hit ratio. From Fig. 6, it is worth noting that C3S and NICoCa always reach their best performance when the cache size is 800 KB. This demonstrates their low cache space requirement. After the cache size increases beyond 800KB, sensor nodes in have enough size and byte hit ratio does not increase significantly.

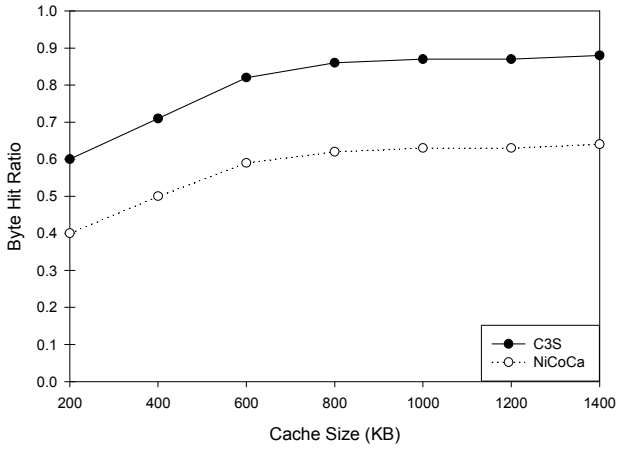

Fig. 6 Effect of cache size on byte hit ratio

VI. Conclusion

A cooperative caching scheme C3S for supporting efficient data dissemination and query processing in WSNs has been proposed. The scheme enables nodes to cooperatively share their data which helps alleviate the longer average query latency and limited data accessibility. As a part of cache management, cache discovery, cache admission control, utility based replacement policy and cache consistency techniques are developed. It has been observed that lifetime of WSN is enhanced through cooperative caching. The proposed cache discovery algorithm ensures that a requested data is returned from the nearest cache or source. The admission control prevents high data replication by enforcing a minimum distance between the same data item, while the replacement policy helps in improving the byte hit ratio and accessibility. Cache consistency ensures that nodes only access valid states of the data. Simulation results show that the C3S caching scheme performs better in terms of various performance metrics in comparison with NICoCa strategy.

References

[1] Tripathi, P. Gupta, Aditya Trivedi and Rahul Kala, "Wireless Sensor Node Placement Using Hybrid Genetic Programming and Genetic Algorithms," International Journal of Intelligent Information Technologies, Vol. 7, No. 2, pp. 63-83, 2011.

[2] Abbasi and M. Younis, "A Survey on Clustering Algorithms for Wireless Sensor Networks," ACM Journal of Computer Communications, Vol. 30, No. 14-15, pp. 2826-2841, 2007.

[3] N. Kimura and S. Latifi, "A Survey on Data Compression in Wireless Sensor Networks," International Conference on Information Technology: Coding and Computing, Vol. 2, pp. 8-13, 2005.

[4] T.P. Sharma, R.C. Joshi and M. Misra, "Dual Radio Based Cooperative Caching for Wireless Sensor Networks," IEEE International Conference on Networking, pp. 1-7, 2008.

[5] J. Xu, K. Li, Y. Shen and J. Liu, "An Energy-Efficient Waiting Caching Algorithm in Wireless Sensor Network," International Conference on Embedded and Ubiquitous Computing, Vol. 1, pp. 323-329, 2008.

[6] J. Li, S. Li and J. Zhu, "Data Caching Based Queries in Multi-Sink Sensor Networks," International Conference on Mobile Ad-hoc and Sensor Networks, pp. 9-16, 2009.

[7] Md. A. Rahman and S. Hussain, "Effective Caching in Wireless Sensor Network," International Conference on Advanced Information Networking and Applications Workshops, Vol. 1, pp. 43-47, 2007.

[8] K. Prabh and T. Abdelzaher, "Energy-Conserving Data Cache Placement in Sensor Networks," ACM Transactions on Sensor Networks, Vol. 1, No. 2, pp. 178–203, 2005.

[9] M.N. Al-Ameen and Md. R. Hasan, "The Mechanisms to Decide on Caching a Packet on Its Way of Transmission to a Faulty Node in Wireless Sensor Networks Based on the Analytical Models and Mathematical Evaluations," International Conference on Sensing Technology, pp. 336-341, 2008.

[10] N. Dimokas, D. Katsaros, L. Tassiulas and Y. Manolopoulos, "High Performance, Low Complexity Cooperative Caching for Wireless Sensor Networks," Springer International Journal of Wireless Networks, Vol. 17, No. 3, pp. 717-737, 2011.

[11] N. Dimokas, D. Katsaros and Y. Manolopoulos,"Cooperative Caching in Wireless Multimedia Sensor Networks," Springer Journal of Mobile Network Applications, pp. 337-356, 2008.

[12] Xiao, H. Chen and S. Zhou, "Distributed Localization Using a Moving Beacon in Wireless Sensor Networks", IEEE Transactions on Parallel and Distributed Systems, Vol. 19, No. 5, pp. 587-600, 2008.

[13] L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks," IEEE Transactions on Mobile Computing, Vol. 5, No. 1, pp. 77–89, 2006.

Dr. Narottam Chand received his Ph.D. degree from IIT Roorkee in Computer Science and Engineering. Previously he received M.Tech. and B.Tech. degrees in Computer Science and Engineering from IIT Delhi and NIT Hamirpur, respectively.

Presently he is working as Associate Professor, Department of Computer Science and Engineering, NIT Hamirpur. He has served as Head, Department of Computer Science & Engineering, during Feb 2008 to Jan 2011 and Head, Institute Computer Centre, during Feb 2008 to July 2009.

His current research areas of interest include mobile computing, mobile ad hoc networks and wireless sensor networks. He has published more than 150 research papers in International/National journals & conferences and guiding PhDs in these areas. He is member of ACM, IEEE, ISTE, CSI, International Association of Engineers and Internet Society.