2013 IEEE 27th International Conference on Advanced Information Networking and Applications

Distributed Spatial Correlation-based Clustering for Approximate Data Collection in WSNs

Zhidan Liu*†, Wei Xing✉ *†, Bo Zeng*†, Yongchao Wang†, Dongming Lu*†
*College of Computer Science, Zhejiang University, Hangzhou, 310027, P. R. China
†Cyrus Tang Center for Sensor Materials and Applications, Zhejiang University, Hangzhou, 310027, P. R. China
Email: {danielliu,wxing,zb9364,ychwang,ldm}@zju.edu.cn

***Abstract*—Grouping sensor nodes with similar readings into the same cluster and scheduling them to alternately report their sensing readings is an effective and efficient method to perform approximate data collection, which exploits the tradeoff between data quality and energy consumption. However, to partition all sensor nodes into exclusive clusters while incurring as less communication overhead as possible in a distributed manner is a challenging task. In this paper, by exploiting inherent spatial and data correlation in wireless sensor network, we have proposed the distributed spatial correlation-based clustering algorithm to complete this tough mission. With nodes information exchange in certain region and the novel ranking strategy, our clustering algorithm can terminate in a small number of iterations. Extensive simulations show that the proposed algorithm outperforms three other noteworthy clustering algorithms, namely EEDC, ASAP and DClocal, on some key metrics, such as number of clusters, energy consumption for clustering, average dissimilarity of clusters and residual energy level of cluster heads.**

***Keywords*—Wireless Sensor Network; Clustering; Spatial Correlation; Energy Efficient;**

I. Introduction

Wireless sensor networks (WSNs) have been widely used in many applications for continuously collecting information of surrounding environment [1]. In these data-driven applications, sensor nodes sample environmental data and transmit to *Sink* (or called base station) periodically for subsequent utilization.

It is generally acknowledged that data collection is a fundamental but challenging task as the limited communication bandwidth and energy budget [2]. Fortunately, most applications of WSNs can tolerate certain accuracy loss of sensor data to perform data analysis or decision making [3]. By exploiting the tradeoff between data quality and energy consumption, approximate data collection can be a wise choice for long-term data collection. Although the number of sampling nodes and data communication quantity have reduced, approximate data collection still can acquire environment data with high fidelity. This is done either with probabilistic models [4] [5] [6] or with other data approximation approaches [7] [8]. Due to the dense and high redundant deployment of WSNs, neighboring sensor nodes usually have similar readings and this phenomenon is named as *spatial correlation* which is frequently exploited in data collection. Spatial clustering, which takes advantage of the ubiquitous spatial correlation in WSNs, could group sensor nodes with similar sensing readings into the same cluster [7] [8] [9] [10]. Obviously, spatial clustering supports further procedures of approximate data collection and can save more energy of WSNs. Because of sharing similar data distribution, instead of having all nodes in the same cluster work simultaneously, it is more energy-efficient to schedule nodes to work alternately to prolong the lifetime of WSNs. Therefor, finding the sets of sensor nodes with similar readings is usually the first procedure of approximate data collection, no matter with model-based methods [5] or with other data restoration approaches [7]. However, spatial clustering in an energy-efficient and distributed manner is a tough work, not only because the resource constraints, but also because it is hard to discover the spatial correlation pattern in a distributed way. Besides, dynamic nature of physical environment makes it difficult to maintain these clusters.

✉Corresponding author.

In this paper, to solve the problem of spatial clustering for approximate data collection, we propose the **D**istributed **S**patial **C**orrelation-based **C**lustering (DSCC) algorithm. By exploiting spatial correlation and application-specific parameters (namely error-tolerance threshold ε and spatial correlation range R_{sc}), our algorithm can select some well-distributed nodes with more residual energy to act as cluster heads. In addition, DSCC makes effects to decrease the variance of cluster sizes to balance the energy consumption of all sensor nodes and minimize reading dissimilarity of nodes in the same cluster. Foremost, DSCC will pay less communication cost for clustering operation when compared with other distributed spatial clustering algorithms.

The rest of this paper is organized as follows. Section II briefly discusses related work on clustering in WSNs. System model and problem statement are described in section III. Details of DSCC algorithm are elaborated in section IV. Performance evaluation with extensive simulations is presented in section V. Finally, we conclude our paper in the last section.

II. Related Work

Clustering is a popular technique for topology management and plays an important role in the objective of energy conservation in WSNs [11]. Recent years many clustering algorithms have been proposed in the literature. LEACH [12] is a well-known clustering algorithm which selects cluster heads randomly with a fixed probability p. Taking the residual energy into account, HEED [13] selects these nodes whose probabilities approach 1 firstly in an iterative manner as cluster heads. However, these kinds of clustering algorithms [11] have neglected the inherent spatial and data correlation in WSNs.

To perform efficient data aggregation or approximate data collection, it is necessary to take the spatial correlation into the design of clustering algorithms. A renowned example for data aggregation is CAG [14], which is a query-driven clustering algorithm and can provide approximate aggregation results within user-predefined error-tolerance threshold. YEAST [15] also takes advantages of spatial correlation and groups nodes into correlated regions for event-based data aggregation.

In the context of approximate data collection, it is preferable to discover similar node sets firstly and then properly schedule sensor nodes to report readings to *Sink* by turns, in order to reduce redundant communication. B. Gedik et al. in ASAP [6] provide a distributed sensed-driven clustering algorithm to group nodes with similar readings. In ASAP, cluster heads are selected based on probability which takes relative energy level of nodes and cluster count factor f_c into account. Ordinary node will score each cluster head candidate according to hop distance and data distance, and join the cluster with the highest score. Leveraging the probability-based method to select cluster heads, ASAP needs a number of iterations to select enough cluster heads to "cover" all nodes and there is no guaranty on the "good" distribution of cluster heads. L. Chong et al. [7] propose an energy-efficient data collection framework called EEDC, which measures the similarity between two sensor nodes with raw readings both on magnitude and trend. EEDC models the clustering problem as clique-covering problem and proposes a centralized algorithm to partition sensor nodes into clusters. Another centralized algorithm proposed in [8] is called DCglobal, which names cluster head as representative node (*R-node*) and models the selection of *R-nodes* as a set-cover problem. By utilizing the concepts of *data coverage range*, where data coverage range of a sensor node is the set of sensor nodes whose readings are very close to this node, and *partial order relation*, DCglobal solves the set-cover problem in a centralized fashion and selects *R-nodes* as few as possible. However, in order to accumulate enough raw sensor readings at *Sink* for similarity measure between nodes, abundant communication overhead for spatial clustering will be triggered in both EEDC and DCglobal. DClocal [9] is a distributed version of DCglobal. By exchanging the sensing readings and energy level among neighboring sensor nodes in a limited region, each node sets a counter according to its energy level and data coverage range, and later dynamically changes the state depending on its own counter which counts down through time or state alterations of other neighboring nodes. Finally, those sensor nodes with higher energy level and larger data coverage range will become the *R-nodes*, and other ordinary nodes choose their *R-nodes* respectively during state transitions. Relying on the counters to select cluster heads, rigorous time synchronization is required to complete the clustering operation accurately in DClocal. Furthermore, there are some other work [5] [10] [16] [17] that involves spatial clustering. However, most of these work has no specific or strict requirements on similarity measure between nodes.

III. Problem Statement

A. System Model

In this paper, we consider a WSN consisting of N sensor nodes randomly deployed in a size of $L\times L$ square sensing field in uniform distribution. We denote sensor node i as s_i, and the sensor node set is $S=\{s_1,s_2,...,s_N\}$, where $|S|=N$. There is a powerful *Sink* node located outside of the sensing field. All sensor nodes and *Sink* node are stationary after deployment. We assume all sensor nodes are homogeneous and have the same initial energy when deploying, yet they may have different residual energy when clustering begins. Sensor nodes can estimate the approximate distance through *Received Signal Strength Indication* (*RSSI*). We assume loose time synchronization among sensor nodes is guaranteed. This assumption could hold true as time synchronization is one of the fundamental requirements of WSNs and there have been a lot of research work on this problem [18]. Besides, it is relatively easy to guarantee loose time synchronization in WSNs. Last and the most important, we assume each sensor node can adjust the transmission power to change the communication range, and single-hop communication can be built between an active node and the *Sink*. Actually, single-hop communication assumption has been made in many previous work [7] [8] [12] [13], and transmission power control can be supported by some real sensor node, e.g., Berkeley motes [19]. Even in large-scale network, some more powerful relay nodes could be deployed to partition the network into hierarchical architecture with similar method presented in EEDC [7], so that direct communication between sensor nodes and *Sink* or relay nodes could be realized.

We adopt a simple but popular radio model introduced by LEACH [12] to estimate energy dissipation for sensor nodes. To transmit and receive a l-bits packet over distance d, energy consumption is given by following equations respectively.

$$E_{Tx}(l,d)=\begin{cases}(E_{elec}+\varepsilon_{fs}d^2)\cdot l, & \text{if } d<d_o \quad (1)\\ (E_{elec}+\varepsilon_{mp}d^4)\cdot l, & \text{if } d\geq d_o \quad (2)\end{cases}$$

$$E_{Rx}(l)=E_{elec}\cdot l \tag{3}$$

where E_{elec} is the baseline to run transmitter or receiver circuitry. $\varepsilon_{fs}d^2$ and $\varepsilon_{mp}d^4$ are energy consumption to amplifier signal of radio, and which one used depends on the distance between sender and receiver with distance threshold d_o.

B. Problem Statement

Spatial clustering considers a problem that partitions all sensor nodes into exclusive clusters by making utilization of the inherent spatial correlation in WSNs [7] [8] [9] [10]. Within the same cluster, sensor readings dissimilarity of any two nodes meets a user-defined *error-tolerance threshold* ε. To accelerate the progress of clustering, *spatial correlation range* R_{sc} which shares the same meaning with *gmax_dist* in EEDC [7] that defines the maximal geographic distance between two nodes with similar readings can be taken as a measurement of spatial correlation. Note that both ε and R_{sc} are application-specific. To effectively and efficiently support approximate data collection, the most important objective of spatial clustering is to generate as few clusters as possible to reduce the number of simultaneously sampling nodes. Besides, the variance of cluster sizes should be minimized so that each node can roughly work the same amount of time. Furthermore, to ensure the stability of a cluster, the dissimilarity among cluster members should be minimized too. Formally, the problem of spatial clustering could be defined as follows.

Definition 3.1: *spatial clustering problem*. For a WSN with a collection $S = \{s_1, s_2, ..., s_N\}$ of N sensor nodes, the whole sensor network can be partitioned into definite node sets $C = \{C_1, C_2, ..., C_y\}$, where $\bigcup_{a=1}^{y} C_a{=}S$. For $\forall s_i, s_j \in C_a$, sensor reading dissimilarity meets $\hbar(s_i, s_j) \leq \frac{\varepsilon}{2}$ where $\hbar$ is similarity measure function in certain metric, and $dist(s_i, s_j) \leq R_{sc}$ in Euclidean metric. Spatial clustering aims to obtain $\min y$.

Just as demonstrated by DClocal [9], the spatial clustering problem could be modeled as a set-cover problem which is an *NP-hard* problem and there are no approximation algorithms with constant approximation factors.

IV. The DSCC Algorithm

A. Preliminaries

To measure the sensor readings dissimilarity between two nodes, we employ *Manhattan distance*, which is also adopted by [2] [7] [9] [10] for similarity measurement, as the similarity function f_{md} that is defined as

$$f_{md}(s_i, s_j) = \frac{\sum_{k=1}^{q} \left| v_k(s_i) - v_k^{'}(s_j) \right|}{q}. \quad (4)$$

In Eq.(4), $v(s_i) = \{v_1, v_2, ..., v_q\}$ and $v^{'}(s_j) = \{v_1^{'}, v_2^{'}, ..., v_q^{'}\}$ are the sensor reading serials of s_i and s_j respectively. It is worthy to note that the selection of an appropriate similarity function depends on specific requirements of applications. Two sensor nodes, say s_i and s_j, are called *similar nodes* if their geographic distance is smaller than R_{sc} and their dissimilarity distance meets $f_{md}(s_i, s_j) \leq \frac{\varepsilon}{2}$.

Given the similarity function, we have following definitions.

Definition 4.1: *similarity coverage rate C_r*. Assume there are n neighboring nodes of s_i in the range of R_{sc}. Among the n neighboring nodes, there are m nodes that are similar nodes of s_i. The neighboring node set of s_i is denoted as $NBR(s_i)$, and the similar node set is denoted as $SN(s_i)$. Then similarity coverage rate of node s_i can be defined as

$$C_r(s_i) = \frac{|SN(s_i)|}{|NBR(s_i)|} = \frac{m}{n}. \quad (5)$$

Definition 4.2: *similarity difference rate Sr*. For node s_i with its corresponding similar node set $SN(s_i)$, the similarity difference rate of s_i is defined as

$$S_r(s_i) = \frac{\varepsilon - \frac{\sum_{s_j \in SN(s_i)} f_{md}(s_i, s_j)}{m}}{\varepsilon}. \quad (6)$$

Theoretically, $C_r(s_i)$ reflects the "covering" power of sensor node s_i, and $S_r(s_i)$ expresses the similar degree between s_i and its similar nodes. We estimate the representative capability of a sensor node via these two parameters. Notice that, both $C_r(s_i)$ and $S_r(s_i)$ fall in the range of $[0, 1]$.

B. Algorithm

The clustering operation in our proposed DSCC algorithm is initiated by the *clustering-command* message broadcasted by the *Sink* node. Through the message, each node obtains the user-defined error-tolerance threshold ε and spatial correlation range R_{sc}. After that, each sensor node adjusts the transmission power to broadcast a "*Hello*" message (*H-msg*), which contains its residual energy and recent reading serial, to neighboring nodes in the range of R_{sc}. Based on the local information, each node decides whether to become a *cluster head candidate* (CHC) according to its energy level and representative capability. Later, all CHC nodes will compete to be the final *cluster head* (CH), and all non-CH nodes will choose an appropriate cluster to join. Details of the clustering procedure are elaborated in what follows.

1) Selection of cluster heads: Right after collecting all the neighboring information from nodes in range of R_{sc}, each node computes its own similarity coverage rate and similarity difference rate according to Eq.(5) and Eq.(6) respectively. Node s_i will become a CHC node if it meets the following two qualifications:

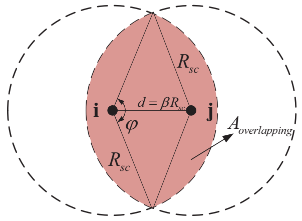
- ***Q1***: Relative energy level $E_R(s_i)$ is above the average energy of s_i's neighboring nodes, i.e.

$$E_R(s_i) = \frac{E(s_i){\cdot}(n+1)}{E(s_i) + \sum_{s_j \in NBR(s_i)} E(s_j)} > 1, \quad (7)$$

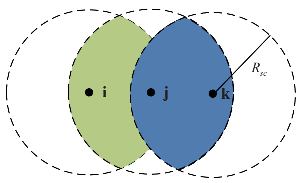
  where $E(s_i)$ is the residual energy of node s_i.
- ***Q2***: Representative capability $p(s_i) \geq \delta$, where $p(s_i)$ is computed as Eq.(8), in which α is an adjustable parameter to decide the relative importance of C_r and S_r. Here, δ is a pre-defined threshold to control the quantity of potential CHC nodes.

$$p(s_i) = \alpha{\cdot}C_r(s_i) + (1-\alpha) \cdot S_r(s_i). \quad (8)$$

If sensor node meets the two qualifications, then it decides to be a CHC node and broadcasts competition message (*C-msg*) which includes information about this CHC node, such as ID and the size of similar node set $SN(s_i)$, to nearby nodes to compete for the role of CH. In the context of spatial correlation, we define that node s_i *defeats* node s_j if and only if they are similar nodes, $SN(s_i)$ contains more nodes than $SN(s_j)$ and more than half nodes of set $SN(s_j)$ are in set $SN(s_i)$, i.e. $|SN(s_i)| > |SN(s_j)|$ and $|SN(s_i) \bigcap SN(s_j)| > \frac{|SN(s_j)|}{2}$, which mean that node s_i can "cover" more than half of s_j's similar nodes. However, it will consume a lot of energy to directly find the common items of two nodes' similar node sets in the distributed situation as considerable messages are needed. Thanks to the uniform distribution of sensor nodes,



(a) Competitive radius



(b) Defeated one by one

Fig. 1. Sample competitions of CHC nodes

we can solve this problem with an approximative method via the distance between two nodes. If node s_i *defeats* node s_j, besides meeting the conditions that they are similar nodes and $|SN(s_i)| > |SN(s_j)|$, their overlapping communication area should be more than half. Formally, assume the geographic distance between node s_i and s_j is $d = \beta R_{sc}$, obviously $0 < \beta < 1$. Then we can formulize the contrast problem of set elements as

$$R_{overlapping} = \frac{A_{overlapping}}{\pi R_{sc}^2} > \frac{1}{2}. \tag{9}$$

With the help of Fig.1(a), we can express $A_{overlapping}$ as Eq.(10) further, where angle φ is expressed as Eq.(11).

$$A_{overlapping} = \frac{(2\varphi - \beta\sqrt{4-\beta^2})\cdot R_{sc}^2}{2} \tag{10}$$

$$\varphi = arccos(\frac{\beta^2 - 2}{2}) \tag{11}$$

Combining equations (9), (10) and (11), we learn that when $0 < \beta < 0.8079$, the overlapping communication area of the two nodes is more than half, i.e., node s_i *defeats* node s_j. Details of solving process are omitted due to space limitation.

Therefore, each CHC node can adjust the transmission power and transmit *C-msg* only in the range of $r = \beta R_{sc} \approx 0.8R_{sc}$, which is called as *competitive radius*. The competing rules of CHCs are as follows: when CHC node s_i receives *C-msg* from s_j, s_i checks whether node s_j is in its similar node set firstly. If s_j is contained in $SN(s_i)$, s_i checks whether the size of its similar node set is larger than s_j's, where residual energy level is the tie-breaker. If true, we say node s_i *defeats* node s_j, otherwise node s_i is *defeated* by node s_j.

Due to the inexistence of global information about the distribution of CHC nodes, it is quite difficult to select a certain amount of appropriate CHs. For example in Fig. 1(b), if CHC node s_j *defeats* CHC node s_k, at the same time s_i *defeats* s_j, then it is very likely that finally there will be no CHs to "cover" node s_k and its similar nodes. To avoid this case, DSCC ranks each CHC node in a competitive scope, i.e., in the range of competitive radius r. At the initial phase, each CHC node ranks itself with number 1. During the CHC competition phase which will last for time T_{CHC} to make sure all *C-msgs* can be received, CHC node s_j will increase its ranking by 1 when it receives a *C-msg* from a stronger CHC node s_i, namely s_j is *defeated* by s_i. At the expiry of T_{CHC}, each CHC node confirms its own ranking and DSCC will select the final CHs in an iterative manner by exploiting these rankings. At each node, the clustering process requires a small number of iterations denoted as N_c. Each step takes time T_{rank} which is long enough to ensure that messages from any neighboring node in range of R_{sc} can be received. During any iteration, each CHC node checks whether its ranking equals to 1, if so it selects itself as a final CH and broadcasts a CH advertising message (*A-msg*) which only includes its node ID to neighbors in range of R_{sc}. Note that other information, e.g., residual energy and size of similar node set have been broadcasted early. After receiving an *A-msg*, any ordinary node or CHC node whose ranking is larger than 1 will add this CH to its $list_{CH}$ and consider itself as being "covered" by a CH node. At the expiry of T_{rank}, every "uncovered" CHC node reduces its ranking a half and goes to next iteration. Totally, CH selection phase will last for time $T_{CH} = N_c \cdot T_{rank}$. The pseudo-code for CHs selections is outlined in **Algorithm 1**.

Algorithm 1: cluster heads selection of DSCC

```
/* Ranking CHC nodes */
All nodes set timer to $T_{CHC}$;
if $s_i$ is CHC node then
    $ranking(s_i) \leftarrow 1$;
    Transmit a C-msg with competitive radius $r$;
    Listen for C-msgs for time $T_{CHC}$, add received
    competitive CHC nodes info to $list_{CHC}$;
    for each $chc \in list_{CHC}$ do
        if $chc$ defeats $s_i$ then
            $ranking(s_i)++$;
/* Iteratively select CH nodes */
All nodes set timer to $T_{CH}$;
if $s_i$ is CHC node then
    repeat
        if ranking($s_i$)==1 then
            Transmit an A-msg with radius $R_{sc}$;
            $s_i \leftarrow$ CH;
        else
            Listen for A-msgs for time $T_{rank}$;
            if received A-msg from stronger CHC then
                $s_i \leftarrow$ non-CH;
            else
                $ranking(s_i) = \lceil \frac{ranking(s_i)}{2} \rceil$;
    until $s_i$ becomes CH or non-CH;
```

2) Formation of clusters: After N_c iterations, a certain number of CH nodes are elected. Each non-CH node calculates *attracting score* for each CH node in its $list_{CH}$ according to relative distance and similarity degree. For example, at node s_i, *attracting score* of a CH node in its $list_{CH}$, say s_{ch}, is scored as

$$G_{s_i}(s_{ch}) = \frac{R_{sc} - dist(s_i, s_{ch})}{R_{sc}} + \frac{\varepsilon - f_{md}(s_i, s_{ch})}{\varepsilon} \cdot \lambda. \quad (12)$$

In Eq.(12), $dist(s_i, s_{ch})$ is the geographic distance between these two nodes, $f_{md}(s_i, s_{ch})$ is the dissimilarity measured by *Manhattan distance*, and positive real number λ is the similarity importance factor to weight the importance of data similarity. Among all the feasible CH nodes, non-CH node chooses the one with the highest score as final CH and joins the corresponding cluster by sending a join message (*J-msg*). Those still "uncovered" non-CH nodes are forced to be CH nodes at the end of time T_{CH}. The clusters formation phase will last for time T_{CM}. At the expiry of T_{CM}, all CH nodes send the cluster information which includes cluster head ID and cluster member list to the *Sink*.

3) Maintenance of clusters: After the clustering operation, sensor nodes in the same cluster are highly data and spatial correlated. Then it is desirable to schedule sensor nodes work alternately to perform approximate data collection [7] [20], so that the network lifetime of WSN could be extended. Here we suggest a scheduling scheme that in each cluster, CH keeps on working and other nodes turn into work state with probability ρ like randomized intra-cluster scheduling method [7]. After collecting readings of all active cluster members, CH restores data following the method in EEDC for all nodes in the cluster and transmits the results to the *Sink*. An advantage of this method is that it facilitates the maintenance of clusters. Explicitly, cluster maintenance is triggered in two cases. One case is that CH runs energy low and the other case is that CH detects that more than one cluster member report seriously different readings, i.e., local spatial correlation of this cluster changes. It is worthy to note that spatial correlation is very stable in many applications even if the monitored phenomenon changes dramatically. Cluster should be split into several new clusters if at least one case comes true. To suppress excessive growth of the number of clusterss, *Sink* node should broadcast *re-clustering* command when the number of clusters increases to beyond a threshold, e.g., user-defined $Max_{clusters}$.

C. Discussion

In this subsection, we firstly analyze the message complexity and time complexity of DSCC algorithm, and then discuss the setting of several timer in DSCC.

Lemma 1. The message complexity of clustering is $O(N)$, where N is the number of sensor nodes.

Proof. In the phase of information exchanging, there are totally N *H-msgs*. At most there will be rough $N/2$ nodes to become CHC nodes and $N/2$ *C-msgs* are produced accordingly. Assume eventually k CH nodes are elected and they send k *A-msgs* to neighboring nodes, the left $N-k$ nodes choose CH node respectively and $N-k$ *J-msgs* are sent. As a result, there are totally $N + N/2 + k + (N-k) = 2.5N$ messages, i.e., the message complexity is $O(N)$.

Lemma 2. The maximal iterations of DSCC algorithm is $\left\lceil \log_2 \frac{\pi R_{sc}^2 N}{2L^2} \right\rceil$, where L is length of the square sensing field and R_{sc} is spatial correlation range.

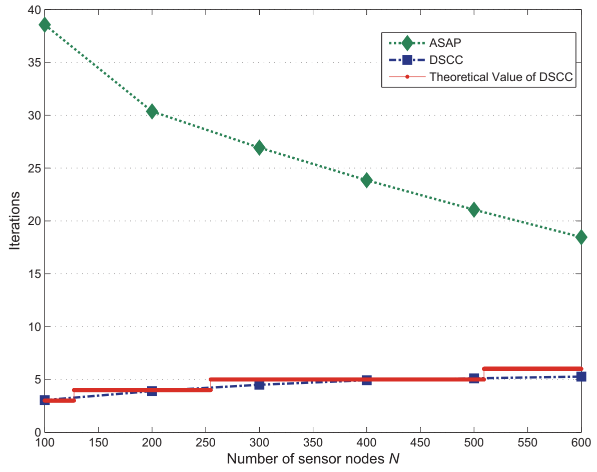


Fig. 2. Iterations for clustering operations with simulation parameters $L = 100m$, $R_{sc} = 20m$, $\varepsilon = 0.5$, $\delta = 0.6$, $\alpha = 0.5$, $\lambda = 1.0$

Proof. Actually the maximal iterations is decided by the maximal ranking, i.e., the maximal number of CHC nodes in a range of R_{sc}, in the sensor network. We assume there will be C clusters in the $L \times L$ sensing field, and then averagely each cluster will have N/C sensor nodes. Assume a node will become CHC node with probability p_{re} to meet **Q1** and with probability p_{rc} to meet **Q2**. In that way, there will be $N_{chc} = (N/C) \cdot p_{re} \cdot p_{rc}$ CHC nodes in the range of R_{sc}. With limitation of spatial correlation range R_{sc}, the potential clusters number C meets $C \geq L^2/\pi R_{sc}^2$. Generally speaking, $p_{re} \approx 0.5$, and $0 < p_{rc} \leq 1$, then we have

$$N_{chc} = \frac{N}{C} \cdot p_{re} \cdot p_{rc} \leq \frac{\pi R_{sc}^2 N}{2L^2}. \quad (13)$$

According to the ranking updating rule of CHC node, we can obtain the maximal iterations of DSCC as

$$N_c = \lceil \log_2(N_{chc}) \rceil \leq \left\lceil \log_2 \frac{\pi R_{sc}^2 N}{2L^2} \right\rceil. \quad (14)$$

Simulation results in Fig.2 show that the real iterations of DSCC corresponds with the theoretical value estimated by Lemma 2. Besides, from Fig.2 we can learn that DSCC spends much less iterations for clustering than ASAP, which repeats the CH selection procedure until that all nodes are "covered".

In DSCC, we have set several timer to assist the clustering, i.e., T_{CHC}, T_{rank} and T_{CM}, all of which are used to guarantee each sensor node could obtain all potential messages from relative nodes, namely *C-msg*, *A-msg* or *J-msg* in different phases. A node could broadcast such kind of messages using CSMA MAC protocol, and the total time for each timer is decided mainly by the number of neighboring nodes in certain communication range. Assume time delay for one message between two nodes is κ. Then with reasonable coordination, a rough time for T_{CHC} would be $\frac{\kappa N \pi r^2}{L^2}$, and a rough time for T_{CM} and T_{rank} would be $\frac{\kappa N \pi R_{sc}{}^2}{L^2}$. The real timer setting in practice can be smaller, especially for T_{rank}. Notice that $T_{CH} = N_c \times T_{rank}$ and N_c could be estimated by Lemma 2.

V. Performance Evaluation

In this section, the performance of our proposed DSCC algorithm is evaluated by comparing it with three noteworthy clustering algorithms, namely ASAP [6], EEDC [7] and DClocal [9]. We choose them as compared algorithms because each of them is representative. For fairness, we have implemented a *single-hop* version of ASAP in our experimental simulations. With extensive simulations, we will compare the four algorithms on the following metrics which have been mentioned in subsection III-B: (a) number of clusters; (b) number of "forced" clusters which have only the CH node; (c) variance of cluster sizes σ_{cs} which is calculated as the standard deviation of cluster sizes using Eq.(15); (d) energy consumption for clustering; (e) average residual energy level of the selected CH nodes; (f) average dissimilarity of all clusters V_{dis} which is defined as Eq.(16) formally. In equations (15) and (16), C_i is assumed as the i-th cluster and C is the set of all clusters. We denote the CH node and a cluster members of cluster C_i as $C_i.ch$ and $C_i.cm$ respectively. Besides, $|C_*|$ denotes size of the set while $\overline{|C_*|}$ is the average cluster size.

$$\sigma_{cs} = \sqrt{\frac{\sum_{C_i \in C}(|C_i| - \overline{|C_*|})^2}{|C|}} \tag{15}$$

$$V_{dis} = \frac{\sum_{C_i \in C} \frac{\sum_{cm \in C_i} f_{md}(C_i.ch, C_i.cm)}{|C_i|-1}}{|C|} \tag{16}$$

A. Simulation Setup

In our simulations, N sensor nodes are randomly deployed in a $100m \times 100m$ sensing field in uniform distribution, *Sink* node is deployed $20m$ away from center of the right boundary, i.e., located at $(120m, 50m)$. Residual energy of each node is randomly chose in the range of $[4.0, 5.0]$ when clustering begins. Parameters of radio model adopt the same setting as those in LEACH [12] and they set as follows: $E_{elec} = 50nJ/bit$, $E_{fs} = 10pJ/bit/m^2$, $E_{mp} = 0.0013pJ/bit/m^4$, distance threshold d_o sets as $75m$ like the setting in HEED [13]. It is worth highlighting that it is the threshold do that decides the energy consumption when transmitting a message to a distant node. We employ similar method with [9] to generate a synthetic data set. Twenty-five event sources are fixed in the sensing field in uniform distribution, and the initial value of each event source is randomly chose in the range of $[20.0, 30.0]$. Value of event source at time t, say $e_w(t)$, follows normal distribution $N(e_w(0), 1)$ where $e_w(0)$ is the initial value of event source e_w. Sensor readings of node s_i are the comprehensive influencing results of all event sources, and the influence of an event source, say e_w, to node s_i is inversely proportional to the geographic distance between them.

We assume the size of data packet is $1000bits$ and the size of control packet is $100bits$. Cluster count factor f_c which decides the rough number of clusters in ASAP [6] sets as 0.05. Other simulation parameters for ASAP, EEDC and DClocal adopt the same settings in their corresponding papers. Lastly,

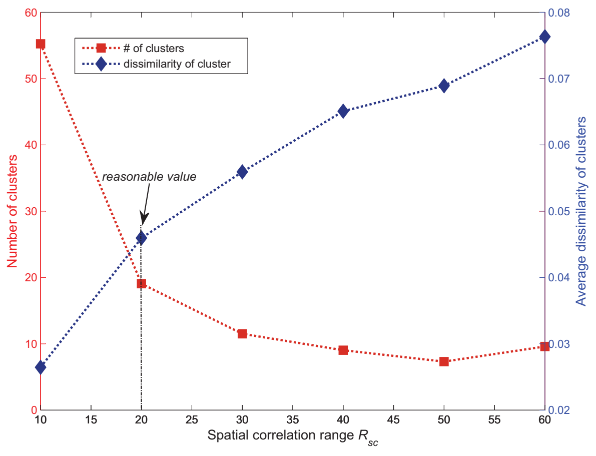


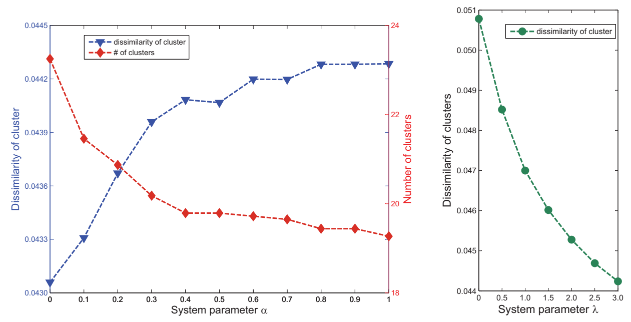
Fig. 3. Impact of R_{sc} on DSCC with simulation parameters $N = 300$, $L = 100m$, $\varepsilon = 0.5$, $\delta = 0.6$, $\alpha = 0.5$, $\lambda = 1.0$

notice that all results in this paper are the average values of 100 simulations.

B. Impact of system parameters

There are several system parameters involved in DSCC algorithm, namely R_{sc}, ε, δ, α and λ. Both R_{sc} and ε are the measurements of spatial correlation and are both application-specific, and we will discuss the impact of error-tolerance threshold ε on clustering in subsection V-D with details. As R_{sc} is aimed to be an assistant parameter to accelerate the clustering, it influences the clustering performance of all the four algorithms in the same way. Therefore, it makes sense for us to perform simulations on R_{sc} only with DSCC to study the relation between R_{sc} and spatial clustering. From Fig.3, we learn that greater R_{sc} will lead to fewer clusters while a larger average dissimilarity of clusters. Obviously, a greater spatial correlation range allows more potential nodes to be grouped into the same cluster. However, the accompanying result is that the data dissimilarity among cluster members becomes greater. As indicated in Fig.3, $R_{sc} = 20m$ could be a reasonable value to obtain compromised cluster number and average dissimilarity under the assumption of data and spatial correlation in our simulation setting. Hence, We set the default value of spatial correlation range R_{sc} as $20m$ in all following simulations.

Regarding to other parameters, parameter δ is used to control the quantity of potential CHCs and is quite application-dependent. Lots of nodes will declare to be CHCs when a small δ is adopted. However, when δ is set too large, the number of nodes which are qualified to be CHCs drops a lot. As mentioned above, α in Eq.(8) is a parameter to weight the relative importance of C_r and S_r. Fig.4(a) shows that cluster dissimilarity increases and cluster number decreases against α. This is because when α adopts a small value, more nodes with high similarity difference rate will become CHCs, and the final clusters will own much lower dissimilarity. Similar reasons for a high value of α, more nodes with strong "covering"



(a) parameter α (b) parameter λ

Fig. 4. Impact of system parameters (α and λ) on DSCC clustering

capability will become CHCs and fewer clusters are needed to "cover" all nodes. Parameter λ influences the attracting scores of CHs in $list_{CH}$ of each node. A high value means that nodes pay more attention on the data similarity, resulting in smaller dissimilarity among nodes in the same cluster. Impact of λ on cluster dissimilarity is showed in Fig.4(b). Note that λ will not affect the number of final clusters. Simulation results of Fig.4 are obtained with $N = 300$, $\varepsilon = 0.6$, $R_{sc} = 20$, $\delta = 0.7$, and $\lambda = 1.0$ for Fig.4(a), $\alpha = 0.5$ for Fig.4(b) respectively. In the following simulations, we fix $\alpha = 0.5$, $\lambda = 1.0$.

C. Impact of network density

In this subsection, we evaluate the clustering performance by varying the number of sensor nodes N from 100 to 600 to enlarge the network density, and we set $\varepsilon = 0.5$, $\delta = 0.6$. With the increase of network density, the number of clusters formed by the four algorithms increases correspondingly. From Fig.5(a) , we can learn that our proposed DSCC algorithm generates the fewest clusters, which facilitates approximate data collection as more nodes can turn into sleeping mode. As ASAP iterates until all nodes can be "covered", there will be few "forced" clusters. From Fig.5(b), we can learn that DSCC still performs the best except ASAP on the metric of number of "forced" clusters. Just like what we explained before, in the context of approximate data collection, equal cluster size guarantees that each node undertakes approximate the same load of data sampling. Fig.5(c) describes that though the variance of cluster sizes of DSCC is little greater than cluster variances of EEDC and ASAP, DSCC still performs much better than DClocal. The variance gaps among DSCC, EEDC and ASAP can be reasonably explained as ASAP and EEDC produce much more clusters than DSCC. Obviously, more energy is needed if more nodes are involved in clustering operation. In Fig. 5(d), energy consumption grows rapidly against network density except EEDC which performs clustering operation mainly at the powerful base station. Owing to choosing a small number of data series of each node for similarity measure, EEDC consumes the least energy. However, when more data is needed in similarity measure, EEDC will use up the most energy without doubt. Compared with the other two distributed algorithms, DSCC consumes less energy. As dissimilarity of cluster is mostly affected by error-tolerance threshold ε and the results on metric of CHs's average residual energy level are quite similar with Fig.6(d), i.e., CHs selected by DSCC have the most residual energy among the four algorithms, the simulation results of these two metrics are omitted in this subsection.

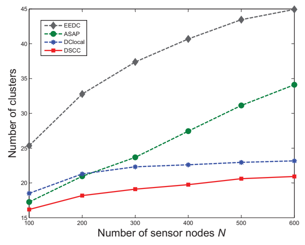
D. Impact of error-tolerance threshold

Besides spatial correlation range R_{sc}, error-tolerance threshold ε is the other critical factor to restrain the spatial clustering. In this subsection, we fix the number of sensor nodes N at 300, and explore the relation between clustering performance and ε by varying its value from 0.2 to 1.2. Notice that the value of pre-defined threshold δ which is used to control the quantity of CHCs is adjusted among $[0.50, 0.95]$ in accordance with the increase of ε. From Fig.6(a) and Fig.6(b), we learn that both cluster number and "forced" cluster number decrease as ε increases. For the restriction of spatial correlation loosens, more nodes can be grouped in the same cluster. As a result, fewer clusters are needed to "cover" all sensor nodes, and the number of "forced" clusters decreases at the same time. DSCC can group the most similar nodes into the same cluster and minimize the dissimilarity of cluster to the best. Just like what reported by Fig.6(c), DSCC owns the minimal dissimilarity of clusters among the four compared algorithms. Minimal dissimilarity among cluster members makes a cluster more steady to withstand the possible change of spatial correlation. In theory, the energy gain of approximate data collection come from the collection of less accurate data. Except EEDC, other three algorithms have paid attention to select nodes with more residual energy as cluster heads which bear more load than ordinary nodes. Fig.6(d) shows that cluster heads elected by DSCC have the most residual energy, and DClocal takes the second place. With more residual energy, node could work as cluster head for a much longer time and undertake more load. The same reason as before, we omit results of variance of cluster size and energy consumption for clustering here. Obviously, these two metrics are more associated with N.

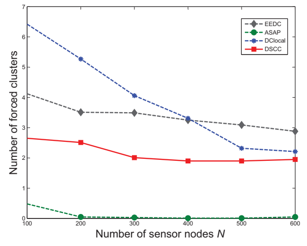
VI. Conclusions

In this paper, we propose the DSCC clustering algorithm to support approximate data collection by grouping sensor nodes with similar readings into the same cluster. By exchanging sensor nodes information in local region limited by the spatial correlation range R_{sc}, capable nodes are selected to be CHC nodes based on the qualifications of relative energy level and representative capability. Ranking strategy is proposed to accelerate the progress of CHs selection, and finally DSCC can terminate in a small number of iterations. Compared with previous work, DSCC not only pursues to generate as few clusters as possible, but also pays more attention to the data similarity between nodes to produce more data and spatial correlated cluster members.

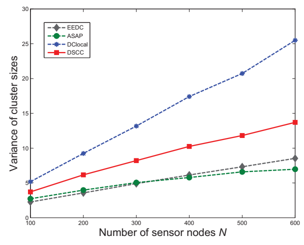
Extensive simulations are performed to evaluate DSCC algorithm. Compared with three noteworthy clustering algorithms, namely EEDC, ASAP and DClocal, simulation results



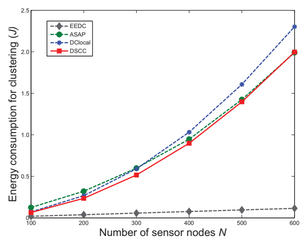
(a) Number of clusters



(b) Number of “forced” clusters

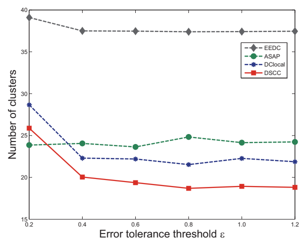


(c) Variance of cluster sizes σ_{cs}

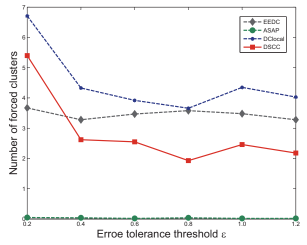


(d) Energy consumption

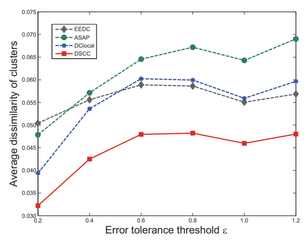
Fig. 5. Clustering performance with various network density



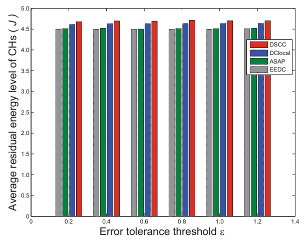
(a) Number of clusters



(b) Number of “forced” clusters



(c) Dissimilarity of clusters V_{dis}



(d) Average residual energy of CHs

Fig. 6. Clustering performance with various error-tolerance threshold ε

show that DSCC produces the fewest and the steadiest clusters which have the smallest dissimilarity. In addition, DSCC selects CHs with more residual energy and consumes less energy for clustering when compared with other distributed algorithms.

Acknowledgment

This work is partially supported by Scientific and Technical Innovation Team Project of Zhejiang Province for Digital Culture and Multimedia Technology (2010R50040).

References

[1] P. Corke, T. Wark, R. Jurdak, H. Wen, P. Valencia, and D. Moore, “Environmental Wireless Sensor Networks,” *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1903–1917, 2010.

[2] C. Wang, H. Ma, Y. He, and S. Xiong, “Adaptive Approximate Data Collection for Wireless Sensor Networks,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, no. 6, pp. 1004–1016, 2012.

[3] J. Li and S. Cheng, “(ϵ,δ)-Approximate Aggregation Algorithms in Dynamic Sensor Networks,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, no. 3, pp. 385–396, 2012.

[4] D. Chu, A. Deshpande, J. Hellerstein, and W. Hong, “Approximate Data Collection in Sensor Networks Using Probabilistic Models,” in *Proceedings of the 22nd International Conference on Data Engineering (ICDE)*, 2006.

[5] D. Tulone and S. Madden, “PAQ: Time Series Forecasting for Approximate Query Answering in Sensor Networks,” in *European Conference on Wireless Sensor Networks (EWSN)*, 2006.

[6] B. Gedik, L. Liu, and P. Yu, “ASAP: An Adaptive Sampling Approach to Data Collection in Sensor Networks,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 18, no. 12, pp. 1766–1783, 2007.

[7] L. Chong, W. Kui, and P. Jian, “An Energy-Efficient Data Collection Framework for Wireless Sensor Networks by Exploiting Spatiotemporal Correlation,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 18, no. 7, pp. 1010–1023, 2007.

[8] C. C. Hung, W. C. Peng, and W. C. Lee, “Exploiting Spatial and Data Correlations for Approximate Data Collection in Wireless Sensor Networks,” in *Proceedings of the Second International Workshop on Knowledge Discovery from Sensor Data (Sensor-KDD)*, 2008.

[9] C. C. Hung, W. C. Peng, and W. C. Lee, “Energy-Aware Set-Covering Approaches for Approximate Data Collection in Wireless Sensor Networks,” *Knowledge and Data Engineering, IEEE Transactions on*, 2011.

[10] A. Meka and A. Singh, “Distributed Spatial Clustering in Sensor Networks,” in *International Conference on Extending Database Technology (EDBT)*, 2006.

[11] A. Abbasi and M. Younis, “A Survey on Clustering Algorithms for Wireless Sensor Networks,” *Computer Communications*, vol. 30, no. 14-15, pp. 2826–2841, 2007.

[12] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “An Application-Specific Protocol Architecture for Wireless Microsensor Networks,” *Wireless Communications, IEEE Transactions on*, vol. 1, no. 4, pp. 660–670, 2002.

[13] O. Younis and S. Fahmy, “HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks,” *Mobile Computing, IEEE Transactions on*, vol. 3, no. 4, pp. 366–379, 2004.

[14] S. Yoon and C. Shahabi, “The Clustered AGgregation (CAG) Technique Leveraging Spatial and Temporal Correlations in Wireless Sensor Networks,” *ACM Transactions on Sensor Networks*, vol. 3, no. 1, pp. 1–39, 2007.

[15] L. A. Villas, A. Boukerche, H. A. de Oliveira, R. B. de Araujo, and A. A. Loureiro, “A Spatial Correlation Aware Algorithm to Perform Efficient Data Collection in Wireless Sensor Networks,” *Ad Hoc Networks*, 2011.

[16] T. Le, N. Pham, and H. Choo, “Towards A Distributed Clustering Scheme based on Spatial Correlation in WSNs,” in *International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2008.

[17] H. Long, Y. Liu, X. Fan, R. Dick, and H. Yang, “Energy-Efficient Spatially-Adaptive Clustering and Routing in Wireless Sensor Networks,” in *Design, Automation & Test in Europe Conference & Exhibition*, 2009.

[18] F. Sivrikaya and B. Yener, “Time Synchronization in Sensor Networks: A Survey,” *IEEE Network*, vol. 18, no. 4, pp. 45–50, 2004.

[19] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, “System Architecture Directions for Networked Sensors,” *ACM SIGPLAN Notices*, vol. 35, no. 11, pp. 93–104, 2000.

[20] L. Chong, W. Kui, and P. Jian, “A Dynamic Clustering and Scheduling Approach to Energy Saving in Data Collection from Wireless Sensor Networks,” in *IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, 2005.