EDA: Event-oriented Data Aggregation in Sensor Networks

Ying Guo, Feng Hong, Zhongwen Guo, Zongke Jin, Yuan Feng
Department of Computer Science and Engineering, Ocean University of China
Email: {guoying, hongfeng, guozhw, jinzongke, fengyuan}@ouc.edu.cn

Abstract—Data aggregation is a crucial technique for energy constrained sensor networks. Previous researches on data aggregation are featured as query-oriented, which only provide partial information of the event happened in the deployment area at the base station. In this paper, we propose an event-oriented data aggregation approach, called EDA. EDA presents the distributed algorithm by exploiting Cloud Membership model of fuzzy logic to aggregate the information of events in the sensor networks. It also presents a distributed algorithm to collect and aggregate the event information. The base station will restore the whole event information when receiving the aggregated packets of event features. EDA can balance the tradeoff between delay, traffic savings, and precision of the restored events. The performance has been evaluated through both theoretical analysis and simulations. We also confirm the performance with the traces of our offshore sensor network testbed (OceanSense).

Keywords-event-oriented; data aggregation; cloud membership model; wireless sensor network

I. INTRODUCTION

Sensor networks are deployed to monitor or detect the critical events in the physical environments. Traditionally, information sampled at the sensor nodes needs to be delivered to the central base station through the network interconnection for assembling and further analysis. This approach, however, is wasteful since it results in excessive communication. Wireless data transmission costs hundreds to thousands of times more energy than performing local computation on the same data. Another important issue is that individual sensor readings are inherently unreliable. In order to address these problems, sensor networks exploit in-network data aggregation to reduce communication cost and improve reliability [1].

Data aggregation is defined as the process of aggregating the sampled data from multiple sensors to provide fused information to the base station along the multi-hop route in sensor networks [2]. Researches on data aggregation featured as query-oriented, which support database queries like MAX, AVG and MEDIAN [3], [4], [5], [6], [7], [8]. According to the query functions, the intermediate sensor node can compute the partial aggregation with its own value and the values of downstream nodes before reporting to its upstream node. Moreover, some researches focus on supporting more complex queries, like contours and histograms [9], [10], [12], [11].

However, one important factor has been left out of consideration in previous data aggregation approaches, which is the logic relationship of the sampled values created by the same physical event. If all the information of one event observed by sensor nodes can be aggregated to some significant features, the whole event can be restored at the base station, when the packet containing these features is delivered back from the node who performs the aggregation. This kind of data aggregation will greatly eliminate the amount of data transmission and we call it event-oriented data aggregation. Event-oriented data aggregation brings out a lot of new research challenges. For example, we need to find out which features should be summarized by aggregation in order to restore the whole event information at the base station. We also need to balance the tradeoff between the traffic savings, the precision and the delay of the restored event. In addition, we aim to avoid heavy computational overhead at one sensor node for resource constrained sensor devices.

This work is motivated by one of our ongoing research projects, OceanSense [13]. As shown in Fig. 1, a number of restricted floating sensors are deployed [14], usually tens of meters away from each other (sparsely deployed), which can acquire and analyze information about environment factors such as temperature, light illuminance, sea depth. Because all the nodes are floating offshore, it is very inconvenient to change the batteries for the dead nodes. Therefore, some data aggregation technique is specially needed to prolong the network lifetime.

Figure 1. OceanSense Project: The upper left photo shows a floating sensor. The upper right figure is the aircscape of the deployment scope. The figure at the bottom is a field photo of 20 floating sensors, which are labeled from 1 to 20.
In this paper, we propose the Event-oriented Data Aggregation (EDA) technique, which is a distributed algorithm by exploiting Cloud Membership model of fuzzy logic to aggregate the information of events in sensor networks. EDA can be used in sensor networks for the target of event detection. Major contributions of this paper are as follows:

- We design a novel data aggregation technique, EDA, which exploits the Cloud Membership model to describe the logical relationship of the sampled values created by the same physical event. The whole event information can be restored at the base station when the aggregated packets of event features are transmitted back.
- EDA presents the distributed data aggregation algorithm, which can balance the tradeoff delay, traffic savings and precision of the restored event. Meanwhile, this distributed algorithm can handle the situations of multiple events happening nearby and the event information of none Normal distribution.
- The performance of EDA is carefully examined through theoretical analysis and simulations. We also confirm its performance with the trace data of OceanSense.

The rest of the paper is organized as follows. Section II introduces Cloud Membership model of fuzzy logic. Section III presents the design details of EDA with the theoretical analysis of its performance. Section IV describes the simulation results. Section V illustrates the performance with the trace data of OceanSense. We discuss the conclusions and the future work in Section VI.

II. CLOUD MEMBERSHIP MODEL

The sampled values of the sensor nodes are of some logical relationship when they are created by the same physical event. These sampled data can be aggregated to some important features of the event, if the relationship between them is exploited. Cloud Membership Model (CM-model) of fuzzy logic contains the uncertain transform between qualitative concepts and their quantitative expressions. So we employ CM-model to represent the relationship of the sampled values created by the same event and to aggregate and restore event information.

We introduce Cloud Membership Model (CM-model) of fuzzy logic in this section. CM-model describes the fuzzy concept only with three parameters: center ($x_0$), width ($b$) and thickness ($\sigma$). CM-model is defined as following [15], [16]:

**Definition 1: Membership Degree and Membership Function** Let $U$ be a sample set, called the universe of discourse. $A$ is a fuzzy subset of $U$, representing some qualitative concept valued on $U$. $\forall x \in U$, give a value $\mu(x)$ ($\mu(x) \in [0, 1]$) corresponding to $x$, called the membership degree of $x$ according to the concept $A$. The mapping of $f : x \rightarrow \mu(x)$ is called the membership function of the concept $A$.

<table>
<thead>
<tr>
<th>Age</th>
<th>Membership Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td>5</td>
<td>0.0</td>
</tr>
<tr>
<td>10</td>
<td>0.0</td>
</tr>
<tr>
<td>15</td>
<td>0.0</td>
</tr>
<tr>
<td>20</td>
<td>0.0</td>
</tr>
<tr>
<td>25</td>
<td>0.0</td>
</tr>
<tr>
<td>30</td>
<td>0.0</td>
</tr>
<tr>
<td>35</td>
<td>0.0</td>
</tr>
<tr>
<td>40</td>
<td>0.0</td>
</tr>
<tr>
<td>45</td>
<td>0.0</td>
</tr>
<tr>
<td>50</td>
<td>0.0</td>
</tr>
<tr>
<td>55</td>
<td>0.0</td>
</tr>
<tr>
<td>60</td>
<td>0.0</td>
</tr>
<tr>
<td>65</td>
<td>0.0</td>
</tr>
<tr>
<td>70</td>
<td>0.0</td>
</tr>
<tr>
<td>75</td>
<td>0.0</td>
</tr>
<tr>
<td>80</td>
<td>0.0</td>
</tr>
<tr>
<td>85</td>
<td>0.0</td>
</tr>
<tr>
<td>90</td>
<td>0.0</td>
</tr>
<tr>
<td>95</td>
<td>0.0</td>
</tr>
<tr>
<td>100</td>
<td>0.0</td>
</tr>
</tbody>
</table>

![Figure 2](https://via.placeholder.com/150)

**Figure 2.** Example on qualitative concept of “young people about twenty” (a) constructed by CM-model of $N^3(20, 0.7, 0.06)$ (b) membership degree curve from real census of questionnaires.

**Definition 2: CM-model, Cloud Drop and Cloud** If $\forall x \in U$, the membership degree $\mu(x)$ is of some stable tendency, the distribution of $\mu(x)$ on $U$ is called CM-model and $(x, \mu(x))$ is called a Cloud Drop, where the stable tendency satisfying $N^3(x_0, b, \sigma)$. All Cloud Drops compose the Cloud.

In order to illustrate CM-model of $N^3(x_0, b, \sigma)$ clearly, we describe an example of the qualitative concept of “young people about twenty”, shorted as YOUNG(20). For CM-model of YOUNG(20), the discourse universe is assigned as the person of age $x$, shorted as $x$. Therefore, $x = 20$ is the point in the discourse universe that can represent YOUNG(20) most properly, labeled as $x_0$. And $\mu(x_0) = 1$.

**Definition 3: Center** $x_0$ is defined as the point in the universe of discourse that can represent the qualitative concept most properly.

Moreover, CM-model assumes that all the examples in the universe belonging to some concepts satisfy Normal distribution of $N(x_0, b^2)$, i.e. $b$ represents the width of the Cloud. As shown in Fig. 2(a), the samples of $(-\infty, x_0 - 3b)$ and $(x_0 + 3b, +\infty)$ will be rarely related to the concept, so $\mu(x_0 + 3b) = 0$.

**Definition 4: Width** ($b$) is the uncertainty measurement of the qualitative concept.

$b$ is the measure of the fuzziness of the concept over the universe of discourse, showing how many elements can be accepted by the concept. $b$ points out the granularity of the concept. Larger $b$ is, higher abstraction the concept is.

Meanwhile, CM-model assumes that the expectation curve of the membership degree ($\mu(x)$) satisfies the following equation:

$$\mu(x) = e^{-\frac{(x-x_0)^2}{2b^2}}.$$ 

**Definition 5: Thickness** ($\sigma$) is the uncertain degree of $b$. CM-model exploits $\sigma$ to describe the randomness of the Cloud Drop $(x, \mu(x))$ to $\mu(x)$. CM-model assumes that the deviation of $(\mu(x) - \mu(x))$ also satisfies Normal
distribution of $N(b, \sigma^2)$. Larger $\sigma$ is, more randomly the set of membership degrees distributed is.

So $N^+(x_0, b, \sigma)$ is composed of three Normal distributions, which are: (1) all samples in the universe belong to some concept satisfying Normal distribution of $N(x_0, b^2)$, (2) $\mu(x)$ satisfies the calculation function like the probability density function of Normal distribution, and (3) $(\mu(x) - \bar{\mu}(x))$ satisfies Normal distribution of $N(b, \sigma^2)$.

Through CM-model, we can generate the Cloud Drops if $(x_0, b, \sigma)$ is defined. Figure 2(a) is created with the Cloud of $N^+(20, 0.7, 0.26)$. A Cloud Drop can be created with the algorithm in Fig. 3, given $x$. The whole Cloud is generated when all drops are created, which is called Cloud Construction.

![Figure 3. Cloud Drop Creation.](image)

Conversely, we can summarize the three parameters of CM-model, if the membership degree curve can be obtained through some census. Figure 2(b) shows the membership degree curve of YOUNG(20) from real census of questionnaires [17]. $(x_0, b, \sigma)$ can be calculated through the statistical analysis process, which is shown in Fig. 4. This process is called Cloud Summarizing. Meanwhile, it also shows that CM-model is applicable when comparing Fig. 2(a) with Fig. 2(b).

![Figure 4. Cloud Summarizing.](image)

As the conclusion, CM-model realizes the uncertain transforms between qualitative concepts and their quantitative expressions, which forms the basis of EDA. It is still one question, however, how CM-model can be integrated in the in-network data aggregation process of sensor networks.

### III. DESIGN OF EDA

We focus on illustrating EDA design in one dimensional (1-D) sensor networks, in order to explain how the aggregation approach works clearly. The EDA design of 1-D sensor network can be directly extended to two dimensional (2-D) situations, so we skip the design in 2-D sensor networks for the page limit.

#### A. Mapping from Events to Clouds

CM-model can describe the fuzzy concept only with three Cloud parameters of center ($x_0$), width ($b$) and thickness ($\sigma$). If one node can aggregate all the data sampled by itself and other nodes in the event scope through Cloud Summarizing, and transmit these feature parameters back to the base station, the users can learn a lot of details of the event from Cloud Construction. This approach will definitely decrease the energy consumption of data delivering and fulfill the application target of event detection, which is the motivation of EDA.

To make the above process work, two questions should be answered, which are: (1) how to fill the gap between detected events of sensor networks and the fuzzy concepts, and (2) how to calculate the membership degree to apply CM-model. The answer to the first question is to map the real event to the fuzzy concept of “event scope”.

We illustrate this mapping through one common example of fire detection in sensor networks, which represents a lot of similar applications, like terrestrial heat, frost, radiation detection et al.

For fire detection, the most usual measurement is the temperature. The traditional approach of fire detection is: a sensor node will transmit an alarm packet containing its position and sampled temperature to the base station, if the temperature is higher than some predefined value. We defined this value as $t_b$ (temperature of fire border). Such nodes are called fired nodes (F-nodes). When all fire alarm packets are transmitted back to the base station, the user will know the situation of the fire through some queries or visualization results.

In order to apply CM-model, we exploit another predefined value $t_u$ (temperature of upper bound), which is used to represent the temperature near the fire center. This means that one sensor knows that it is at some place near the fire center, if its sampled value is higher than $t_u$, $t_u$ can be easily estimated through fire histories for specified environments. Such kind of nodes are called near fire center nodes (NFC-nodes).

Another parameter of $t_c$ is defined in the process of calculating the membership degree, which representing the highest temperature sampled among all NFC-nodes. The corresponding node is called the fire center node (FC-node). FC-node with $t_c$ is at the position which can represent this qualitative concept of “fire scope” most properly. FC-node does not need to be at the geometric center of the fire, which...
will be illustrated in the distributed algorithm and proven by simulation results.

If a F-node knows the value of \( t_c \), it can calculate its own membership degree \( \mu_x \) through Equation (1), where \( x \) represents the location of the sensor node with sampled temperature \( t_s \). This answers to the second question proposed in the beginning of this subsection.

\[
\mu_x = \frac{t_s - t_b}{t_c - t_b}. \tag{1}
\]

The membership degree of FC-node will be 100\% and the membership degree of F-node near the fire border will be just over 0. Assuming that the FC-node can get all the sampled values and corresponding locations of the fire, it can construct the membership degree curve through Equation (1). So it can aggregate the Cloud parameters \((x_0, b, \sigma)\) through Cloud Summarizing process and transmit such features back to the base station for event restoring.

**B. Distributed Aggregation Algorithm**

The mapping from the physical event to Cloud can be solved when the FC-node collects all the information from other sensor nodes in the fire scope. The FC-node has the value of \( t_c \), so it can perform the membership degree calculation through Equation (1), when it receives the sampled values from other nodes. Then it can run the Cloud Summarizing process and create the aggregated packets with the event features of \((x_0, b, \sigma)\), which will be transmitted back to the base station. However, there are two performance problems left for the FC-node to collect the event information and calculate the aggregation, which are:

- **The FC-node must wait a long time for collecting sampled data in the whole event scope, before sending out the alarm packet of event features. It is important to control the delay of the first alarm packet, which can not be supplied now.**
- **If the event covers too much area like the whole deployment, how can the FC-node finish the calculation of the Cloud Summarizing process with such amount of data? This indicates that the aggregation calculation should be limited to certain amount of data, for sensor nodes are resource-constrained.**

The key to solve the two problems, is to limit the scope of data aggregation for one FC-node. One physical event should be divided into several logical events for data aggregation. EDA uses the level parameter \( l \) to realize this division. EDA is composed of five phases as following:

**Phase 1: center determining.** This phase is to determine the fire centers of the physical events. This can be achieved by all NFC-nodes exchanging the packets of \((NFC_id, temperature)\). Such node will be the FC-node, if it finds that its sampled value is higher than all its NFC neighbors. Meanwhile, the FC-node will record all its NFC neighbors for the next phase.

**Phase 2: event notification.** After the first phase, the FC-node will broadcast its event notification packet to all its NFC neighbors. The event notification packet contains four items of \((E_id, FC_id, NFC_id, TTL)\). \( E_id \) is a unique identifier to represent the event. \( NFC_id \) is the node ID who broadcasting this packet. \( TTL \) is initialized with \( l \), which is used in level control of the logical events.

The event notification packet will be repeatedly broadcasted, with \( NFC_id \) changing to the current node ID and \( TTL = TTL - 1 \), until the receipt node is not a F-node or \( TTL = 0 \). All the F-nodes will record \( E_id \) and \( NFC_id \) of the first notification. It will label itself belonging to event \( E_id \) and record its father on the notification tree as \( NFC_id \). At the same time, it will inform its father of its presence, who will record that too. The F-node only deals with the first notification packet received and ignores the followings if there are any.

When a F-node receives a notification packet of \( TTL = 0 \), it knows that it is on the border of the current event \( E_id \), labeled as B-node. Meanwhile, it will decide to be the FC-node of a new logical event, and broadcast the new event notification packet with \( TTL = l \) again. Hence the FC-node no longer need to be of the sampled value higher than \( t_b \).

When the packet arrives the nodes on the border of the sensor network, it will also become the B-node of event \( E_id \). When the packet arrives the node whose sampled value is lower than \( t_b \), this node out of the border of the event will inform its father to be the B-node of event \( E_id \).

At the end of this phase, the tree of notification broadcasting will be built, with every node knowing its father and sons, except the FC-node as the root and the B-nodes as the leaves. This tree will be used in the data collection phase as the aggregation tree.

**Phase 3: data collection.** When one node is sure that it is a B-node, this phase begins. The B-node will transmit the data packet of (location, temperature) back to its father on the notification tree. After one node has received all the packets of its sons, it combines all the information in these packets with its own location and sampled value to form a new data packet. Some compress coding skills can be applied in this process [19]. After that, it will transmit this packet of combined data to its father. Therefore, all the data packets will be transmitted back to the FC-node, which contains sampled values and corresponding locations of event \( E_id \).

**Phase 4: data aggregation.** The FC-node will run the Cloud Summarizing process in Fig. 4 to calculate the Cloud parameters, when all its NFC neighbors send back their data packets. The FC-node will set \( t_l \) to the lowest sampled value from all the data received, which will replace \( t_b \) in Equation (1) in the process of membership degree computing. This aggregation process summarizes all the sample data and their locations to one alarm packet, which contains \((E_id, x_0, b, \sigma, t_c, t_l)\). When the alarm packet is created, the
FC-node will transmit it back to the base station.

**Phase 5: event restoring.** The base station can directly restore one event, when it receives an alarm packet. This is achieved by exploiting Cloud Construction process to create the membership degree at any location. Furthermore, Equation (1) can be used reversely to calculate temperature value from the membership degree. Therefore the event restored can fulfill the user’s need, for it provides most information about the event and can answer any query submitted.

When the base station receives several alarm packets with their locations of FC-nodes close to each other, the base station combines the restored information of different events under such order as from center to border. If two event centers cannot be decided which is the center, EDA adopts the rule of high values covering low values of restored events, because high values are often more important than low ones.

For example, one fire happens around Node 8 in Fig. 5(a), where $t_b = 60^\circ C$ and $t_u = 180^\circ C$ are labeled. The base station is set at $x = 0$. The node ID and its location are same in Fig. 5(a), shown by its x-axis value. The temperature sampled by the sensor node is described as the y-axis value. For the temperature sampled at Node 7, 8, 9 are higher than $t_u$, so they become NFC-nodes. They exchange the sampled temperatures, and Node 8 knows that it is the fire center. It will label $E_{id}$ with some unique string, such as $E_{id} = e$. This indicates the end of center determining phase.

If the level parameter is set to 7, Node 8 will broadcast the notification packet $(e, 8, 7)$ to Node 7 and Node 9, which will further broadcast the notification packet of $(e, 8, 7, 6)$ and $(e, 8, 9, 6)$ independently. The broadcast process repeats and every F-node will learn its father and sons on the notification tree, until the notification packet arrives node 1 or node 15. Node 1 will inform Node 2 to be the B-node of event $e$, which is the same for Node 14 being the B-node. Event notification phase ends here.

Node 2 and Node 14 will send the data packet back to Node 3 and Node 13 independently. And Node 3 will send the data packet of $(2, t_2, 3, t_3)$ back to Node 4. Eventually, Node 8 will get $(2, t_2, 3, t_3, \ldots , 7, t_7)$ from Node 7 and $(14, t_{14}, 13, t_{13}, \ldots , 9, t_9)$ from Node 9, which is the end of data collection phase.

With all these data, Cloud parameters of $(8, 3.8944, 0.7412)$ can be calculated through Cloud Summarizing process on Node 8. And the alarm packet of $(e, 8, 3.8944, 0.7412, 198.9, 61.4)$ will be transmitted back to the base station, which will restore the event information. These operations form the data aggregation and the event restoring phase. The red points show the temperature curve restored at the base station in Fig. 5(a).

If set $l = 3$ in the above example, three alarm packets will be transmitted back to the base station, which are $(e_l, 5, 1, 1.291, 0.4926, 148.5, 61.4)$, $(e_8, 8, 2.1602, 0.4866, 198.9, 147)$ and $(er, 11, 1.291, 0.4924, 147, 62.7)$. According to the construction order, the base station will create the event information with $(8, 2.1602, 0.4866, 198.9, 147)$ first in (5, 11), which is described as red points in Fig. 5(b). As the other two Clouds are at the same distance to the fire center, the construction order is not important between them. One event information will be constructed with $(e_l, 5, 1, 1.291, 0.4926, 148.5, 61.4)$ in [1, 5], and the other with $(er, 11, 1.291, 0.4924, 147, 62.7)$ in [11, 15]. As the restored temperatures are lower than $t_b$ on Node 1 and Node 15, so the restored values of these two events are limited in [2, 5] and [11, 14], which are described as black and green points in Fig. 5(b).

**C. Performance Analysis**

Assuming that the energy consumption of communication costs one unit per packet per hop and the packet delay is the same for one hop, we compare EDA’s performance with the traditional approach that all packets are transmitted back to the base station. Three parameters are evaluated, defined as following:

**Delay Rate (DR):** Delay is defined to evaluate the delay of the alarm packet from the center of the physical event to the base station, for this packet contains the most important information of the event. $DR$ is calculated as the ratio of the delay of the alarm packet of EDA to the traditional approach.

**Communication Cost Rate (CR):** $CR$ is defined as the ratio of energy consumption on communication of EDA to the tradition approach. We don’t count the computing energy consumption, for it is very low comparing to the communication cost.

**Restored Error (RE):** $RE$ is defined as the average difference between the restored values to the original sampled values per node, which can evaluate the precision of EDA.

The delay of the alarm packet of the traditional approach from the fire center to the base station is proportional to the hop distance between them, labeled as $d$. The delay of the alarm packet of the fire center under EDA contains two parts: the transmission delay from the fire center to the base station and the time consumed in event notification and data
collection phase, which is \( d + 2l \). So
\[
DR = \frac{d + 2l}{d} = 1 + \frac{2l}{d} \tag{2}
\]

The communication cost in traditional approach is for all nodes in the fire scope to transmit their alarm packets back to the base station. Assuming that the base station is deployed at the border of the sensor network and the nodes are distributed evenly, the communication cost will be \((2r + 1)d\), when the event scope is of radius \( r \). For EDA, the communication cost contains two parts: the packet transmission for collecting the sampled data and the delivering of the alarm packets from FC-Nodes to the base station, which is \( 4r + (2r/l − 1) \). So
\[
CR = \frac{4r + (2r/l − 1)d}{(2r + 1)d} = \frac{4r}{(2r + 1)d} + \frac{(2r − 1)}{(2r + 1)} \tag{3}
\]

As \( r << d \) and \( l < r \), so \( CR \) will be less than one, which illustrates the traffic savings of EDA. Meanwhile, Equation (3) shows that \( CR \) is reversely proportional to \( l \).

\( RE \) is directly limited by CM-model, which is proportional to \( \sigma \times (te – tl) \). If \( l \) gets larger, \( \sigma \) will be lower, for less nodes are put into the calculation process of \( \sigma \). And \( te – tl \) will be lower too, for the two nodes sampled them will be nearer limited by lower \( l \). So \( RE \) is directly proportional to \( l \), which shows the precision of EDA is reversely proportional to \( l \).

The above analysis shows that the level parameter is an important parameter for EDA. The delay rate is directly proportional to \( l \). The communication cost and precision are reversely proportional to \( l \). So \( l \) should be carefully chosen for real deployment, which can balance the delay, traffic savings and precision for EDA.

For the example in Fig. 5, the performance of different level parameters are comparing in Table I, if the base station is put at \( x = 0 \). It shows that the precision of \( l = 3 \) is 30\% higher than \( l = 7 \). And the delay rate of \( l = 3 \) is 46\% shorter than \( l = 7 \), while the communication cost of \( l = 3 \) is about 50\% higher than \( l = 7 \). The impact of \( l \) to EDA’s performance is further demonstrated in the following simulations.

**IV. SIMULATIONS**

In this section, we first show three simulation results in 1-D sensor networks. The first result demonstrates the impact of level parameter to the performance. The second result shows the performance when multi-events happening nearby. The third result describes the performance when the distribution of the sampled values of the event falls out of the Normal distribution. After that, we give one simulation of two events happening nearby in 2-D space. \( t_b = 60^\circ C \) and \( t_u = 180^\circ C \) are set for all the following simulations.

For the first simulation of 1-D sensor network, 100 nodes are distributed evenly in the deployment area. We evaluate the impact of level parameter to the performance. The values of original event and restored event with \( l = 6, 12, 18 \) are comparing in Fig. 6(a). It shows that the restored values fluctuating harder when \( l \) getting larger. The cumulative distribution function of the restored errors is shown in Fig. 6(b), which shows less \( l \), more precise the restored results. The results of delay rate, communication cost and precision are comparing in Table II, which is consistent with the theoretical analysis. So there exits the tradeoff between delay, traffic savings and precision, when determining the value of the level parameter in advance for specific application.

For the second and third simulation of 1-D sensor network, 200 nodes are distributed evenly in the deployment area. Figure 7 shows that two close events happen nearby in the deployment area. The red line illustrates the re-

![Figure 6. Impact on performance of different level parameter](image)

![Figure 7. Performance on two events happening nearby](image)

**Table I**

<table>
<thead>
<tr>
<th>level</th>
<th>( DR(%) )</th>
<th>( CR(%) )</th>
<th>( RE(\circ C) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( l = 3 )</td>
<td>186</td>
<td>40</td>
<td>3.2934</td>
</tr>
<tr>
<td>( l = 7 )</td>
<td>211</td>
<td>26.6</td>
<td>7.5732</td>
</tr>
</tbody>
</table>

**Table II**

<table>
<thead>
<tr>
<th>level</th>
<th>( l = 6 )</th>
<th>( l = 12 )</th>
<th>( l = 18 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( DR(%) )</td>
<td>124</td>
<td>148</td>
<td>172</td>
</tr>
<tr>
<td>( CR(%) )</td>
<td>24.4</td>
<td>14.4</td>
<td>10.0</td>
</tr>
<tr>
<td>( RE(\circ C) )</td>
<td>4.93</td>
<td>8.31</td>
<td>9.89</td>
</tr>
</tbody>
</table>
stored events. The restored error is only 5.6°C and the communication cost rate is 25.5%. Because EDA applied the distributed data aggregation algorithm, it can handle multi-events nearby correctly. Even there only happens one physical event, EDA will handle it as several logical events whose scopes are limited by the level parameter. This simulation result also shows that the event center do not need to be at the geometric center of the deployment area.

Another simulation is conducted in 1-D sensor network according to different distributions of event information, which are one event of triangle-like distribution and one of semicircle-like distribution. The original and restored events of these two simulation results are shown in Fig. 8. The performance of precision and communication cost are comparing in Table III. It shows that EDA can well suit the above two distributions of event information. As the CM-model assumes that the value distribution of events is Normal distribution, someone may wonder why EDA can still work in above two cases. This is because EDA adopts the distributed data aggregation limited by the level parameter. Therefore, this simulation proves that EDA can keep its performance under none Normal distribution of event information.

For 2-D simulation, 3500 nodes are deployed in the grid of 50 × 70. The original distributions of two close events are shown as the color map in Fig. 9(a). Both the distributions of the two events are created through Normal distributions with Gauss noises. The restored events are shown in Fig. 9(b). Comparing Fig. 9(a) and Fig. 9(b), it can be concluded that the restored events keep the form of the original events. $RE = 7.91°C$ and $CR = 17.54\%$ are calculated from the result. This confirm the feasibility of EDA in 2-D sensor networks.

### Table III

| Impact of different original distribution of event information on precision and communication cost ($l = 6$) |
|---|---|---|
| Event Distribution | Triangle-like | Semicircle-like |
| $RE(°C)$ | 5.2 | 1.1 |
| $CR(\%)$ | 24.9 | 22.8 |

V. EXPERIMENT ON DATA TRACE OF OCEANSENSE

Our offshore wireless sensor network testbed contains about twenty TelosB motes floating over the sea surface. A TelosB mote is encapsulated in the water-proof bottle and mounted on the top of the floating pole over the sea surface. Every pole is equipped with lightweight supporting equipment so that it can raise the TelosB mote 1.5 meter over the sea surface.

Our floating sensor node is carrying one Lithium cell of 8.5Ah, which can support its monitoring work continuously for about twenty days. However, this is not enough for the long-term ocean surveillance target of OceanSense. In the past year, we kept sailing out and changing the batteries for the sensor nodes every twenty days.

Before deploying EDA on the sensor nodes offshore, we conduct EDA on the traces of OceanSense in order to prove its functionality for future deployment. All the traces of OceanSense are published on our website [13].

We conduct our experiment of EDA on the trace data around 15:00 on April 14th, 2008, when 15 nodes are operating properly. The experiment results on other time’s trace data shows the same result, which is skipped for limit of spaces. The location of each sensor node is obtained through PI approach [18] and labeled on Fig. 10(a). The deployment area covers about 400 × 200m$^2$. The level parameter is set to three. We analyze the monitored data of light illuminance for event detection.

The monitored light intensities are shown in Fig. 10(b). $t_a$ is set as 1340CD and $t_b$ is set as 1100CD and there are four nodes whose sampled values are higher than 1340CD, which are Node 3, 8, 10 and 14. Node 8 and Node 10 are one-hop neighbor between each other, so Node 8 will be the event center for its sampled value is higher than Node 10’s. And Node 3 and Node 14 will be another two event centers, for they do not have NFC neighbors. Therefore, the restored information are calculated from three Clouds of $(\epsilon, (3, 80, 90), 6.3471, 4.3205, 3.2192, 1.7625, 1362, 1227)$, $(\epsilon, (240, 90), 2.1679, 2.1679, 4.9943, 2.7695, 1346, 1254)$ and $(\epsilon, (360, 170), 2.8868, 1.5275, 3.8391, 0.6671, 1331, 1266)$. The feature packet for 2-D sensor networks contains $(E_{id}, (x_0, y_0), b_x, b_y, \sigma_x, \sigma_y, t_c, t_l)$.
VI. CONCLUSION AND FUTURE WORK

Data aggregation is a crucial technique for energy constrained sensor networks. Previous research on data aggregation can be featured as query-oriented, however, they can not restore all the event information at the base station. This paper proposes an event oriented data aggregation approach, which aims to restore the whole event with the aggregated packets of event features.

EDA makes use of Cloud Membership model to describe the event features for aggregation. It also presents the distributed algorithm to collect and aggregate the event information. The base station will restore the event information upon event feature packets. EDA can balance the tradeoff between delay, traffic savings, and precision of the restored events. The performance has been confirmed through theoretical analysis and simulations. The simulations also prove that EDA can work for the cases of multiple events happening nearby and the event information of none Normal distributions. The feasibility of EDA is further proven by the traces of our offshore sensor network testbed (OceanSense).

This work is supported by NSF China under grant No. 60703082, 60933011 and 60873248, the National Basic Research Program of China (973 Program) under grant No. 2006CB303000, the National High Technology Research and Development Program of China (863 Program) under grant No. 2006AA09Z113.

ACKNOWLEDGMENT

REFERENCES


