2011 Eighth IEEE International Conference on Mobile Ad-Hoc and Sensor Systems

CCD: Locating Event in Wireless Sensor Network without Locations

[1]Shuo Lian, [2]Yuan He, [1]Jizhong Zhao

[1]Department of Computer Science and Technology
Xi'an Jiaotong University
lian.shuo@stu.xjtu.edu.cn, zjz@mail.xjtu.edu.cn

[2]TNLIST, School of Software, Tsinghua University
CSE Department, Hong Kong University of Sci. & Tech.
heyuan@cse.ust.hk

ABSTRACT

Event detection is a typical application of wireless sensor networks. The existing approaches of event detection usually employ certain event models that are constructed with prior domain knowledge. The resulting event detection processes appear to be cost-inefficient, which require either intensive data exchanges among neighboring nodes or caching large columns of history data. In this paper, we focus on the issue of locating event in wireless sensor network without locations. This involves two tasks, namely detecting an event and identifying an area in the network where the event occurs. Motivated by the real system, we propose a model-free approach for event detection called CCD (Coding Cost based event Detection). Coding cost is a metric that quantifies the diversity of a set of sensor readings. Incorporated into the inherent data collection mechanism, CCD passively constructs a gradient map of coding cost throughout the network. An event is then detected where a change point of gradient appears and identifies the event pattern automatically. CCD is fully distributed and does not incur apparent communication overhead. We implement CCD and evaluate its performance with extensive experiments and simulations. The results demonstrate that CCD is accurate, scalable, and applicable to a variety of sensor networks.

Keywords: Sensor Networks, Event Dectection, Coding Cost

I. INTRODUCTION

Wireless sensor networks (WSNs) have been widely applied for environment surveillance, battlefield monitoring, healthcare, and traffic monitoring. Typical WSNs consist of a number of cheap sensor nodes with resource-limited hardware and work in a self-organized manner. Event detection is a significant application of WSNs, where sensor nodes monitor the environment and collaboratively issue alarm to the base station upon the occurrence of an event.

Locating event, namely detecting an event and identifying an area in the network where the event occurs, is a significant issue but face many challenges. Due to the self-organized nature and resource constraints of WSNs, many existing approaches address the accuracy and robustness of event detection. Those approaches usually employ certain event models constructed with prior domain knowledge of the deployed system. The resulting event detection processes appear to be cost-inefficient, which require either intensive data exchanges among neighboring nodes or caching large columns of history data. Moreover, in order to determine an event type, most approaches require the location information of sensor nodes, which cannot be easily realized in many WSN deployments.

Our work is motivated the need of low-cost highly-accurate event detection in GreenOrbs [1], a large-scale WSN system in the forest. GreenOrbs collects various environmental data, such as temperature, humidity, illumination, and the content of carbon dioxide in the air. GreenOrbs is aimed at supporting long-term environmental surveillance, which requires the ability of locating event, e.g. to detect and locate forest fire and other disasters. Like many WSN systems, GreenOrbs deploys hundreds of resource-constrained sensor nodes and uses non-rechargeable batteries as the power supply. The resource-consuming event detection approaches cannot be applied in the context of GreenOrbs. Nevertheless, it is very hard to obtain the location information of all sensor nodes in the wild forest. Locating events in such a scenario is clearly a challenging task.

In order to address the above issues, we propose a model-free event detection approach called *Coding Cost based event Detection* (CCD). Coding cost is a metric that quantifies the diversity of a set of sensor readings. Our real-world observations indicate an interesting and significant fact that when an event occurs, the data distribution of sensor readings in the network must become more diverse, and the coding cost corresponded to the event region will dramatically increase.

We incorporate CCD into the inherent data collection mechanism of WSNs. CCD passively constructs a gradient map of coding cost throughout the network. An event is then detected and located where a change point of gradient appears without any sensor node's location.

Our contributions in this work are summarized as follows.

First, we present our findings on the characteristics of data distribution upon event occurrences based on a real system. Then, we propose to adopt coding cost as a novel metric for event detection and present the essential definitions.

Second, we propose CCD, a model-free approach for event detection and auto-identification event in location-free manner. CCD is fully distributed and does not incur apparent communication overhead. To the best of our knowledge, we are the first to use coding cost to detect event.

Third, we implement CCD and evaluate its performance with extensive experiments and simulations. The results demonstrate the CCD outperforms the state-of-the-arts

978-0-7695-4469-4/11 $26.00 © 2011 IEEE
DOI 10.1109/MASS.2011.72

approaches in terms of detection accuracy, scalability, and applicability.

The rest of this paper is organized as follows. Section II discusses the related work. We elaborate on the design of CCD in Section III, followed by performance evaluation in Section IV. We conclude in Section V.

II. Related Work

Event detection is an essential task in many applications [2-6] of WSNs, such as environment surveillance[7, 8] and safety guard. Most of the existing works resort to event-oriented query processing or model fitting procedure [9-16] to detect events.

Directed Diffusion [13] addresses the event-based queries by diffusing the interests into the monitoring sensor network. When the predefined events are detected, every node can cache, or transmit data based on the interests request and cached data previously. However [13] does not consider the temporal and spatial relationship among the sensor readings, and the event-based approach only detects the predefined events.

The GOUGAR [17] addresses the event queries by introducing a sensor network database system and employing a threshold-based detection mechanism providing three types event queries: snapshot, historical, and long-running queries.

TinyDB [12] aims to address event detection by employing a various SQL-like specified attribute thresholds to compare the sensor readings of attributes. It proposes a distributed method to construct contour maps of environment readings with global knowledge of node locations.

But active detection method would raise extra traffic cost. Compared to the active approaches like TinyDB, our approach in this work passively detects events in a location-free context and does not incur any extra traffic cost.

Fault and anomaly detection is a typical application of event detection. FIND [18] addresses the issue of detecting data faults of sensor nodes, by utilizing the monotonicity of sensor readings (e.g. acoustic and radio signals) around events. They assume sensing readings of nodes imply their physical distances to the event region as well. A node would be marked as a faulty node if the sensing data rank and the distance rank significantly mismatch with each other. For effective detection, FIND requires knowledge of node locations and intensive computation.

Ding *et al.* [19] propose an algorithm for fault node detection and event boundary detection, which employs the concepts of outlier detection and data mining in history data set. Each sensor obtains its physical location from GPS equipment, and exchanges the history sensing data sets with its neighbors. Based on the history sensing data sets, each node calculates the fault status by using statistics methods and data mining and determines the event boundary with random bisection or random trisection methods. These methods usually incur high traffic cost among nodes.

To sum up, the existing work of event detection usually relies on pre-defined models or energy-consuming event queries to identify events. Meanwhile, knowledge of node locations is common requirement for purpose of locating events. The above factors limit the detection accuracy, scalability, and applicability of existing approaches, making them unsuitable to resource-constrained large-scale WSNs.

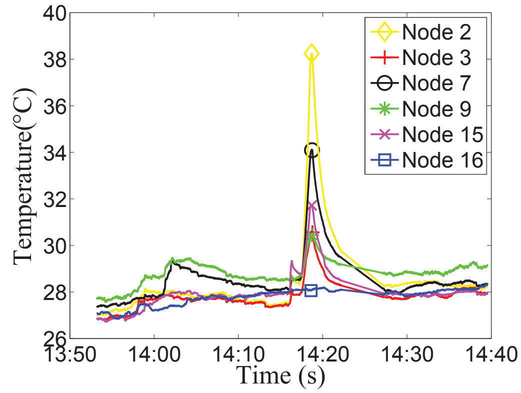


Figure 1: The changing model of demo experiment

III. Design

In this section, we present the design of CCD. First, we propose the characterization for events, including real-world observation. Second, based on the gradient map of coding cost, we identify three patterns of events according to their relative network locations. Correspondingly, a FSM (Finite State Machine) is designed to determine the event pattern passively along the data collection process. Third, we present the CCD algorithm by using the proposed FSM.

A. Characterization of Event

As we discuss in the previous section, most existing approaches of event detection are model-based or rely on cost-inefficient event queries. Is there a way to identify the occurrence of events without any query or predefined model? To answer this question, we take some preliminary attempts to observe the event characteristics on a test-bed (40 TelosB motes), with particular interest on the data distribution of sensor readings in the network.

In order to imitate the real scenarios, we use a heater to heat a portion of nodes for around 50 seconds. In this way, we imitate a fire event which occurs in a small square area. We then collect all the sensor readings during the whole process back to the sink node for analysis.

The experimental result is shown in Figure 1. Interestingly, we see that the sensor readings of event nodes, (i.e. Nodes 2, 3, 7, 9, and 15) have similar changing trends along time. Specifically, some sensor readings reach a very high climax (Node 2), while the readings of some other sensors (Nodes 3 and 9) are relatively less increased. In words, the fire event increases the diversity of sensor readings in the event region. Meanwhile, the readings on a node outside the event region (i.e. Node 16) don't change apparently throughout the event.

Regarding the above observation, an immediate question is, can we identify events according to the diversity of sensor readings? The answer is yes. As we know, the extent to which a data set can be compressed is determined by the data diversity in that set. We can borrow the technique of data compression to infer the diversity of a data set. Thus we introduce a new metric coding cost into event detection.

Definition: The coding cost metric is the average expected shortest length of code based on the probability density function (PDF) of sensing data as follows:

$$CC_{coding\cos t} = CC(z) = \frac{\sum_{i=1}^{N} -\log(pdf(x_i))}{N}$$

Here z denotes the sensing data set; x_i represents a unique reading and N denotes the number of sensing reading in the data set z. We assign 2 to the basis of logarithm for reasonable representation in computer information.

In order to better understand the relationship of coding cost in network. We introduce a concrete example as illustrated in Figure 2. The whole region covered by the sensor network is segmented into three areas: event origin area, event extension area, and non-event area. Lines depict the routing links among the nodes and the numbers in circles denote sensor IDs. Event origin area is the area where some event occurs, such as abnormal temperature, pressure, the concentration of carbon dioxide, denoted as R_{event}. Event extension area, which is denoted as $R_{ex\text{-}event}$, does not include any node covered by the event, but the sensors within are used to forward the event data to the sink node. Non-event area is the normal area where there is not any event influence, represented by $R_{non\text{-}event}$.

Nodes 6 and 7 produce the environment readings which is different from the readings from nodes in the non-event region in Figure 2. Every node can get the compression rate by collecting sensing data using the existing routing tree. For example Node 4 has three children nodes, namely two event nodes and one non-event node. We can calculate its naive compression rate by $rate(n_i)=c_e/c_{all} = 2/3$. There c_e and c_{all} denote the types and the total number of collected data packets from the sub-routing tree, respectively. Moreover we similarly get the compression rates of Nodes 2 and 1. Intuitively, compression rates on these nodes have gradient relationship, i.e. $rate(n_4)>rate(n_2)>rate(n_1)$. The following theorem defines the relationship of coding cost from event region to sink.

Theorem I. Suppose there are N_e sensors in event region R_e and N_n sensors in non-event region R_n. D_x denote the data set of sensors group x in the corresponding region. Then the coding cost of three data sets D_{N_n} , D_{N_e}, and $D_{N_n} \cup D_{N_e}$ satisfy the inequality $cc_{D_{N_n}} < cc_{D_{N_n} \cup D_{N_e}} < cc_{D_{N_e}}$

Proof: Referring to Figure 2, let the coding costs of R_n, R_e , and $R_n \cup R_e$ be denoted by c_n, c_e, and c_a, respectively.

$$c_n = Coding(D_{N_n}),\ cc_{D_{N_n}} = c_n / N_n$$

$$c_e = Coding(D_{N_e}),\ cc_{D_{N_e}} = c_{N_e} / N_e$$

$$c_a = Coding(D_{N_a}),\ cc_{D_{Na}} = c_a / N_a$$

Here $D_{N_a} = D_{N_n \cup N_e} = D_{N_n} \cup D_{N_e}$ and $N_a = N_e + N_n$.

$$\begin{aligned} c_a &= Coding(D_{N_n \cup N_e}) \le c_n + c_e \\ &= cc_{D_n} * N_n + cc_{D_e} * N_e \quad \because cc_{D_n} \ll cc_{D_e} \\ &< cc_{D_e} * (N_n + N_e) \end{aligned}$$

Then we have

$$\begin{aligned} cc_{D_a} &= \frac{c_a}{N_a} < \frac{cc_{D_e} * (N_n + N_e)}{N_n + N_e} = cc_{D_e} \\ &= \frac{c_a}{N_a} > \frac{cc_{D_n} * (N_n + N_e)}{N_n + N_e} = cc_{D_n} \end{aligned}$$

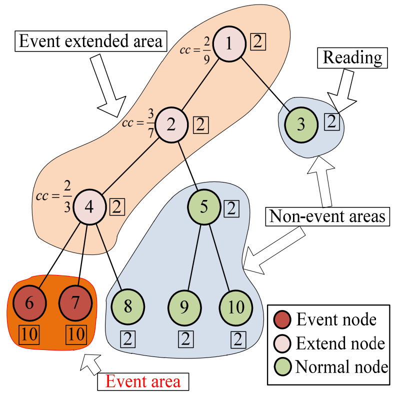


Figure 2: Coding costs in a WSN upon event occurrence

Furthermore the coding cost based gradient map is constructed from event region to sink in a manner that every node calculates the coding cost by itself. We present CCD that takes advantage of this discipline to identify event without locations.

B. Event Patterns

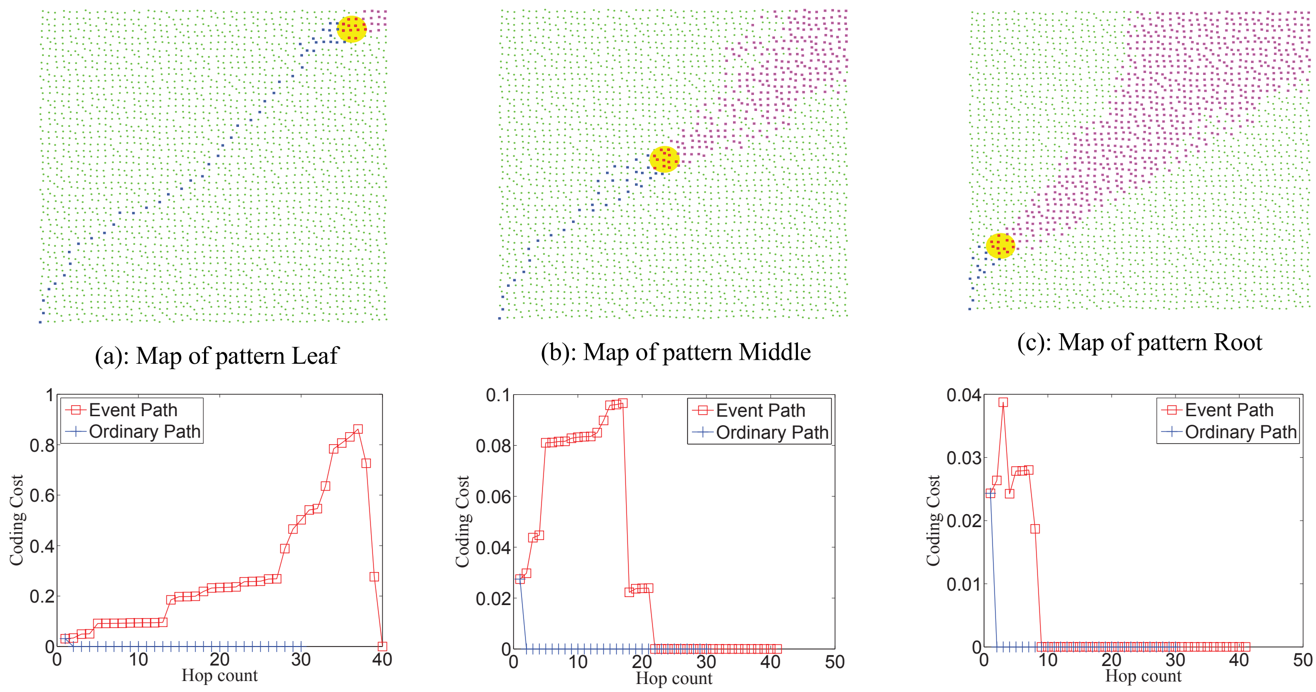
In this sub-section, we focus on analyzing the event patterns on the routing tree, based on which CCD determines the relative network location of an event. We present three patterns to describe different event types and a FSM (finite state machine) to identify pattern of which the event belongs to.

As illustrated in Figure 3, 2500 nodes are randomly deployed in 500*500 m[2] two-dimensional square field. As mentioned above, every sensor senses the environment and generates sensor readings within its corresponding field. Note that we assume that sensors close to each other have similar sensor readings. The closer the distance between two sensors is, the more similar values they have.

Event region is a circular area and the average links number of each node is equal to 7.34. Furthermore we classify the interested nodes into three groups: Group G_1 denotes the set of nodes that use nodes in event region as relay nodes, marked in pink; Group G_2 denotes the set of nodes in event region, marked in red; Group G_3 denotes the set of nodes that relay packets for the nodes in event region, marked in blue.

On all sensor nodes, we employ aggregation of the coding cost of children nodes, so as to create a gradient map along the routing tree. Along with the routing tree, a series of coding cost gradient map is constructed concomitantly in a distributed manner without extra communication cost. We define three event patterns, namely Leaf, Middle, and Root. They respectively represent the event types that the event region is close to the network boundary, the middle of the network, and the area near sink node, as shown in Figures 3 (a), (b), and (c). Note that the readings of non-event region are aggregated easily to a same value interval.

In pattern Leaf, the sensor data in the non-event region do not go through the event region, but the sensor data in the event region have to enter and pass the non-event region via the



(a): Map of pattern Leaf (b): Map of pattern Middle (c): Map of pattern Root

(d): Coding cost of pattern Leaf (e): Coding cost of pattern Middle (f): Coding cost of pattern Root

Figure 3: Leaf, Middle and Root patterns and corresponding data distributions in event occurring scenario

existing routing tree to deliver data to sink node. Figure 3(d) shows the comparison between two routing paths. One path crosses the event region and the other is completely outside the event region. That figure gives us more evidences to support our findings. Dealing with pattern Middle is more complicated than dealing with pattern Leaf. There are actually two phases of the gradients, namely the data entering the event region and the data going out the event region. According to Figure 3(e), we can get similar results to explain the curves, which significantly change during the above-mentioned two phases. In pattern Root, the peak of curve is in the front, which illustrates the location of event region near the sink node, as shown as Figure 3 (f).

We use these three patterns of data distribution to describe different types of events. Furthermore, the curves of coding cost are also viewed as a perspective of network gradient map, which can then be utilized to detect events in a passive way. Hence we need a mechanism to identify the event pattern along the routing tree. We propose a FSM to depict this automated procedure, as shown in Figure 4. We predefine Δ as a threshold of the change in coding cost. If the change in coding cost is greater than Δ, the first input parameter of FSM is set to 1, otherwise set to 0. We define n_r, n_m, and n_f as the three hop count types that match distance between event region to sink and segment the whole network into three regions: the boundary, the middle, and the sink-around area. If a hop-count n is greater than n_r, it belongs to the remote distance type; if a hop-count n is greater than n_m and less than n_r, it belongs to the middle distance type; if a hop-count n is less than n_f, it belongs to the sink-around distance type.

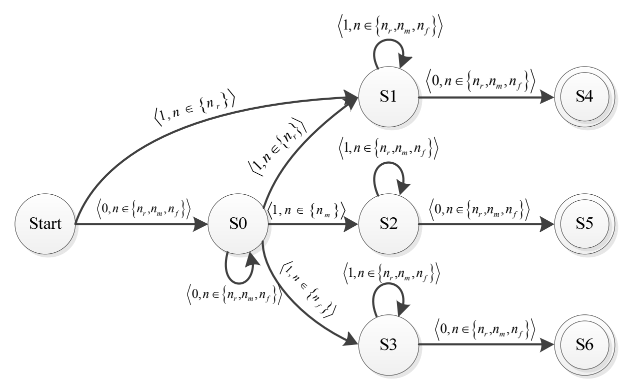


Figure 4: A FSM for determining the event patterns

We define three groups of FSM states: pre-entry event state (S0), entry event states (S1, S2 and S3), post-entry event states (S4, S5 and S6). The pre-entry event state S0 indicates the packet does not enter the event region; the entry event states denote the packet has entered the event region; the post-entry event states represent the package has been left the event region already. S1 and S4 belong to pattern Leaf; S2 and S5 belong to pattern Middle; S3 and S6 belong to pattern Root; S0 belongs to pattern Leaf, Middle and Root commonly.

As shown in Figure 4, from the start state, we transit to state S0 when the inputs are equal to $\langle 0, n \in \{n_r, n_m, n_f\} \rangle$ that indicate mismatching changing coding cost and normal hop-count, alternatively transit to state S1, when the inputs are equal to $\langle 1, n \in \{n_r\} \rangle$ that indicate the packet enter the event region which belong to pattern Leaf. Then according to the inputs, FSM transfer the state along the routing tree. The final states S4, S5 and S6 denote the event patterns Leaf, Middle and Root respectively. Why does the FSM seem incomplete in the first state transfer? Since the inputs of first state transferring can only have two combinations: $\langle 0, n \in \{n_r, n_m, n_f\} \rangle$ and $\langle 1, n \in \{n_r\} \rangle$. Input $\langle 1, n \in \{n_m, n_f\} \rangle$ is illogical value because of previous input of this value must be the value $\langle 0, n \in \{n_r\} \rangle$ which would cause FSM transfer state to S0. As a result, the inputs of first transferring are completeness.

The node would receive the multiple FSM packets from its children nodes. The merging principle of FSM state is the priority level of Root greater than Middle greater than Leaf which imply S3 greater than S2 greater than S1 and S6 greater than S5 greater than S4. In other words, multiple FSM states were conflated in parent node based on the FSM states priority level. Note that if there is not event occurrences in network, the FSM state would stay at S0 until sink receives this packet. According the network size and density, the parameters can be decided straightforwardly.

C. CCD Algorithm

In this sub-section, we propose the CCD algorithm to detect the event based on the gradient map of the coding cost and identify the event pattern simultaneously.

The approach constructs gradient map of the coding cost and determines which pattern the event belongs to and the corresponding proximate event region size (hop count). First, we assume each node has approximate sensing period which potentially implies the relationship of the sensing readings in whole sensor network and the routing tree is stable in a period of time owing to environment.

Note that our algorithm runs in each node in distributed manner. The detail as follows: When receives all packet M_c that consist of the coding C of data, maximal changing of coding cost CC_{max} and a two-tuple $<m_{state}, n_{hopcount}>$ of FSM from router children, the node aggregates the coding of data based on the compression method, calculates the CC_{max} based on the changing of coding cost itself, merges the state of FSM, updates the hop count $n_{hopcount}$ in packet. Here When FSM stays S0, S1, S2 or S3, the $n_{hopcount}$ denotes the hop count in current state. When FSM stays S4, S5 or S6, the $n_{hopcount}$ denotes the hop count in previous state. Last, the node sends the new aggregating packet M_c to its parent along the routing tree. In this processing, the whole sensor network would construct a gradient map of the coding cost in distributed style, and the sink get the pattern of event and the proximate corresponding event region size as well. Note that m_{state} denotes the current state of this packet in routing path shown in Algorithm I.

Algorithm I: Event Capture
receive all packet M_c $<C, CC_{max}, m_{state}, n>$
aggregates the coding C
calculate cc, changing of cc
recalculate C based on the Compression Coding
if $cc > CC_{max}$
$\quad CC_{max}=cc$
end if
update $<m_{state}, n>$ based on $<\Delta$, n$>$ and FSM
deliver $<C, CC_{max}, m_{state}, n>$ to parent node

IV. EVALUATION

In this section, we conduct overall implementation and analysis to evaluate the effectiveness and the detection accuracy of our CCD approach. A series of comparison results, our CCD and centralized optimal approach are given.

A. Detection Accuracy

In order to evaluate the performance of our approach, we obtain the optimizing results as ground truth by the existing centralized event detection solution. Firstly our implementation is composed of 100 TelosB motes with MSP430 MCU and CC2420 radio. The distance of each row and column is equal to 10 m. The sink node can collect the data from the every mote using the routing protocol such as CTP [20]. Due to the setting of sensing range, every node has 8 one-hop neighbors at most.

In our experiment, the event detection accuracy is defined as the detection accuracy metric at the event level which has three aspects by varying the parameters Δ, r, and d. Here Δ is the changing of coding cost threshold, r is diameter of event region and d is the data distribution in event region. We consider three different data distributions in event region, which consisted of Uniform distribution, Normal distribution and Laplace distribution.

We assume the mean of the nodes readings in event region is three times more than the reading in normal region, the data distribution in event region is equal probability combination of three data distributions: the Uniform distribution, Normal distribution and Laplace distribution and the location of event region is random in the deployment area. Corresponding parameters of data distributions are set to Uniform (50, 70), Normal (60, 20) and Laplace (60, 10).

Figure 5 plots the event level accuracy by varying Δ in [0.28, 1.39] 100 times. We can lean from the curve that if the coding cost threshold is small than 0.38, the event detecting accuracy would guarantee 100%.

The influences of the data distributions in event region and the event detection accuracy are shown in Figure 6. Due to the feature of kurtosis, the Normal distribution would cause the higher coding cost than other two data distributions and get the higher event detection accuracy under the same coding cost threshold 0.43. Hence, the small enough threshold would obtain 100% event detection accuracy among three data distributions as well.

By setting three different event regions, we get three curves that imply the larger event region would get the higher event detection accuracy which is determined the higher coding cost shown in Figure 7. The setting of event region diameter must guarantee the event nodes is minority within the whole network nodes.

Note that our experiments do not assume the prior domain knowledge of the event and use corresponding ranking scheme detection method. Because CCD approach would steady get 100% detection accuracy based on the accurate event prior knowledge and the ranking list of coding cost.

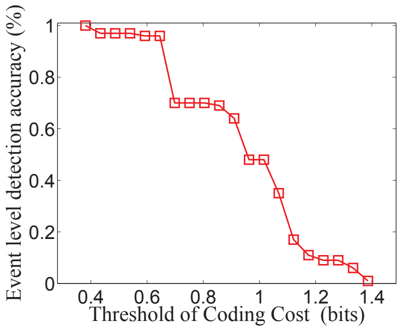


Figure 5: The event level detection accuracy varying threshold *Δ*, (*r*=Uniform (15, 30))

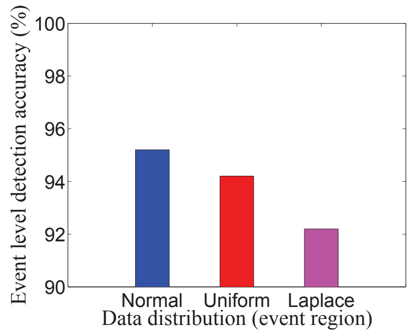


Figure 6: The event level detection accuracy among different *d*, (*r*=16, *Δ*=0.43)

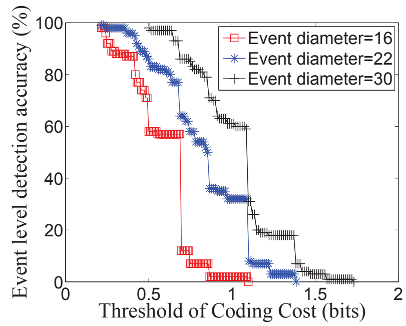


Figure 7: The event level detection accuracy among different *r*, (Uniform (50, 70), Normal (60, 20) ,Laplace (60, 10))

V. Conclusion

The proliferated use of WSN technology attracts increasing attention in the past decade. In most WSN applications for environmental monitoring and field surveillance, event detection is a fundamental task. Our work in this paper addresses event detection and identification with an integrated approach CCD. The main novelty of CCD includes two aspects. First, by introducing the idea of coding cost, CCD realizes model-free passive event detection, which saves communication cost and ensures scalability. Second, CCD supports location-free event identification, which makes it a practical and attractive solution for real WSN systems.

In our future work, we plan to implement CCD with the large-scale GreenOrbs system and carry out observations on CCD's behavior with real-world events. We believe such observations will provide us more insights on improving the applicability, accuracy, and efficiency of event detection techniques for WSNs.

Acknowledgments

This work is supported in part by China 973 Program under Grant No.2011CB302705, NSFC under Grant No.60873262, and NSF CNS-0832120.

References

[1] www.greenorbs.org.
[2] P. Vicaire*, et al.*, "Achieving long-term surveillance in vigilnet," *ACM Transactions on Sensor Networks (TOSN),* vol. 5, p. 9, 2009.
[3] G. Tolle*, et al.*, "A macroscope in the redwoods," in *ACM SenSys*, 2005, pp. 51-63.
[4] L. Selavo*, et al.*, "Luster: wireless sensor network for environmental research," in *ACM SenSys*, 2007, pp. 103-116.
[5] P. Dutta*, et al.*, "Trio: enabling sustainable and scalable outdoor wireless sensor network deployments," in *IEEE/ACM IPSN*, 2006, pp. 407-415.
[6] A. Mainwaring*, et al.*, "Wireless sensor networks for habitat monitoring," in *ACM WSNA*, 2002, pp. 88-97.
[7] M. Li and Y. Liu, "Underground coal mine monitoring with wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN),* vol. 5, pp. 1-29, 2009.
[8] L. Yunhao*, et al.*, "Passive diagnosis for wireless sensor networks," *Networking, IEEE/ACM Transactions on,* vol. 18, pp. 1132-1144, 2010.
[9] M. Li*, et al.*, "Non-threshold based event detection for 3D environment monitoring in sensor networks," presented at the ACM ICDCS, 2007.
[10] W. Xue*, et al.*, "Contour map matching for event detection in sensor networks," in *ACM SIGMOD*, 2006, pp. 145-156.
[11] D. J. Abadi*, et al.*, "REED: robust, efficient filtering and event detection in sensor networks," in *ACM VLDB*, 2005, pp. 769-780.
[12] J. Hellerstein*, et al.*, "Beyond average: Toward sophisticated sensing with queries," in *ACM IPSN*, 2003, pp. 553-553.
[13] C. Intanagonwiwat*, et al.*, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *ACM MobiCom* 2000, pp. 56-67.
[14] S. Li*, et al.*, "Event detection services using data service middleware in distributed sensor networks," presented at the ACM IPSN, 2003.
[15] I. Solis and K. Obraczka, "Efficient Continuous Mapping in Sensor Networks Using Isolines," presented at the IEEE MobiQuitous, 2005.
[16] M. Li and Y. Liu, "Iso-map: Energy-efficient contour mapping in wireless sensor networks," *IEEE transactions on knowledge and data engineering,* pp. 699-710, 2009.
[17] P. Bonnet*, et al.*, "Querying the physical world," *Personal Communications, IEEE,* vol. 7, pp. 10-15, 2000.
[18] S. Guo*, et al.*, "FIND: faulty node detection for wireless sensor networks," in *ACM SenSys*, 2009, pp. 253-266.
[19] W. Wu*, et al.*, "Localized outlying and boundary data detection in sensor networks," *IEEE transactions on knowledge and data engineering,* pp. 1145-1157, 2007.