Cluster Head Influence based cooperative Caching in Wireless Sensor Networks

Ashok Kumar

***Abstract*— Cooperative caching harnesses the combined data storage capacity of memory constrained Sensor Nodes (SNs) for data caching. There are a number of Wireless Sensor Network (WSN) applications where, sink node may require recent history data communicated by a particular SN. In such situation, the data request at the sink needs to be served in short latency and with minimal energy consumption. A cooperative caching protocol termed as Cluster Head Influence based cooperative Caching (CHIC) has been proposed as an effective and efficient technique to achieve these goals concurrently. The proposed cooperative caching scheme is based on the influence of cluster heads in clustered WSN. We have proposed new cache admission control, cache discovery and cache replacement protocols under the CHIC cooperative cache management.**

***Index Terms*—Admission control, byte hit ratio, cache discovery, cache replacement, cluster head, query latency.**

I. Introduction

WSN field has emerged as an effective technology for distributed sensing and monitoring of a remote field/terrain which may not be accessible by conventional means of sensing/monitoring. One of the major drawbacks of WSNs is that SNs are highly energy constrained devices, having limited battery life and it is hard or rather impossible to rejuvenate/replace the batteries in inaccessible and inhospitable environments. Despite such constrains, WSNs are expected to perform their intended task of sensing/monitoring the sensing field for maximum possible time period. The short life span of WSN may lead to enhanced cost for continuous sensing/monitoring of the area of interest, as a lot of efforts and expenses are involved in frequent redeployment of SNs.

Majority of WSN applications are data centric which require transmission of sensed data to sink node situated inside/outside the sensing field, through single/multi hop transmission. Once the data is collected at sink, it preprocesses the data as per requirement of the application and further transmits to the end user via Internet or some other wireless/wired media. Due to presence of large number of SNs in the sensing field, vast amount of data is communicated to the sink during each sensing round resulting in large number of transmit/receive operations. As data transmission and reception operations are major sources of energy consumption in the WSN, considerable amount of energy could be preserved if number of data transmission/reception operations could be reduced. This could be achieved if data communication to the sink is performed only as per the request of the user/application via sink node. As the request pattern of data is hard to be predicted, sink node can request data from any region at any time. Therefore, SNs need to perform continuous periodic sensing to fulfill this objective and sensed data must be stored either by the same SN who has sensed it or by some other SN in the network, so that data could be provided to the sink node as and when request for such data is received. Since, the data storage capacity of the SN is limited, it can only store limited amount of data temporarily. However, collective data storage capacity of SNs combined together can be harnessed for data caching operation and sensed data is spread across distributed nodes throughout the sensor field.

There are numerous applications, such as tracking the movement of certain object/animal/vehicle, tracking the spread of certain phenomenon such as forest fire, creating panoramic view of an area by collecting the visual information transmitted by SNs equipped with micro-camera capable of capturing only a narrow view of area surrounding the node, etc., sink node may require recent history data reading along with present data readings transmitted by the SNs. This requirement can be easily fulfilled if sink node keeps storage of all history data it receives. But this requires very large storage capacity available at the sink, as hundreds of SNs keep on pumping sensed data continuously to the sink. Although, in many WSN protocols reported in literature, sink node has been assumed to have unlimited power, processing, and data storage resources available at its disposal; however, there may be a number of scenarios where such assumptions may not be valid and sink node may have limited resources.

Data caching is a well-researched field in wireless ad hoc networks and lot of caching protocols including optimized cache discovery, cache admission, cache replacement, and cache consistency processes are available in literature for such networks [1]-[11]. However, data caching is altogether a different and challenging task in WSNs due to (i) limited storage capacity of individual SNs, (ii) data request pattern (many-to-one data dissemination), and (iii) limited battery capacity of SNs. The fundamental objective of cooperative caching in WSN is to identify the SNs which will cache the data, deciding whether SN should cache the particular data or not, deciding which of the cached data of SN be deleted to accommodate new data, how long the data should be cached in the network, and deciding optimal path to get the query resolved with shortest possible latency and minimum communication overhead for energy efficient operation of WSN.

Considering the special requirements of WSNs to resolve above problems, we have proposed an energy efficient cooperative caching technique called Cluster Head Influence based

Ashok Kumar is working as Associate Professor E&CED, National Institute of Technology Hamirpur (HP) India-177 005. (Email: ashok@nith.ac.in)

cooperative Caching (CHIC), which effectively utilizes the clustering in WSN to optimize the query resolution. The proposed scheme utilizes the combined storage capacity of SNs present in the sensing field, thus increasing the size of cumulative caching. It is aimed to achieve low energy consumption per query response, low access latency, and creating optimal number of data copies in the network to avoid the wastage of cache space on one hand and provide enough replication of data for efficient query resolution on the other.

II. Network Model and Assumptions

Proposed scheme is designed for hierarchical WSNs in which SNs are divided into clusters. If it is to be used for flat network, the network first needs to be divided into clusters using any energy efficient clustering protocol available in literature. Our proposed scheme is directly applicable for such clustered networks where clusters and CHs required for cooperative cache implementation are already available in the network. Following assumptions are made to implement proposed caching scheme in WSN:

- N_T number of SNs are assumed to be deployed randomly in a sensing field of radius R with uniform node density ρ.
- SNs are static and are aware of their location coordinates.
- SNs are homogeneous and have same caching capacity.
- WSN is assumed to have been divided into clusters, each cluster having elected its own CH.
- All SNs communicate their sensed data to a single sink node situated at the center of circular sensing field.
- Data request (query) is initiated by the sink node only and requested data is always destined to the sink node i.e. all data request (query) paths originate at the sink and all data path (query resolution path) terminate at the sink only.
- Computational capability, data storage capacity, initial battery energy, and communication range of sink are higher than all other SNs deployed in the sensing field.
- The set of data items is denoted by $D = \{d_1, d_2, \ldots d_N\}$, N is the total number of data items and d_j $(1 \le j \le N)$ is a data identifier. D_i denotes the actual data for item with id d_i. Size of data item d_i is S_i. Data item can be originated from any SN.

III. Proposed Cooperative Caching Scheme

Cooperative caching protocols proposed in literature for WSN are based on the estimation of SN importance in given network topology and nodes are selected as data caching nodes, based on their relative importance. CHs in a hierarchical WSN are always selected based on their communication capability within their own cluster members [12]. As, all SNs communicate their sensed data through their respective CHs and further this data is relayed towards the sink via CHs only; this signifies the importance of CHs in data caching. In proposed protocol, a cluster head is allowed to cache data utilizing the combined storage capacity of its cluster members and has been termed as caching cluster head (CCH).

A. Cache Discovery Process

Cache discovery is the process by which sink locates the requisite data by broadcasting query related to the data item, within the sensor network. To avoid unnecessary communication query overhead, efficient and optimized cache discovery process is an essential requirement of any WSN cooperative caching protocol. Query initiated by sink is addressed in terms of data item id $D = \{d_1, d_2, \ldots d_N\}$, which essentially comprises of the sensing node id $N_{id} = \{d_1, d_2, \ldots d_T\}$ and time stamp $t_s = \{t_{s1}, t_{s2}, \ldots\}$. Time stamp t_s is the time at which the data is sensed. In the proposed protocol, it is assumed that whenever a SN senses an event, it communicates the sensed event to its CH by transmitting a data frame DF. The data frame comprises of SN id N_{id}, time stamp t_s, and TTL of the data item. Two SNs situated in different clusters can be assigned same node id N_{id} and since the process of assigning node id is cluster centric it is relatively easy due to small number of SNs in a cluster.

CHs may or may not carry out data aggregation/fusion operation on data received from their respective cluster members during one data sensing round. In any case, CH generates a data frame by adding its own cluster head id CH_{id} to the received frame from SNs, and relays it towards the sink. While this data frame traverse towards sink through various relay CHs on the way, each CH maintains a table of CH_{id} whose data they have routed. Since, each CH need to relay data of small number of CHs, the size of data table is quite small. However, this tabulation saves a lot of communication overheads during the query resolution process.

The proposed cache discovery process is illustrated in Fig. 1. Before sending query for a data item, sink first of all checks its own local cache. In case data item is found there and is valid, query is immediately served without broadcasting it any further. If the data item is not found in its cache, it broadcasts the query within its first hop CLUSTER called FH-CLUSTER with radio power R. All CHs residing within first hop CLUSTER search their local cache (local cache includes the cache of CH and its associated cluster members), if data is found within their local cache, query is served by sending the queried data item to the sink. If data item is not found in their cache, they search their routing table and check if such data was routed through them (simply by matching the CH_{id} of queried data item with CH_{id} in their respective routing table). Those CCHs for which CH_{id} of queried item does not match with any of CH_{id} in routing table, stop further broadcast of query. This process avoids query broadcast through those routes where data in not residing, thus saving considerable amount of energy in the network. Those CCHs which find routing information of queried data item in their respective routing table, broadcast the query to CCHs residing in CLUSTER with higher index (away from the sink). Those CCHs which are residing either in same CLUSTER or CLUSTER with lower CLUSTER index value (towards the sink) ignore the query. This process continues till the query is resolved. If queried data item is not available anywhere on the routing path, the query is served by the data originating node through its CH (as data item is always cached at originating node till expiry of data item TTL).

B. Cache Admission Control

When a node receives the data, the process of cache admission control decides whether the received data is to be cached or not. In WSN, each sensor node can store the received data in its local storage. In caching terminology, the sensor node, which stores the received data, is termed as caching node (CN). Caching of every received data at the CN is not possible as it requires large amount of memory at the CN and if proper and efficient caching decision is not made, it may lower the efficiency of caching scheme, leading to increase in data latency with lower probability of cache hits [8]-[11].

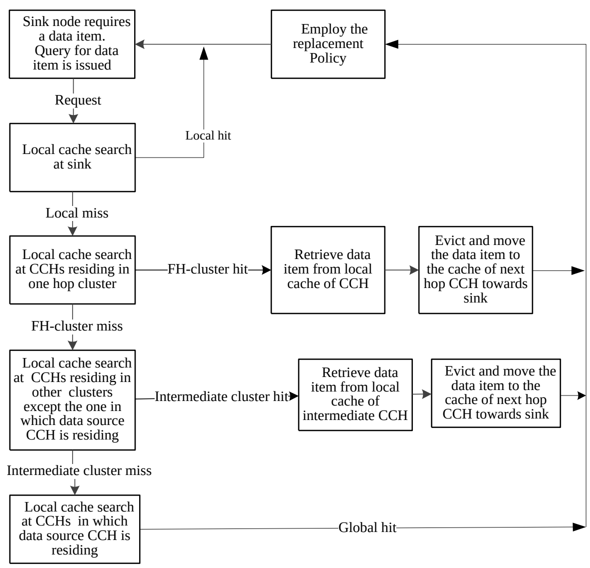


Fig. 6.2: Cache discovery and data retrieval process in CHIC protocol.

In our proposed scheme, the caching decision is based on two prime questions (i) whether the CN is appropriate caching site to store a particular data, and (ii) whether a particular data is appropriate data for caching at a particular CN. Data is admitted to the cache only if appropriate data is received at appropriate CN. Fitness/appropriateness of a CN or a data item for cache admission control is decided by significance of data item and CN. In our protocol, we have used the appropriate CHs for caching known as Caching Cluster Head (CCH). Although, every CH can act as CCH due to higher significance compared to other SNs deployed in the network; however there may be difference in the significance of CHs due to variation in their cluster members, adjacent CHs, and distance from the sink. In proposed protocol significance of CH has been defined in terms of Cluster Head Significance Index (CHSI) as defined below:

• *Cluster Head Significance Index (CHSI)*

CHSI describes the probability of cluster head to act as Caching Cluster Head (CCH) in the network. It is measured based on following parameters:

i. Cluster Head Influence Index (CHII)

The influence of a CH in the network is measured in terms of its one hop degree of connectivity to its neighboring CHs. If a CH is surrounded by more number of CHs, probability of a query to traverse through such CHs is also high; therefore, efficient query resolution can be achieved if such CHs are selected as CCHs.

ii. Node Influence Index (NII)

NII of a CH is measured in terms of number of nodes present in its own cluster. If a CH is having more number of SNs in its cluster, it has the availability of more cache for data storage; as it can use collaborative cache of all its cluster members, along with its own cache. Availability of larger cache size within such cluster enhances the probability of query resolution by such CHs.

iii. Location Influence Index (LII)

LII of a cluster head is measured as its distance in terms of hops from the sink. In case of our scenario, WSN is having single sink, and the data request/query is always originated at the sink and if the sink is not able to resolve it from its own cache, it broadcasts it within the network. To achieve minimum latency and minimum energy expenditure for query resolution, the query must get resolved at nearest possible CH from the sink.

As $CHSI \propto (CHII) \cap (NII)$ and $CHSI \propto \frac{1}{LII}$. Therefore,

$$CHSI = \left\lfloor \frac{CHII \times NII}{LII} \right\rfloor \tag{1}$$

CHSI provides a means for indexing the CH as per its fitness to act as CCH.

Other parameter which influences the cache admission protocol in WSN is fitness of a particular data item for caching at a particular CCH. Therefore, another parameter i.e. Data Popularity Index (DPI_i) has been defined to measure the fitness of data item d_i for its caching at a particular CCH.

• *Data Popularity Index (DPI)*

DPI is a measure and indexes the popularity of data at the sink. Such indexing of data popularity is useful for deciding the cache admission and cache replacement policy for an efficient cooperative caching scheme. In proposed protocol DPI has been based on following parameters:

i. Access Frequency of Data Item

Access frequency $f_i(t)$ of a data item d_i is a direct measure for data popularity of d_i at the sink. Frequently accessed data items are supposed to be more popular than other data items. Access frequency $f_i(t)$ for data item di in local cache of a CCH is given as:

$$f_i(t) = \frac{a_i}{\sum_{k=1}^{n} a_k} \tag{2}$$

Where a_i is mean access rate of data item d_i over a time period T and n is total number of data requests generated at the sink over time T.

ii. Time to Live (TTL) of Data Item

TTL of a data item is a measure of time for which a particular data item di is valid. Data item is declared invalid after expiry of its TTL period. A data item is assumed to have high DPI value if TTL of that data item is high. TTL value for a particular data item is decided by the SN and is based on the application. The data sensing node adds a data sensing time stamp t_s and TTL value t_{TTL} while transmitting the data to its CH. Data validity in the network is defined by (t_s - t_{TTL}) value.

iii. Distance of CCH from the Data Source

Data item sensed by a particular SN is not cached by the CCH within its own cluster, as in the proposed protocol the data is always cached by the sensing node for a time window based on the TTL value of data item. Therefore, the data item d_i is assigned higher DPI as it moves away from the data originating node i.e. closer towards the sink. The popularity of a data item d_i at a CCH is proportional to $d_{(CCH)i}$, the distance of CCH in terms of hops from data originating source of data item d_i. Taking into consideration all above mentioned parameters, DPI of a particular data item di can be given as:

$$DPI_i = f_i(t) \times TTL_i \times d_{(CCH)i} \quad (3)$$

Based on the fitness/appropriateness function of CH and data item expressed by (1) and (3), a new cache admission control policy has been proposed which allows caching of suitable data at a suitable CCH so that data request/query get resolved with minimum possible communication overhead and least possible data latency.

The proposed cache admission control policy allows the SN to cache all data items it has sensed and the data item remains in the cache of sensing node till the TTL expiry of data item. The node which senses the data item has been named as Data Originator Node (DON). This policy is proposed so as to avoid failure of any data request resolution as in worst case scenario the data item should be available at least at DON. Other SNs and CH of same cluster under which DON is residing are not allowed to cache the data item originated within same cluster. To increase proximity of the data items nearer to sink, it is always better to start caching of data items nearer to sink [9]. However, WSN utilizes reverse multicast data transmission policy for sensed data, where data sensed by the SNs is forwarded to the sink; whereas, query follows the multicasts data transmission policy, where single sink node multicast the query within the network. This type of conflicting data transmission policy in WSN makes the accomplishment of efficient data admission control a difficult task compared to MANET. Following policy has been employed for proposed cache admission control:

1. Initially all CHs present in the sensor network calculate their own CHSI using (1) and communicate CHSI value to the sink. On completion of this step sink node receives $[CHSI_1, CHSI_2, CHSI_3, ... CHSI_k]$, where k is number of CHs in the sensing field.
2. Sink node calculates $[CHSI_1, CHSI_2, CHSI_3, ... CHSI_k]_{\max}$, the CHSI index having the maximum value and communicates the $CHSI_{max}$ value to the CHs. Apart from communicating $CHSI_{max}$ value, sink node also decides the threshold value of cluster head significance index ($CHSI_{threshold}$) and communicates it to the CHs. This value is used by CHs while deciding the cache admission of a particular data item.
3. On receiving $CHSI_{max}$ value, each CH calculates normalized cluster head significant index value. For CH_j, normalized CHSI value is calculated as:

$$\left(CHSI_j\right)_{norm} = \frac{CHSI_j}{CHSI_{\max}} \quad (4)$$

4. Cache admission decision for data item d_i, received by CH_j is based on the following logic:

$$\left(CHSI_j\right)_{norm} \geq CHSI_{threshold} \quad (5)$$

If the logic of (5) is true, only then CH can act as CCH for a particular data item d_i, otherwise it simply route the data item to its next hop CH. However, even if the (5) is true, it does not guarantee the caching of data item di by CH_j, as the CH needs to check the CHSI index of its next hop CH on routing path i.e. $CHSI_{j+1}$ as well as the DPI indices DPI_i^j and DPI_i^j. DPI_i^j signifies the importance/significance of data item d_i at CH_j. Therefore, the caching decision for CH_j is based on the following logic:

$$\left\lfloor DPI_i^j \cap \left(CHSI_j\right)_{norm} \right\rfloor \geq \left\lfloor DPI_i^{j+1} \cap \left(CHSI_{j+1}\right)_{norm} \right\rfloor \quad (6)$$

Equation (6) requires the value of DPI_i^j and DPI_i^{j+1} i.e. data popularity index for data item d_i both at CCH_j as well CCH_{j+1}. However, soon after the deployment of WSN, when SNs initially start communicating their sensed data to the sink node, there may not be any data request/query from the sink. This preliminary sensed data also needs to be cached by some CCH in the network. In absence of any information regarding the data access rate during preliminary stage of sensor network, the value of access frequency $f(t)_i$ for data item d_i is assumed to be unity. Subsequently when the sink node starts sending request/query for certain data item, the CCH receiving the request starts calculating the access frequency $f(t)_i$ for that data item It is intuitive that if (6) is satisfied, CCH_j is the most appropriated node to cache the data item d_i, as it has better caching fitness as compared to its next hop CCH i.e. CCH_{j+1}. In the event of (6) is not true, the CCH_j simply needs to route the data item to CCH_{j+1}, without caching.

However, if cache of CCH_{j+1} is full then the problem of data caching becomes trivial, as CCH_{j+1} has to evict less popular data from its cache.

C. *Cache Replacement Policy*

Cache replacement policy is required when a CCH attempts to cache the data item but its cache is full. In such event, CCH has to evict some existing data from its cache to accommodate the new data item. To accommodate new data item, there is no option other than evicting an existing data item from the cache. The victim must be selected based on policy which results in minimum loss of information and minimum overheads [9]. In proposed protocol, cache replacement is carried out based on Data Popularity Index (DPI) value calculated as per (3). The proposed cache replacement works as follows:

- Data item originated at a particular node is always cached at that node and remains in the cache of that node till the expiry of its TTL value.
- CCH immediately evicts a data item from its cache as soon as its TTL value expires.
- A cost based cache replacement policy is proposed for eviction of cached items from local cache of CCH. The cost is based on DPI value of the data item. CCH always keeps the track of maximum and minimum DPI value of data items present in its local cache. (i.e. DPI_{max} and DPI_{min}). DPI_{max} and DPI_{min} value is updated regularly on arrival/eviction of new data item. When a new data item arrives at the CCH, it calculates its DPI value and compares the calculated DPI value with DPI_{max} value of data items stored in its local cache. If the DPI value of arrived item is more than the DPI_{max} value, it initiates the process of cache replacement; otherwise, the data item is routed

to next CCH. In the event of cache replacement, CCH first replaces the item with minimum DPI value i.e. DPI_{min}. The evicted item is routed to neighboring CCH for storage if its TTL value is still valid as it may be requested by the sink.

TABLE I
SIMULATION PARAMETERS

Parameter	Default value	Range
Network diameter	100 meters	50~400 meters
Number of nodes	400	100~500
Initial Energy of node	2 Joule	
Data packet size (k)	100 byte	
Threshold distance (d0)	87 meters	
Cache size	800 KB	200~1400 KB
Time to live (TTL) of data	300 sec	
Data rate	10 Kbps	
Transmission range	10 meters	
Zipfian skewness parameter (z)	0.8	
Mean query generate time	5 sec	2~100 sec
E_elect	50 nJ/bit	
ϵ_{fs}	10 pJ/bit/m2	
ϵ_{mp}	0.00134 pJ/bit/m4	
E_{aggr}	5 nJ/bit/signal	

IV. PERFORMANCE EVALUATION

A. Simulation Parameters and Performance Metrics

The simulation parameters are given in Table I. Performance of proposed protocol is compared with NICoCa [8] and is evaluated using following performance metrics:

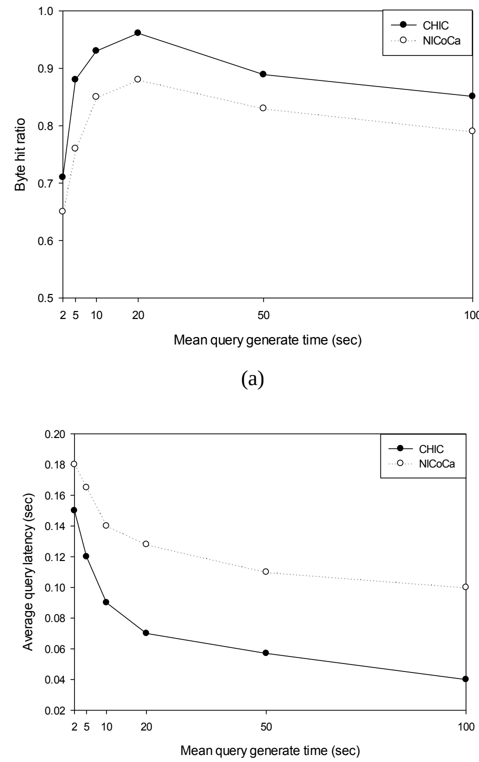
• *Average query latency (T_{mean})*: The query latency is the time elapsed between transmission of query and its resolution at the sink. Average query latency (T_{mean}) is the query latency averaged over all the queries generated by the sink.

• *Byte hit ratio (B)*: It is the ratio of 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 cache management.

B. Simulation Results

• *Effect of mean query generate time (T_q)*

The effect of mean query generate time (T_q) on byte hit ratio (B) and average query latency (T_{mean}) has been illustrated in Fig. 2. Mean query generate time is the time between two consecutive queries, averaged over total number of queries. T_q has been varied from 2 to 100 seconds. It can be seen from Fig. 2(a) that byte hit ratio increases with increase in T_q, reaches its maximum value around 20 and then again starts reducing. Initially, when the T_q value is very small, more number of queries are generated per unit time and very little time is available for settlement of cache to make required data items available at cache of CCHs near the sink. Therefore, all queries may not get resolved at local/FH-CLUSTER/intermediate CLUSTER cache and global hits are inevitable. As the T_q is increased, byte hit ratio starts increasing, since network has to resolve less queried per unit time, which gives enough time for settlement of network cache for providing highly queried data item at the cache of CCHs closer to the sink. This accounts for higher byte hit ratio. However, if Tq is further increased, it does not result in better hit ratio, in actual the hit ratio starts reducing. The reason behind this behavior is that, though enough time is available for cache to settle down between the queries, but due to expiry of TTL value of various cached data items, eviction



(a)

(b)

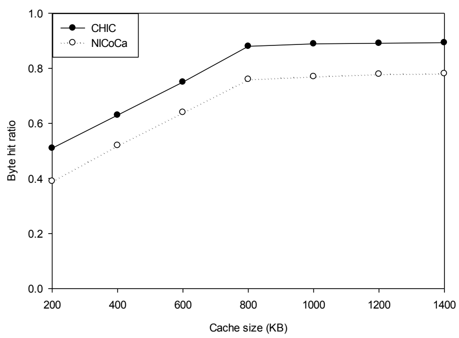
Fig. 2: Effect of mean query generate time on (a) byte hit ratio, and (b) average query latency.

process of these data items starts in the network, again destabilizing the balance of cache in the network, which results in lower byte hit ratio. Similar trends are observed for CHIC and NICoCa. Proposed CHIC protocol provides higher byte hit ratio compared to NICoCa due to better cache management policy proposed in CHIC. The cache admission control of CHIC ensures the availability of highly required data at CCH near the sink. Only small number of global hits are observed in case of CHIC protocol thus, proving the superiority.

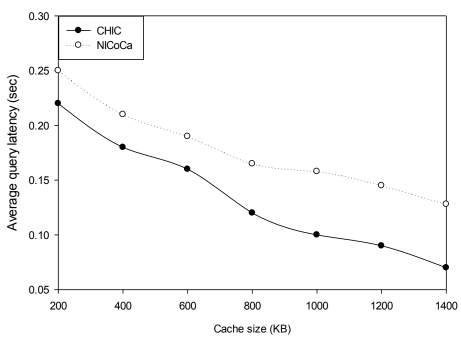
Fig. 2(b) illustrates the variation in average query latency (T_{mean}) with T_q. It can be observed that at lower T_{mean}, the T_q value is high. It is due to the fact that generation of more number of queries per unit time results in more congestion and data collisions in the network thus, increasing the time in query resolution. Intuitively, T_{mean} reduces with increase in T_q. However, rate of decrease of T_{mean} keeps on reducing. This is due to the fact that if mean query generation time is very large, expiry of TTL value of large number of data items starts in between, resulting in small number of cache hit and in turns reduction in rate of decrease of T_{mean}. The cache management policy of proposed CHIC protocol ensures the availability of highly queried data items near the sink. Therefore, average query latency in proposed protocols is always less than NICoCa protocol, irrespective of the mean query generate time. Thus, CHIC outperforms the NICoCa in terms of average query latency.

• *Effect of cache size*

Fig. 3 illustrates the effect of cache size on byte hit ratio (B) and average query latency (T_{mean}). The cache size of SNs is varied from 200 KB to 1400 KB. It can be seen in Fig. 3(a) that initially the byte hit ratio increases with increases in the cache size. It is an obvious result, since higher size of cache helps the placement of data items at local/FH-CLUSTER/intermediate CLUSTER cache thus, increasing the probability of cache hits. However, the byte hit ratio becomes almost constant after about 800 KB cache size. This is due to the fact that in network, the data items start getting stale and evicted from the cache, due to expiry of their TTL. As the rate of data sensing in the sensing field is assumed constant, therefore an appropriate value of cache is enough to maintain the optimum byte hit ratio. Increasing the cache size beyond this limit does not add substantially to the byte hit ratio. It can be seen that proposed CHIC protocol provides better byte hit ratio compared to the NICoCa protocol.



(a)



(b)

Fig. 3: Effect of cache size on (a) byte hit ratio, and (b) average query latency.

Fig. 3(b) illustrates the effect of cache size on T_{mean}. Initially, when the cache size is small, T_{mean} is high, because due to smaller cache availability the queried data items may be available away from the sink. The data item availability nearer to the sink is better in CHIC protocol as can be seen by the lower value of T_{mean} compared to NiCoCa. With increase in cache size, T_{mean} starts reducing. Initially, the rate of decrease is high which reduces with further increase in the cache size. This behavior is attributed to the expiry of TTL value of data items and their eviction from the cache. Therefore, to maintain certain minimum standard value of T_{mean} and B, minimum value of cache size (800 KB in our case) is required. Increasing the cache size beyond this is futile. Byte hit ratio and average query latency of proposed CHIC protocol is better than NICoCa protocol irrespective of SN cache size. It is attributed to the efficient cache management policy adopted in the proposed protocol, which ensures the availability of highly queried data items closer to the sink; thus, enhancing the byte hit ratio and reducing the average query latency.

V. Conclusion

Proposed cooperative caching scheme CHIC exploits the influence of CHs already present in the sensing field. In CHIC protocol, CHs are used as CCH to cache the data. They can efficiently use the caching space of all SNs present in their respective clusters. In clustered network all data items are routed through the CH, this helps in efficient query resolution. We have proposed new cache admission control, cache discovery and cache replacement protocols under the CHIC cooperative cache management. The simulation results demonstrate the effectiveness of proposed protocol compared to similar caching protocol NICoCa.

REFERENCES

[1] G B. Raj, S.K. Vishvakarma, A.K. Saxena and S. Dasgupta, “Techniques for Low Power SRAM Design,” National Conference on Design Techniques for Modern Electronic Devices, VLSI & Communication Systems, pp. 55-61, 2007.

[2] A. Dunkels, J. Alonso and T. Voigt, “Distributed TCP Caching for Wireless Sensor Networks,” Annual Mediterranean Ad-Hoc Networks Workshop, pp. 1-17, 2004.

[3] A. Ayadi, P. Maille and D. Ros, “Improving Distributed TCP Caching for Wireless Sensor Networks,” IFIP Annual Mediterranean Workshop, pp. 1-6, 2010.

[4] 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.

[5] S. Gupta, S. Mittal, S. Dasgupta and A. Mittal, “MIMO Systems For Ensuring Multimedia QoS Over Scarce Resource Wireless Networks," ACM International Conference On Advance Computing, 2008.

[6] 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.

[7] M.N. Al-Ameen and M.D.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.

[8] D. Nikos, K. Dimitrios and M. Yannis, “Cooperative Caching in Wireless Multimedia Sensor Networks,” Mobile Networks and Applications, Vol. 13, No. 3-4, pp. 337-356, 2008.

[9] T.P. Sharma, R.C. Joshi and Manoj Misra, “Cooperative Caching for Homogeneous Wireless Sensor Networks,” International Journal of Communication Networks and Distributed Systems (IJCNDS), Vol. 2, No. 4, 2009.

[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] Amir Shiri, Shahram Babaie and Javad Hasan-zadeh, “New Active Caching Method to Guarantee Desired Communication Reliability in Wireless Sensor Networks,” Journal of Basic and Applied Scientific Research, Vol. 2, No. 5, pp. 4880-4885, 2012.

[12] Xiaorong Zhu, Lianfeng Shen, and Tak Shing Peter Yum, “Hausdorff Clustering and Minimum Energy Routing for Wireless Sensor Networks,” IEEE Transaction on Vehicular Technology, Vol. 58, No. 2, pp. 990-997, Feb 2009.