DISTRIBUTED ANOMALY DETECTION IN WIRELESS SENSOR NETWORKS

Sutharshan Rajasegarar[1], *Christopher Leckie*[2], *Marimuthu Palaniswami*[1]

ARC Special Research Center for Ultra-Broadband Information Networks
[1]Department of Electrical and Electronic Engineering
[2]Department of Computer Science and Software Engineering
The University of Melbourne, Australia.
Email: {r.sutharshan, swami}@ee.unimelb.edu.au,
caleckie@csse.unimelb.edu.au

James C. Bezdek

Computer Science Department
University of West Florida
USA.
Email: jbezdek@uwf.edu

ABSTRACT

Identifying misbehaviors is an important challenge for monitoring, fault diagnosis and intrusion detection in wireless sensor networks. A key problem is how to minimise the communication overhead and energy consumption in the network when identifying misbehaviors. Our approach to this problem is based on a distributed, cluster-based anomaly detection algorithm. We minimise the communication overhead by clustering the sensor measurements and merging clusters before sending a description of the clusters to the other nodes. In order to evaluate our distributed scheme, we implemented our algorithm in a simulation based on the sensor data gathered from the Great Duck Island project. We demonstrate that our scheme achieves comparable accuracy compared to a centralised scheme with a significant reduction in communication overhead.

1. INTRODUCTION

Wireless sensor networks comprise a large number of tiny sensor nodes that have limited power, bandwidth, memory and computational capabilities [1]. These inherent limitations of sensor nodes can make the network more vulnerable to faults and malicious attacks [2]. A key challenge in identifying misbehaviors in wireless sensor networks is to develop algorithms for detecting anomalies in the network, such that these algorithms minimise their communication overhead and energy consumption in the network. In this paper, we address this problem by presenting a distributed method to identify misbehavior or anomalies in wireless sensor networks.

Misbehaviors in the network can be identified by analysing the data from the sensor nodes. A node may show misbehaviors whenever a fault occurs or due to malicious activity by compromised sensors [3]. In both cases, misbehaviors can be identified by analysing sensor or traffic measurements to discriminate normal behavior from anomalous behavior. Note that the underlying distribution of these measurements may not be known *a priori*. Therefore, anomaly detection in data with an unknown distribution is an important problem to be addressed in wireless sensor networks. Our approach to this problem is to use a form of cluster-based anomaly detection in a distributed environment.

Previous attempts to identify anomalies and perform distributed data clustering can be found in the literature. Bandyopadhyay et al [4] have proposed a distributed k-means clustering algorithm, but have not addressed the problem of anomaly detection. Onat et al [5] have identified anomalies using a predefined statistical model. Loo et al [3] have proposed an intrusion detection scheme for identifying abnormal traffic patterns using fixed-width clustering. However they have not considered co-operation between nodes.

Our approach to this problem is based on a distributed, non-parametric anomaly detection algorithm to identify anomalous measurements in nodes. Rather than communicating each individual sensor measurement to a central node for analysis, we first cluster the measurements. Sensor nodes can then report cluster summaries, rather than individual measurements. Moreover, intermediate sensor nodes merge cluster summaries before communicating with other nodes. Using this approach, our distributed anomaly detection scheme can minimise communication overhead, which is a major source of energy consumption for sensor nodes.

In this paper, we demonstrate the effectiveness of our approach based on sensor data gathered from the Great Duck Island project [6]. In comparison to a centralised approach to anomaly detection, we show that our distributed approach can achieve significant reductions in communication overhead, while achieving comparable accuracy. The rest of the paper is organised as follows. Section 2 formally introduces the problem of distributed anomaly detection. Section 3 describes our distributed solution to the problem. The evaluation results for our scheme are explained in Section 4.

2. PROBLEM STATEMENT

Our aim is to identify anomalies in the data gathered by sensor nodes in a wireless sensor network. We consider a set of sensor nodes $S = \{s_i : i = 1...n\}$ having a hierarchical topology as shown in Figure 1. The sensors are deployed in a homogeneous environment, in which the measurements taken have the same unknown distribution. All the sensor nodes are time synchronised. At every time interval Δ_k each sensor node s_i measures a feature vector x_k^i. Each feature vector is composed of features or attributes v_{kj}^i, where $x_k^i = \{v_{kj}^i : j = 1...d\}$ and $x_k^i \in \Re^d$. After a window of m measurements, each sensor s_i has collected a set of measurements $X_i = \{x_k^i : k = 1...m\}$. An outlier or anomaly in a set of data is defined in [7] as an observation (or subset of observations) that appears to be inconsistent with the remainder of that data set. Our aim is to find the outliers $O \subset X$ in the combined set of measurements $X = \bigcup_{i=1..n} X_i$.

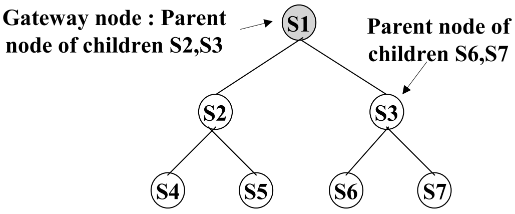
Gateway node : Parent node of children S2,S3
S1
Parent node of children S6,S7
S2
S3
S4
S5
S6
S7

Fig. 1. Example of a Hierarchical Network Topology

3. ANOMALY DETECTION

Clustering is the process of finding groups of similar data points, such that each group of data points is well separated [8]. We have used Euclidean distance as the dissimilarity measure between pairs of data [8]. The clustering algorithm we use here is based on fixed-width clustering, used by Eskin et al [9] for anomaly detection. Once the clusters are formed, outliers or anomalous clusters are classified using an anomaly detection algorithm as detailed in Section 3.4. This algorithm identifies the anomalous clusters based on the nearest neighbor distance among clusters.

One way of detecting anomalies is to use a centralised approach. In this approach, at the end of every time window of measurements, each sensor node s_i sends all its data to its gateway node $s_g \in S$. A gateway node is the root node in a hierarchical topology of sensors. The gateway node s_g combines its own data X_g with the received data set $X_R = \bigcup_{i=1..n-1, i \neq g} X_i$ and forms a combined data set $X = X_R \cup X_g$. A clustering algorithm is run on this data set X to form a set of clusters $C = \{c_r : r = 1...c\}$.

Figure 2a shows an example of the centralised approach for a single level hierarchical topology. Initially, the data vectors at each node before the data transmission are shown in Figure 2a(i). Once the leaf nodes $S2, S3$ and $S4$ transmit all their data to the gateway node $S1$, the combined data vectors are shown in Figure 2a(ii). Then node $S1$ clusters the combined data set as shown in Figure 2a(iii). Finally, the anomaly detection algorithm is run at node $S1$ on those clusters. This centralised approach has several drawbacks. First, a large volume of raw data is transmitted over the network. This requires each sensor to be in active mode for communication for a longer time duration than in sleep mode. This communication overhead can significantly reduce the life time of the network [10]. Second, there is a greater communication load in the nodes that are in close proximity to the gateway node, which in turn depletes the life time of the network. Therefore, there are two challenges to overcome in the anomaly detection process. First, we require a distributed anomaly detection scheme to detect anomalies. Second, we need to minimise communication in order to maximise energy efficiency.

Our approach is to distribute the anomaly detection process to all sensors in the network. In this approach, at every time window of m measurements the following operations are performed.

- Each sensor node $s_i \in S$, performs the clustering operation on its own local data X_i and produces the clusters $C_i = \{c^i_r : r = 1...l\}$. Note that the number of clusters l is determined algorithmically.
- Sensor node s_i sends the sufficient statistics (see below) of its clusters C_i to its immediate parent $s_p = Parent(s_i)$. Each cluster $c^i_r \in C_i$ can be sufficiently represented by its centroid and the number of data vectors it contains [11]. If the number of data vectors in the cluster c^i_r is $N^i_r \leq m$ and the set of data vectors contained in that cluster is $X^i_r = \{x^i_q : q = 1...N^i_r\}$, then the linear sum of the data vectors of that cluster can be defined as $L^i_r = \sum_{q=1}^{N^i_r} x^i_q$. Hence, the centroid of the cluster can be expressed using the above two quantities as L^i_r / N^i_r. Therefore, the sufficient statistics of a cluster c^i_r are the number of data vectors N^i_r and the linear sum of the data vectors L^i_r of that cluster.
- The parent node s_p combines its clusters C_p with the clusters $C = \bigcup_{i \in children(s_p)} C_i$ from its immediate children and forms a combined set of clusters $C_c = C \cup C_p$.
- The parent node s_p merges the combined cluster set C_c to produce a merged cluster set $C_h = \{c^h_r : r = 1...f\}$, where $f \leq |C_c|$ (Section 3.3).
- Then the parent node s_p sends the sufficient statistics of the merged clusters C_h to its immediate parent.
- This process continues recursively up to the gateway node $s_g \in S$, where an anomaly detection algorithm is applied to its merged clusters C_h to identify the anomalous cluster set $C_a \subset C_h$ (Section 3.4).

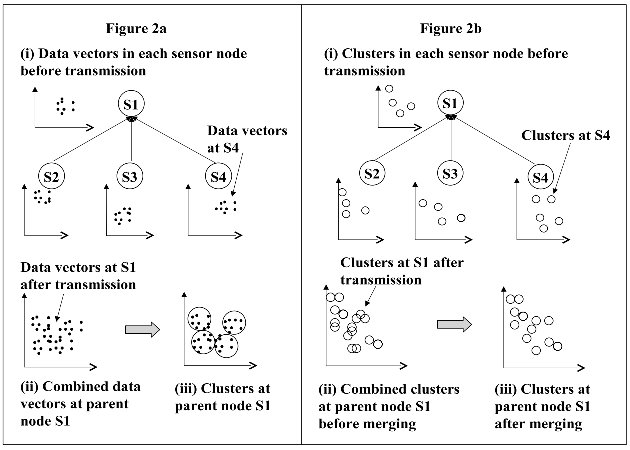
Figure 2a
(i) Data vectors in each sensor node before transmission
S1
Data vectors at S4
S2
S3
S4
Data vectors at S1 after transmission
(ii) Combined data vectors at parent node S1
(iii) Clusters at parent node S1
Figure 2b
(i) Clusters in each sensor node before transmission
S1
Clusters at S4
S2
S3
S4
Clusters at S1 after transmission
(ii) Combined clusters at parent node S1 before merging
(iii) Clusters at parent node S1 after merging

Fig. 2. Figure 2a. Centralised approach: a(i) Data vectors at individual nodes, a(ii) Combined data vectors at the gateway node $S1$, a(iii) Clusters formed at node $S1$. Figure 2b. Distributed approach: b(i) Clusters formed at each node, b(ii) Clusters combined at gateway node $S1$, b(iii) Clusters merged at node $S1$.

Figure 2b shows an example of our distributed approach for a single level hierarchical topology with child nodes $S2, S3$ and $S4$ and gateway node $S1$. Initially, all nodes perform the clustering operation on their own local data (Figure 2b(i)). Then the nodes $S2, S3$ and $S4$ send the sufficient statistics of the clusters to their parent node $S1$. $S1$ combines the received cluster set with its own clusters (Figure 2b(ii)) and then merges the clusters and produces merged clusters C_h (Figure 2b(iii)). Finally, node $S1$ identifies the anomalous clusters C_a from the merged clusters C_h.

Better load balancing is achieved by distributing the clustering process between all the nodes. Also the communication overhead is reduced by only sending the merged clusters, rather than the raw data. This also helps to extend the lifetime of the network. In the centralised case, the gateway node has complete information about the data in the network, whereas in the distributed case, the gateway node only has the merged cluster information of the nodes. Therefore there may be a slight reduction in the detection accuracy in the distributed case compared to the centralised case. In the following sections, we describe our approach in more detail.

3.1. Data Conditioning

Data conditioning transforms the data vectors from sensor measurements to a suitable form for use in distance based clustering. We use the commonly used Euclidean distance $D(x_1, x_2)$ as our distance measure between data vectors x_1 and x_2 [8].

Data features of a sensor node often lie within different dynamic ranges. In order to alleviate the effect of this on distance calculations, each data feature v_{kj}, $k = 1...m$ is transformed [12] via their respective mean and variance as $u_{kj} = (v_{kj} - \mu_{v_j})/\sigma_{v_j}$, where μ_{v_j} and σ_{v_j} are the mean and the standard deviation of the features v_{kj}, $\forall k$ respectively, and u_{kj} is the transformed feature. The resulting transformed feature u_{kj} will have zero mean and unit variance.

Further, each feature is normalised [8] to a range [0,1] as $\bar{u}_{kj} = (u_{kj} - min_{u_j})/(max_{u_j} - min_{u_j})$, where min_{u_j} and max_{u_j} are the smallest and the largest value of the transformed data feature respectively.

In the centralised detection scenario, all the data vectors from all the nodes are transmitted and available at the gateway node. Therefore the data conditioning can be performed at the gateway node on all the data. In the distributed case, the data conditioning is done at the local nodes using their own local data using the parameters (Mean, Standard deviation, Minimum and Maximum) calculated on their local data. However, the resulting normalised data will not exactly match that of the centralised case. We use the following procedure to find the global data conditioning parameters and to normalise the local data.

- Each sensor node s_i computes the following information on its local data vectors X_i.
  - Linear sum $LLS_i = \sum_{k=1}^{m} x_k^i$ and the linear sum of squares $LLSS_i = \sum_{k=1}^{m} (x_k^i)^2$ of the local data vectors.
  - Number of local data vectors $LN_i = |X_i| = m$.
  - Vector of maximum x_{max}^i and minimum x_{min}^i values for each attribute of the local data X_i.
- Each sensor node s_i sends the above information (LLS_i, $LLSS_i$, LN_i, x_{max}^i, x_{min}^i) to the gateway node s_g.
- Gateway node s_g collects the above local information from the children nodes and computes the global data conditioning parameters as follows.
  - Total data points in the network $N_G = \sum_{i=1}^{n} LN_i$.
  - Global mean $\mu_G = \frac{1}{N_G} \sum_{i=1}^{n} LLS_i$ and varance $\sigma_G^2 = \frac{1}{N_G} \sum_{i=1}^{n} LLSS_i - \mu_G^2$.
  - Global data maximum $x_{max}^G = maximum(x_{max}^i)$ and minimum $x_{min}^G = minimum(x_{min}^i)$, where $i = 1...n$.
- Gateway node s_g, then sends the global conditioning parameters (μ_G, σ_G, x_{max}^G, x_{min}^G) to all children.
- Each local node s_i uses these global conditioning parameters to condition its local data.

3.2. Clustering Algorithm

Our clustering algorithm is based on the fixed-width clustering algorithm used in [9, 3]. Fixed width clustering creates a set of clusters of fixed radius (width) w. Here the width w is a parameter to be specified by the user. First, a data vector is taken and used as the centroid (center) of the first cluster with radius w. Then for each subsequent data vector the Euclidean distance between the centroid of the current clusters and this data vector is computed. If the distance to the closest cluster center from the data vector is *less* than the radius w, the data vector is added into that cluster and the centroid of that cluster is adjusted to the mean of the data vectors it contains. If the distance to the closest cluster center is *more* than the radius w, then a new cluster is formed with that data vector as the centroid. This operation produces a set of disjoint, fixed width (radius of w) clusters in the feature space. The principle advantage of this simple approach is that only one pass is required, thus minimising storage and energy consumption. This efficiency is traded against the loss of flexibility and possible accuracy engendered by using a single threshold to determine all the clusters.

3.3. Merging of Clusters

We use a cluster merging technique to combine a pair of similar clusters into a single cluster. A pair of clusters c_1 and c_2 are similar if the inter-cluster distance $d(c_1, c_2)$ between their centers is less than the width w. If c_1 and c_2 are similar, then a new cluster c_3 is produced whose center is the mean of the centers of c_1 and c_2 and whose number of data vectors is the sum of those in c_1 and c_2. In our system, the merging procedure compares each cluster c_i with clusters $\{c_{i+1}, c_{i+2}, ...\}$, and merges c_i with the first cluster c_j such that $d(c_i, c_j) < w$ and $j > i$, should such a c_j exist.

3.4. Anomaly Detection Algorithm

The anomaly detection algorithm classifies clusters as either normal or anomalies. We use the average inter-cluster distance of the K nearest neighbor (KNN) clusters [13] to identify the anomalous clusters. The algorithm is as follows.

- For each cluster c_i in the cluster set C, a set of inter cluster distances $D_{c_i} = \{d(c_i, c_j) : j = 1...(|C| - 1), j \neq i\}$ is computed. Here $d(c_i, c_j)$ is the Euclidean distance between centroids of c_i and c_j, and $|C|$ is the number of clusters in the cluster set C.
- Among the set of inter-cluster distances D_{c_i} for cluster c_i, the shortest K (parameter of KNN) distances are selected and using those, the average inter-cluster distance ICD_i of cluster c_i is computed as follows,

$$ICD_i = \begin{cases} \frac{1}{K} \sum\limits_{j=1,\neq i}^{K} d(c_i, c_j) & K \leq |C| - 1 \\ \frac{1}{|C|-1} \sum\limits_{j=1,\neq i}^{|C|-1} d(c_i, c_j) & K > |C| - 1 \end{cases}$$

Our average inter-cluster distance computation differs from the one proposed by Chan et al [14] in the following way. Chan et al have used the whole cluster set C to compute the average inter-cluster distance ICD_i for a cluster c_i, whereas we have used the K nearest neighbor clusters of the cluster c_i to compute the average inter-cluster distance

ICD_i. The advantage of our approach is that clusters at the edge of a dense region are not overly penalised compared to clusters in the center of the region.

- A cluster is identified as anomalous if its average inter-cluster distance ICD_i is more than one standard deviation of the inter-cluster distance $SD(ICD)$ from the mean inter-cluster distance $AVG(ICD)$, i.e., a set of anomalous clusters $C_a \subset C$ are defined as $C_a = \{c_i \in C | ICD_i > AVG(ICD) + SD(ICD)\}$, where ICD is the set of average inter-cluster distances.

3.5. Complexity Analysis

Here we analyse the communication overhead, memory and computational complexity of the algorithms in more detail.

The data conditioning algorithm involves two types of communication overhead. First, each sensor node has to communicate once to the gateway node with the tuple $<LLS_i$, $LLSS_i$, LN_i, x^i_{max}, $x^i_{min}>$. Second, the gateway node has to communicate with all the children nodes once with the tuple $<\mu_G$, σ_G, x^G_{max}, $x^G_{min}>$. In practice, collection of global data conditioning parameters can be performed in the previous time window of measurements. Hence each sensor node incurs a computational and memory complexity of $O(m)$ to keep the normalised data, where m is the number of measurements during the time window.

The fixed-width clustering algorithm we use requires a single pass over the data. For each data vector, it computes the distance to each existing cluster. Hence the computational complexity is $O(mN_c)$, where $N_c(<< m)$ is the number of clusters. The memory complexity for each sensor node is $O(N_c)$, since each cluster requires a fixed length record.

The cluster merging operation compares cluster pairs with computational complexity $O(N_c^2)$. The anomaly detection algorithm compares each cluster to all other clusters in the cluster set C to find the K nearest neighbors with computational complexity $O(N_c^2)$.

In summary, each sensor node requires memory to keep $O(N_c)$ clusters and does not need to keep all data measurements in memory. Each sensor incurs $O(mN_c)$ computational complexity.

4. EVALUATION

Our aim is to evaluate the accuracy of the anomaly detection process using the centralised and the proposed distributed scheme. For our evaluation we consider a three-level hierarchical topology as shown in Figure 1.

Our test data were the sensor measurements collected by the Great Duck Island Project [6]. In every five minute interval, each sensor node recorded light, temperature, humidity, and pressure readings. For our evaluation, we selected data from 7 nodes, namely nodes 101, 109, 111, 116, 118, 122 and 123, within a 24 hour period on 1st July 2003. We selected three features: humidity, temperature and pressure readings from each sensor node. We have plotted the scatter plots for all the data from the 7 nodes and cleaned them by removing spurious or erroneous data. The cleaned data were labeled as $Normal$ and used for our evaluation purposes. The hierarchical topology was formed with these 7 nodes as shown in Figure 1. Node 101 is used as the gateway node, nodes 109 and 111 are used as the intermediate parent nodes and the other nodes are used as leaf nodes. A uniformly distributed, randomly generated set of anomalous data (of 20 data vectors for each node) were introduced into the tails of the distribution of each feature for two of the nodes (nodes 118 and 123). These introduced anomalous data measurements were labeled as $Anomalies$.

We implemented our centralised and distributed algorithms in a C++ simulation based on the real data. We used $K = 4$ as the KNN parameter in our simulations. The above process was repeated for different cluster width values w ranging from 0.02 to 2.02 in 0.04 intervals. For each of these results, the false positive rate and the false negative rates were calculated. A false positive occurs when a normal measurement is identified as anomalous by the algorithm, and a false negative occurs when an anomalous measurement is identified as normal by the algorithm. The false positive rate is the ratio between false positives and the actual normal measurements, and the false negative rate is the ratio between false negatives and the actual anomalous measurements. Also during the evaluation, the number of data vectors and the number of clusters communicated were recorded to compute the reduction in communication overhead. We have evaluated the system performance by analysing the sensitivity of detection accuracy by varying the cluster width and keeping the K value fixed. Evaluation of the distributed detection scheme for varying K values will be the subject of future work.

Figure 3 shows a graph of the false positive rate and the false negative rate as a function of the cluster width for the centralised scheme. Figure 4 shows the similar graph for the distributed scheme. It can be observed that in the lower cluster widths (0.02), the false negative rate is higher. This is because at lower cluster widths, a large number of clusters is produced for the data set, in which some of the clusters may even be singletons. Therefore, there is a greater chance of the anomalous data being divided into multiple clusters in close proximity. This may result in the average inter-cluster distances being smaller for the anomalous data, so that it is not detected by our anomaly detector. Similarly at higher cluster widths ($\geq$ 0.38) false negative rate again increases while the false positive rate becomes zero. This is because, at larger cluster widths, a small number of large clusters are produced for the data set. Hence there is a higher chance of the anomalous data being included in the large clusters which have been identified as normal by the detector. Hence, there is a range of threshold values (cluster width w) within the two extremes in which the system performs better. In this case, in the cluster width range from 0.06 to 0.38, the system provides better detection performance for both the centralised and distributed cases. This shows that the cluster width (w) is an important parameter for achieving better detection performance. In practice a proper cluster width can be selected by training the system before deployment.

Further, the average false positive rate for the distributed case is around 4%, while for the centralised case is around 3%. Hence the distributed scheme achieves a comparable accuracy to that of the centralised case. This is achieved with a huge reduction in the communication overhead in the network in the distributed case compared to the centralised case. Figure 5 shows that the distributed approach, when compared to the centralised case, realises savings in communication overhead that ranges from 85% to 98% over the cluster width range 0.06 to 0.38, i.e., upto a 50 fold reduction in communication overhead.

5. CONCLUSION

In this paper, we have presented a distributed anomaly detection algorithm based on data clustering to identify misbehavior in a

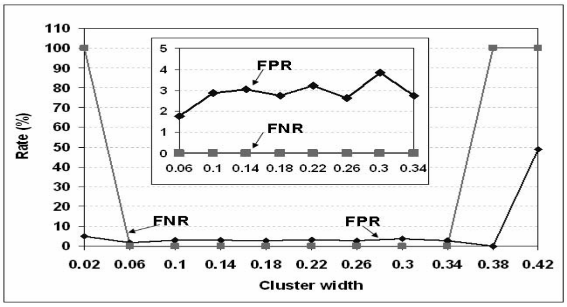

Fig. 3. Centralised scheme: False positive rate (FPR) and False negative rate (FNR) (%) Vs Cluster width (w). The graph in the inset is the expanded view of the same graph for the cluster width range from 0.06 to 0.34.

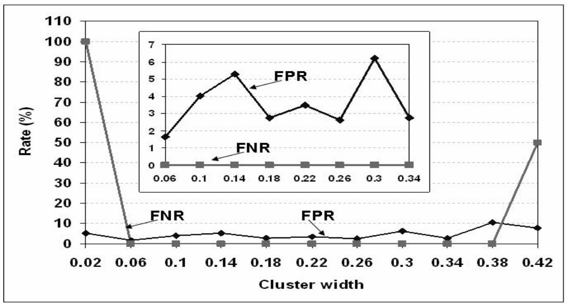

Fig. 4. Distributed scheme: False positive rate (FPR) and False negative rate (FNR) (%) Vs Cluster width (w). The graph in the inset is the expanded view of the same graph for the cluster width range from 0.06 to 0.34.

wireless sensor network. We have evaluated the scheme in a hierarchical topology, using a simulator on real data gathered from Great Duck Island. Our evaluation has shown that our distributed approach achieves comparable performance with the centralised case, while achieving a significant reduction in communication overhead. Our future research includes extending our evaluation to multiple KNN parameters. We will also simulate different kinds of known sensor network attacks and evaluate the accuracy of our approach in detecting these attacks.

Acknowledgment

We thank Mr. Robert Szewczyk of University of the California at Berkeley for giving access to the data of Great Duck Island. We also thank ARC Research Network on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), and DEST International Science and Linkage Grant. This work was supported by the Australian Research Council (ARC).

6. REFERENCES

[1] A.P.da Silva et al., “Decentralized intrusion detection in wireless sensor networks,” in *ACM international Workshop on Quality of Service and Security in Wireless and Mobile Networks*, 2005, pp. 16–23.

[2] A.Perrig, J.Stankovic, and D.Wagner, “Security in wireless sensor networks,” in *CACM*, June 2004, vol. 47, pp. 53–57.

[3] C.E.Loo, M.Y.Ng, C.Leckie, and M.Palaniswami, “Intrusion detection for sensor networks,” in *International journal of distributed sensor networks - Accepted for publication*, 2006.

[4] S.Bandyopadhyay et al., “Clustering distributed data streams in peer-to-peer environments,” in *Information Sciences*. 2006, vol. 176 of *14*, pp. 1952–1985, Elsevier.

[5] I.Onat and A.Miri, “An intrusion detection system for wireless sensor networks,” in *Wireless And Mobile Computing Networking And Communications*, August 2005, vol. 3, pp. 253–259.

[6] R.Szewczyk, A.Mainwaring, J.Polastre, J.Anderson, and D.Culler, “An analysis of a large scale habitat monitoring application,” in *International conference on Embedded networked sensor systems*. 2004, pp. 214–226, ACM Press.

[7] V.Barnett and T.Lewis, *Outliers in Statistical Data*, John Wiley and Sons, 3rd edition, 1994.

[8] J.Han and M.Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2001.

[9] E.Eskin, A.Arnold, M.Prerau, L.Portnoy, and S.Stolfo, “A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data,” in *Data Mining for Security Applications*. 2002, Kluwer.

[10] V.Raghunathan, C.Schurgers, S.Park, and M.B.Srivastava, “Energyaware wireless microsensor networks,” in *IEEE Signal Processing Magazine*, March 2002.

[11] T.Zhang, R.Ramakrishnan, and M.Livny, “Birch: A new data clustering algorithm and its applications,” in *Data Mining and Knowledge Discovery 1 (2)*, 1997.

[12] S.Theodoridis and K.Koutroumbas, *Pattern Recognition*, Academic press, 2nd edition, 2003.

[13] S.Ramaswamy, R.Rastogi, and K.Shim, “Efficient algorithms for mining outliers from large data sets,” in *ACM SIGMOD ’00*. 2000, pp. 427–438, ACM Press.

[14] P.K.Chan, M.V.Mahoney, and M.H.Arshad, “Learning rules and clusters for anomaly detection in network traffic,” in *Managing Cyber Threats: Issues, Approaches and Challenges*. 2005, pp. 81–99, Springer.

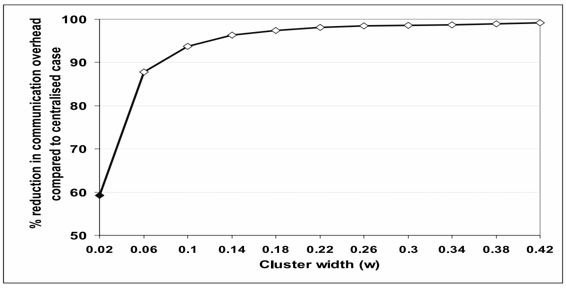

Fig. 5. % Reduction in the communication overhead in the network Vs Cluster width (w). The reduction in the communication overhead in the network is calculated as (the number of data vectors transmitted in the centralised case - the number of clusters transmitted in the distributed case) / (the number of data vectors transmitted in the centralised case).