Energy efficient data gathering using prediction-based filtering in wireless sensor networks

Govind Gupta* and Manoj Misra
Department of Electronics and Computer Engineering,
Indian Institute of Technology,
Roorkee, 247667, India
E-mail: gpg.india@gmail.com
E-mail: manojfec@iitr.ernet.in
*Corresponding author

Kumkum Garg
Department of Computer Science and Engineering,
Manipal University,
Jaipur, 302026, India
E-mail: kgargfec@gmail.com

Abstract: In wireless sensor network, sensor readings generated by nearby nodes are redundant and highly correlated, both in space and time domains. Since transmitting redundant and highly correlated data incurs a huge waste of energy and bandwidth, spatial and temporal correlation should be exploited in order to reduce redundant data transmission. In this paper, we propose an energy efficient data gathering protocol that uses a prediction-based filtering (EEDGPF) mechanism to solve the problem of redundant data transmissions. Our data gathering protocol organises a WSN into clusters, using data similarity that exists in readings of sensor nodes and cluster heads and uses a GARCH (1, 1) model-based non-linear predictor to exploit the temporal correlation of sensor readings. Experimental results over real dataset show that our protocol significantly outperforms linear predictor (AR(3))-based protocol proposed in Jiang et al. (2011), in terms of number of data packets delivered, number of successful predictions and average energy consumption.

Keywords: wireless sensor networks; WSNs; data filtering, predictor; spatial clustering.


Biographical notes: Govind Gupta received his Bachelor’s degree in Computer Science and Engineering in 2000 from CCS Meerut University, Meerut, India, and Master’s in Computer Science and Engineering in 2007 from UP Technical University, Lucknow, India. He is currently a PhD candidate in the Department of Electronics and Computer Engineering at Indian Institute of Technology, Roorkee, India. His research interests include mobile computing, wireless ad hoc and sensor network, network security, trust, distributed computing and performance evaluation. He is a student member of IEEE.

Copyright © 2013 Inderscience Enterprises Ltd.
Manoj Misra received his Bachelor’s degree in Electrical Engineering in 1983 from HBTI Kanpur, India and Master’s in Computer Science in 1986 from University of Roorkee, India. He received his PhD in Computer Science in 1997 from University of Newcastle, Upon Tyne, UK. He is currently a Professor in the Department of Computer Science and Engineering at Indian Institute of Technology, Roorkee. He has guided several PhD theses, ME/MTech dissertations. His areas of interest include mobile computing, distributed computing, wireless ad hoc and sensor network and performance evaluation. He is a senior member of IEEE.

Kumkum Garg received her Bachelor’s in Electronics Engineering from University of Roorkee, India in 1971 and Master’s in Computer Engineering in 1977 from University of Roorkee, India. She received her PhD in Computer Engineering in 1984 from University of London, UK. She has been a Professor in the Department of Computer Science and Engineering, Indian Institute of Technology, and Roorkee, India from 1989 to 2010. She has been the Head of Information superhighway Centre from 2005 to 2006 at IIT, Roorkee. Currently, she is Professor of Computing and Dean, Faculty of Engineering at Manipal University Jaipur, India. She has guided several PhD theses, ME/MTech dissertations and completed various projects. Her research interests include mobile computing, mobile agents, network security, trust, wireless ad hoc and sensor network. She is a senior member of the IEEE Computer Society.

1 Introduction

In a typical wireless sensor network (WSNs), sensor nodes (SNs) are equipped with limited resources, such as battery power, memory, computational capacity and network bandwidth and these are generally densely deployed for better connectivity and fault tolerance. Due to high density of deployment, sensor readings generated from adjacent nodes are spatially correlated (i.e., they are almost same) (Akyildiz et al., 2004). Further, sensor readings generated by the SN itself over a short period of time are also temporally correlated. Transmitting redundant and highly correlated data incurs a huge waste of energy resources. By exploiting spatial and temporal correlation of observed data, redundant data transmissions can be reduced, resulting in energy savings, so as to prolong network lifetime (Akyildiz et al., 2004; Vuran et al., 2004).

In order to exploit spatial correlation, spatial correlation-based clustering algorithm is proposed in Meka and Singh (2006), and Gedik et al. (2007). Here, the authors assume that all nodes are equipped with the same amount of battery power and since a cluster head (CH) acts as a sensor, aggregator and router, the battery power of CHs is exhausted earlier than SNs, thereby reducing network lifetime. In order to solve this problem, we assume that CHs are equipped with additional energy and memory resources. Our proposed spatial correlation-based clustering protocol organises SNs around CHs in a cluster, such that the mean value of a set of sensor readings at both CH and SN is within a specified threshold (τ) as shown in equation (1).
Energy efficient data gathering using prediction-based filtering in WSNs

\[ |\text{meanCH}_i - \text{meanSN}_j| < \tau \] (1)

where \( \text{meanCH}_i \) denotes the mean of past \( n \) sample values at CH \( i \) and \( \text{meanSN}_j \) denotes the mean of past \( n \) sample values at SN \( j \).

Time series forecasting techniques have been proposed in order to exploit temporal correlation (Tulone and Madden, 2006; Li et al., 2009; Jiang et al., 2011) in WSN. In these techniques, each node uses a predictor based on past sensor readings to perform prediction operations (Tulone and Madden, 2006). The predictor prevents the transmission of a sensor reading that differs from a predicted value by less than a prespecified error bound (Li et al., 2009). To the best of authors’ knowledge, in all proposals for WSN, time series of sensor readings are assumed to be stationary and linear (Tulone and Madden, 2006; Li et al., 2009; Jiang et al., 2011) and a linear time series model such as AR, ARMA or ARIMA is used for time series forecasting. But time series generated by SNs during observation of environmental phenomenon are generally non-stationary and non-linear (Mustafaraj et al., 2011; Min and Chung, 2010) and a linear predictor cannot predict non-stationary and non-linear time series with high accuracy. To overcome this problem, we designed a predictor based on a non-linear time series model, to predict sensor readings for reducing redundant data transmission. In order to compare the prediction accuracy of linear and non-linear predictor, we performed experiments with Intel sensor dataset (http://db.lcs.mit.edu/labdata/labdata.html) and the results are shown in Figure 1. It is evident from Figure 1 that prediction accuracy of non-linear predictor $GARCH(1, 1)$ is better than linear predictor AR(3).

In this research work, we focus on the problem of redundant data transmission by exploiting spatial and temporal correlation. In order to exploit spatial correlation, we propose a spatial correlation-based distributed clustering algorithm, where SNs around the CH are organised into clusters. In order to exploit temporal correlation, we have used $GARCH(1, 1)$ model (Hansen and Lunde, 2005; Engle, 2001) based predictor, both at CHs to predict the future sensor readings of individual cluster members and at SNs to check when they generate readings that differ by more than a prespecified error bound.

The main contributions of this paper are summarised as follows:

- Use of $GARCH(1, 1)$ non-linear time series model for modelling and forecasting the time series of sampled data.
- An improved spatial correlation-based distributed clustering protocol (SCDCP) for exploiting spatial correlation of sampled data.
- Comparison of linear and non-linear model-based predictors on Intel sensor dataset (http://db.lcs.mit.edu/labdata/labdata.html).

The rest of this paper is organised as follows: In the next section, we briefly describe related work. In Section 3, we discuss the system model and assumptions made. Section 4 presents the time series prediction model. The proposed protocol is given in Section 5. Section 6 gives the performance evaluation. Finally, Section 7 concludes this paper.
2 Related work

Much research work has been done on energy conservation to prolong sensor network lifetime. One class of approaches for energy conservation takes advantage of spatial and temporal correlations that exist in WSNs readings. In this section, we review existing research work that exploits spatial and temporal correlation especially prediction-based temporal correlation in order to reduce transmission of redundant sensor readings.

Tulone and Madden (2006) proposed a probabilistic adaptive query (PAQ) system which used autoregressive (AR) models at each SN to predict sensor readings. All SNs need to transmit their local model parameters to a sink node, which uses them to predict sensor readings without directly communicating with them. When a predicted value lies within a prespecified error bound, the model is believed to be appropriate; otherwise the node may mark the readings as outliers. Alternately, the time series model is marked as invalid and the relearning process started. Nodes send their model parameters to the sink. In addition to the prediction model, a clustering procedure is proposed to form a cluster of nodes based on similarity in sensor readings.

Liu et al. (2007) proposed an energy efficient data collection (EEDC) framework which exploits spatiotemporal correlation that exist in sampled data, to cope with redundant data transmission. In this framework, the sink node executes a clustering algorithm to organise the SNs into clusters and dynamically maintains clusters in response to change in spatial correlation. This work assumes that all SNs can directly communicate with the sink in a single hop, which is an impractical assumption in large-scale networks. The drawback of this clustering algorithm is that it is executed at the sink, which requires the sink to collect and store time series data from all SNs. This process incurs much communication cost.
Gedik et al. (2007) proposed an adaptive sampling approach-based data collection (ASAP) protocol. ASAP uses a dynamically changing subset of nodes as sampler nodes for directly collecting readings from them. It uses a local and periodically constructed probabilistic model to predict the value of the non-sampler nodes. Only sampler nodes within each subcluster can sense environmental data and send them to the base station. ASAP collects sensor readings from only sampler nodes and these are predicted using a multivariate normal (MVN) probabilistic model. ASAP achieves energy saving by allowing data collection services from sampler nodes only. The sink node derives readings of non-sampler nodes by using a prediction model. ASAP also proposes a spatial correlation-based clustering protocol. The drawback of the algorithm is that CH selection is based on probability, which exhausts more energy of the nodes during cluster construction.

Min and Chung (2010) proposed an approximate data gathering technique called EDGES, which utilises the multiple model Kalman filter as an approximation approach to exploit temporal correlation. It also exploits spatial correlation by organising spatially close SNs into clusters to extend the network lifetime. The EDGES uses a localised technique to notify the change of CH to small portion of a network. Pham et al. (2010) proposed a distributed clustering algorithm based on spatial correlations among SNs. In addition to clustering, a linear approximation scheme-based compressing algorithm is also proposed to exploit temporal correlation for reducing the volume of data required to transmit.

Jiang et al. (2011) proposed an energy efficient framework for cluster-based data collection, which integrates clustering and prediction techniques to achieve energy efficiency and stability. This work qualitatively derives a sufficient condition to determine whether a prediction technique is helpful or not and provides an adaptive scheme to enable or disable the predictor. This paper used a simple linear third order autoregressive (AR(3)) predictor to capture the temporal correlation. The disadvantage of this work is that it cannot capture non-linear patterns of time series samples at SNs.

All of these protocols use a linear model-based predictor for predicting the next sensor reading. The main limitation of such a predictor is that it cannot capture non-linear patterns of time series data. For this reason, we have proposed a cost effective non-linear (GARCH (1, 1)) model-based predictor for predicting the next sensor reading with higher accuracy.

3 Network model

We consider a typical WSN which has two types of nodes: SNs and CHs as shown in Figure 2. We assume that there are \( N \) nodes, which are uniformly deployed within a two-dimensional region. We organised the network into a clustering hierarchy where CH is more powerful than SN in terms of energy and memory as considered in Smaragdakis et al. (2004) and Qing et al. (2006). We consider the heterogeneous WSNs as discussed in Smaragdakis et al. (2004) where \( m \) is the fraction of the CHs, which have \( x \) times more energy and memory than the normal SNs. Thus, there is \((1 - m)N\) normal SNs equipped with initial energy of \( E_0 \) and memory capacity of \( M_0 \), and \( mN \) CH nodes equipped with initial energy of \( xE_0 \) and memory capacity of \( xM_0 \). Each SN as well as CH has the same sensing and radio transmission range. There is a static data collection unit called sink...
node, placed at the centre of the area for collecting data from CH. We also assume node density is sufficiently high for successful end-to-end data delivery.

Figure 2  Network model (see online version for colours)

4 Time series prediction model

In this section, we briefly describe autoregressive (AR) and the GARCH models based prediction operation and perform some tests to show the suitability of GARCH model for prediction of sensor data.

4.1 Autoregressive model

Auto-regressive (AR(p)) model is time series model where value at time $t$ is a linear combination of past $p$ values together with a residual error term. Term $p$ is the order of the model. It is written as:

$$x_t = c + w_t + \sum_{i=1}^{p} \phi_i x_{t-i}$$

(2)

Here, $\Phi_1, \Phi_2, \ldots, \Phi_p$ are the parameters, $c$ is a constant always presumed to be zero and $w_t$ is a residual error at time $t$. 
A third order autoregressive model (AR(3)) can be expressed as,

\[ x_t = c + w_t + \Phi_1 x_{t-1} + \Phi_2 x_{t-2} + \Phi_3 x_{t-3} \]  

(3)

The values of residual error \( w_t \) term are uncorrelated white noise process from a fixed distribution. For simplicity, it is assumed to be identically distributed with mean zero and variance one (Pourahmadi, 2001). The value of model parameters can be calculated using Yule-Walker equations as discussed in Pourahmadi (2001). After estimation of model parameters, we can calculate the predicted values by using equation (3). This model is homoskedastic, i.e., it assumes a constant volatility. AR(p) model is normally used for forecasting a stationary and linear time series. However, most real world phenomena observed by SNs are appeared to be non-linear and non-stationary. If time series are not stationary and show non-linearity, i.e., heteroscedasticity which can be tested by some standard test such as Ljung-Box-Pierce Q-test and Engle ARCH test. If time series is non-stationary, the statistical properties such as mean and variance are not constant. We performed these test on Intel sensor dataset (http://db.lcs.mit.edu/labdata/labdata.html) and observed that time series of this dataset show non-linearity. Due to this reason, we use a non-linear model GARCH (1, 1).

4.2 GARCH model

The GARCH (Bollerslev, 1986) model is a non-linear time series model. The name GARCH stands for generalised autoregressive conditional heteroskedasticity. The term autoregressive describes a feedback mechanism that incorporates past observation into present. Conditional implies a dependence on the observations of the immediate past. Heteroskedasticity refers to the time-varying variance or volatility of the time series. The GARCH model (http://math.bu.edu/misc/DOCSERVER/raw/garch.pdf) of order p and q is denoted as GARCH (p, q) and is defined as:

\[ X_t = \sigma_t e_t \]  

(4)

\[ \sigma_t^2 = \alpha_0 + \sum_{j=1}^{q} \alpha_j X_{t-j}^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2 \]  

(5)

Here, \( p \) refers to how many autoregressive lags appear in equation (5) and \( q \) refers to how many moving average lags are specified. A stochastic process \( \{X_t\} \) defined by the equation (4) is called a GARCH (p, q) process. Here, \( \alpha_0 \geq 0, \alpha_i \geq 0 \) and \( \beta_j \geq 0 \) are constants with constraint \( \sum_{j=1}^{p} \alpha_j + \sum_{j=1}^{q} \beta_j < 1 \). \( e_t \) is Gaussian white noise which is independent and identically distributed (i.i.d.) with mean 0 and variance 1, and independent of \( \{X_{t-k}, k \geq 1\} \) for all \( t \).

In our time series model, we have used a simple GARCH (1, 1) model to capture the variability in time series due to its low computational complexity. For GARCH (1, 1) model, conditional variance \( \sigma_t^2 \) can be expressed as,

\[ \sigma_t^2 = \alpha_0 + \alpha_1 X_{t-1}^2 + \beta_1 \sigma_{t-1}^2 \]  

(6)
where it is required that $\alpha_1 + \beta_1 < 1$, and the effect of volatility is longer if value of $\alpha_1 + \beta_1$ is closer to 1. The method used for estimation of the GARCH (1, 1) parameters is given in Pourahmadi (2001), Bollerslev (1986) and Zeng et al. (2008). After the parameters are computed, we can predict the future volatility of the time series (Ranjan et al., 2010). The predicted volatility (PV) (Ranjan et al., 2010) can be expressed as:

$$ PV_t = \sigma_t = \sqrt{\alpha_0 + \alpha_1 X_{t-1}^2 + \beta_1 \sigma_{t-1}^2} $$

(7)

The realised volatility (RV) (Ranjan et al., 2010) is expressed as,

$$ RV_t = x_t - X $$

(8)

Here, $x_t$ is the series value at time $t$ and $X$ is the mean of past sample values taken over long period.

The error between the actual RV ($RV_t$) and the PV ($PV_t$) is written as (Ranjan et al., 2010):

$$ e_t = (RV_t - PV_t) / RV_t $$

(9)

By using equation (9), each SN calculates error $e_t$ in prediction and if $e_t$ is greater than predefined error bound, SN transmit the sensed value to CH otherwise predicted value is used by CH.

4.3 Need for a non-linear model

One of the important issues in time series analysis is how to select an appropriate model for modelling the time series. Since the time series may be linear or non-linear, it is crucial to opt for an adequate model class and to estimate parameters included in the model. Parameter estimation depends on the features of the time series and the purpose of the time series analysis. There are two main set of models, namely linear and non-linear. Due to low computational complexity, the linear model is used in many studies to predict sensor readings (Tulone and Madden, 2006; Li et al., 2009; Jiang et al., 2011). However, linear models cannot capture non-linear patterns. This is because in linear time series modelling, it is assumed that the difference between actual and predicted values follows a Gaussian white noise distribution with mean zero and constant variance (Bollerslev, 1986). To prove the applicability of the GARCH model, we apply standard tests to check the heteroskedasticity in Intel sensor dataset (http://db.lcs.mit.edu/labdata/labdata.html) as given in Section 4.3.1.

4.3.1 Tests for heteroskedasticity

In this section, we discuss some standard tests on the Intel sensor dataset (http://db.lcs.mit.edu/labdata/labdata.html) to prove the applicability of the GARCH model for time series prediction. In order to build a GARCH non-linear time series model, we have to first test for conditional heteroskedasticity, also known as ARCH effect. This is done using Engel’s ARCH test and the Ljung-Box-Pierce Q-test. These
two tests show that there is sufficient evidence to confirm that sensor data is heteroskedastic in nature. Due to this, we have selected GARCH as the model of choice. We performed Engel’s ARCH test on Intel sensor dataset (http://db.lcs.mit.edu/labdata/labdata.html) using lags of 10, 15 and 20. Results are shown in Table 1. The value of $H = 1$ clearly point outs the existence of ARCH effects on the given dataset. We also performed the Ljung-Box-Pierce Q-test on the Intel sensor dataset (http://db.lcs.mit.edu/labdata/labdata.html) using lags of 10, 15 and 20. Table 2 shows the results, which indicate, similar to the Engel’s ARCH test, the value of $H = 1$ which clearly shows significant evidence in support of GARCH effects (heteroscedasticity) (http://math.bu.edu/misc/DOCSERVER/raw/garch.pdf).

<table>
<thead>
<tr>
<th>Lags</th>
<th>$H$</th>
<th>$p$-value</th>
<th>Stat</th>
<th>Critical value</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>1,305.3</td>
<td>18.3</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>0</td>
<td>1,300.3</td>
<td>25</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>0</td>
<td>1,295.4</td>
<td>31.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lags</th>
<th>$H$</th>
<th>$p$-value</th>
<th>Stat</th>
<th>Critical value</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>13,137</td>
<td>18</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>0</td>
<td>19,639</td>
<td>25</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>0</td>
<td>26,076</td>
<td>31</td>
</tr>
</tbody>
</table>

As shown in the previous analysis, the presence of heteroscedasticity indicates that GARCH model is appropriate. By observing the results of both tests, we can conclude that sensor dataset does indeed exhibit heteroskedasticity. Hence, it does not fit the normal distribution for the residual while modelling this dataset. Therefore, for such types of time series data, non-linear models like GARCH which take into account dependency of second order moments, are effective. This is the underlying reason for the use of GARCH model for prediction of such time series data.

### 5 Proposed protocol

In this section, we present an energy efficient data gathering protocol using prediction-based filtering (EEDGPF). EEDGPF contains two main components: spatial correlation-based clustering and prediction-based filtering.

#### 5.1 Spatial correlation-based clustering

The proposed protocol has three phases: initialisation phase, cluster construction phase and cluster joining phase. The data structures used in the algorithms are listed in Table 3.
Table 3  Data structures/variables used in processing

<table>
<thead>
<tr>
<th>Data structure/variables</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>nodeID</td>
<td>Identifier for node</td>
</tr>
<tr>
<td>Circular queue(Q)</td>
<td>For storing a set of latest sensor reading</td>
</tr>
<tr>
<td>hopCountToSink</td>
<td>Number of hops between node and Sink</td>
</tr>
<tr>
<td>hopCountToCH</td>
<td>Number of hops node away from cluster head</td>
</tr>
<tr>
<td>meanSNj</td>
<td>Mean of sensor readings at sensor node j</td>
</tr>
<tr>
<td>meanCH</td>
<td>Mean of sensor readings at cluster head</td>
</tr>
<tr>
<td>senderID</td>
<td>Id of node from which node receive packet</td>
</tr>
<tr>
<td>isGrouped</td>
<td>Flag variable which is true when node join</td>
</tr>
<tr>
<td>chID</td>
<td>ID of cluster head</td>
</tr>
<tr>
<td>chBelong</td>
<td>For storing chID to which node belong</td>
</tr>
<tr>
<td>$x_i$</td>
<td>Sensed value at node $i$</td>
</tr>
<tr>
<td>$x'_i$</td>
<td>Predicted value at node $i$</td>
</tr>
<tr>
<td>$\varepsilon$</td>
<td>Error bound</td>
</tr>
<tr>
<td>isCH</td>
<td>Flag variable which is true when node is CH</td>
</tr>
<tr>
<td>$t_{\text{reestimate}}$</td>
<td>Timer for re-estimation of model parameters</td>
</tr>
</tbody>
</table>

5.1.1 Initialisation phase

In the initialisation phase, we perform neighbour discovery of each node and establish a neighbour table. In the beginning, we assume that all nodes are infinite hops away from the sink, i.e., the hop count to sink ($\text{hopCountToSink}$) of each node is set to 1,000, a large number. The operation of neighbour discovery works as follows. After network deployment, the sink broadcasts a neighbour discovery packet (NDPkt) to all its 1-hop neighbour nodes. The NDPkt packet contains three fields: $\text{nodeID}$, $\text{nodeEnergy}$ and $\text{hopCount}$, where $\text{nodeID}$ is an identifier, $\text{nodeEnergy}$ is the remaining energy of sender node and $\text{hopCount}$ denotes the number of hops the sender node is away from the sink.

When the NDPkt packet starts from the sink, its $\text{hopCount}$ field is set to 0. The node receiving the NDPkt packet checks the condition as shown in Line 4 of Algorithm 1 below. If the condition is true, the node updates the value of $\text{hopCountToSink}$ and updates its neighbour table. The neighbour table at each node maintains three records: $\text{nodeID}$ of neighbour node from which it receives an NDPkt packet, remaining energy of neighbour node ($\text{nodeEnergy}$) and hop distance from neighbour node to sink ($\text{hopCountToSink}$).

After updating the neighbour table, the node updates the fields of the NDPkt packet by its $\text{nodeID}$, remaining energy and $\text{hopCountToSink}$ and broadcasts it to its one hop neighbours only once as shown in Lines 9 to 14 of Algorithm 1. The neighbour nodes which receive this packet repeat the same. This process continues till all nodes set their neighbour table entry accordingly by receiving the NDPkt packet at least once.

Each node can inform its current energy level to its 1-hop neighbours by exchanging beacon packets periodically. After completion of initialisation phase, each node knows the hop distance between itself and the sink and has established its neighbour table. The hop distance and nodeEnergy is used as routing criteria for forwarding the aggregated data through multi-hop routes from CH towards the sink.
Algorithm 1  Initialisation phase

For the sink:
begin
1:   hopCount=0;
2:   broadcast a neighbor discovery packet NDPkt (nodeID, nodeEnergy, hopCount)
   to its all one hop neighbors
end

For each node (i):
begin
1: Initially: setOnce = false;
2: hopCountToSink = 1000 (a large number equivalent to $\infty$);
3: if (received NDPkt packet p) then
4:   if (hopCountToSink > p.hopCount +1) then
5:      hopCountToSink = p.hopCount +1;
6: end if
7: update neighbor table (p.nodeID, p.nodeEnergy, p.hopCount);
8: end if
9: if (! setOnce) then
10:   setOnce=true;
11:  p.hopCount= hopCountToSink;
12:  p.nodeID = nodeID;
13:  p.nodeEnergy = nodeEnergy;
14:  broadcast NDPkt (p.nodeID, p.nodeEnergy, p.hopCount);
15: end if
end

5.1.2 Cluster construction and joining phase

After initialisation, each CH prepares a cluster construction packet (clusterCPkt) which contains five fields: CH identifier (chID), mean of sensor readings sampled at CH (meanCH), node id of sender (senderID), maximum number of hops the packet travels (maxHop) and hCount which gives the number of hops the packet has to travel in the network. It then broadcasts the packet to its one hop neighbours.

Algorithm 2  Cluster construction and joining process

For any cluster head (i)
begin
1: Calculate meanCH;
2: senderID = chID;
3: hCount = (hopCountToSink >=2)? 2:1;
4: maxHop = hCount;
end
G. Gupta et al.

5: Broadcast Cluster construction packet clusterCPkt (chID, meanCH, senderID, maxHop, hCount);
6: if (receive clusterJoinPkt and isCH) then
7: Store sending nodeID in its cluster member node list
8: end if

End

For any sensor node (i):
begin
1: Initialise: IsGrouped=false;
2: \( \tau = 0.2; \) //similarity threshold
3: hopCountToCH= 100; //let max distance
4: if (receive clusterCPkt q) then
5: temp = 1+ q.maxHop – q.hCount; // calculate hop distance of packet from CH
6: if (abs (q.meanCH – meanSNi) < \( \tau \) && temp < hopCountToCH) then
7: senderID = q.senderID;
8: chBelong = q.chID;
9: isGrouped=true;
10: hopCountToCH = temp;
11: q.hCount = q.hCount – 1;
12: if (q.hCount >0) then
13: q.senderID= nodeID;
14: Broadcast clusterCPkt (q.chBelong, q.meanCH, q.senderID, q.maxHop, q.hCount);
15: end if
16: end if
17: end if
18: if (isGrouped&& timer \(_t_{\text{clusterformation}}\) is over) then // cluster joining phase
19: Prepare cluster join packet clusterJoinPkt (nodeID, hopCount, chID, destination)
20: unicast to the node from which it received clusterCPkt
21: end if
22: if (receive clusterJoinPkt and !isCH) then
23: Forward the clusterJoinPkt to given CH
24: end if
25: if (! isGrouped&& \(_t_{\text{clusterformation}}\) is over) then // for nodes that are not in cluster
26: send their data packet to sink
27: end if
end

To balance energy consumption, we adopt the unequal clustering technique suggested in Chen et al. (2009) and Liu et al. (2010). Our protocol constructs 1-hop clusters closer to the sink and 2-hop clusters farther away from the sink. When a SN (SNi) receives the clusterCPkt packet, it checks the similarity condition shown in line 6 of Algorithm 2. If
Energy efficient data gathering using prediction-based filtering in WSNs

the similarity condition is satisfied by a SN, \( SN_i \), it updates senderID, chBelong, isGrouped and hopCountToCH as shown in lines 8–11 of Algorithm 2.

When timer \( t_{\text{clusterformation}} \) is over, each SN checks the condition specified in line 23 of Algorithm 2. If the condition is satisfied, each SN invokes the cluster joining process where each SN which is \( \text{isGrouped} \), sends a cluster join packet (\( \text{clusterJoinPkt} \)) to its CH to confirm joining the cluster. Lines 25 to 27 of Algorithm 2 are executed by a SN if it is not part of any cluster after timer \( t_{\text{clusterformation}} \) expires. In this case, the node sends its data packet to the sink using routing protocol proposed in Ye et al. (2001).

5.2 Prediction-based data filtering

In the EEDGPF protocol, each SN maintains a circular queue to keep a set of latest sensor readings. Each \( CH \) also maintains one circular queue per cluster member (\( CM \)) to keep a set of past sensor readings of the corresponding CM. In our experiment, we assume that the size of the circular queue is 25. After the cluster construction phase, each CM of a cluster (\( C \)) periodically sends its sensor reading to its CH. After filling the circular queues at CHs and CMs, each CH constructs the GARCH (1, 1) based time series model, using a set of past sensor readings, for capturing the correlation in sensed data. Next, each CH estimates the model parameters and unicasts these parameters to every CM. After receiving the model parameters, the predictor of each CM is able to predict the next value based on the set of past sensor readings.

The proposed protocol sets a periodic timer at each \( CH \) for re-estimation of GARCH (1, 1) parameters and forwards these parameters to its CMs. In our experiments, we reevaluate parameters of the GARCH (1, 1) model after 50 samples. Our approach is efficient in terms of energy because it updates parameters after 50 samples, whereas the AR(3) model proposed in Jiang et al. (2011), updates parameters after each sample, resulting in waste of energy in computation of parameters.

During each sampling period (\( t_{\text{sample}} \)), each cluster member senses the physical phenomenon from the environment. Let the sensed value be \( x_i \). Next, each cluster member calculates the predicted value say \( \hat{x}_i \) using GARCH (1, 1) model. If absolute prediction error \( |x_i - \hat{x}_i| \) is less than the error bound (\( \varepsilon \)), the sensed value is not sent to \( CH \) and the predicted value is stored in a circular queue (Jiang et al., 2011). Otherwise the sensed value is delivered to \( CH \) and stored in its circular queue. The pseudo code for prediction-based data filtering is given in Algorithm 3.

The main computational complexity of predictor algorithm is due to the calculation of the AR(3)/GARCH (1, 1) model parameters. In the AR(3) model, parameters are calculated using Yule-Walker equations (Pourahmadi, 2001). Whereas in case of GARCH (1, 1), the model parameters are normally calculated using an iterative method (Pourahmadi, 2001; Ranjan et al., 2010) and the complexity depends on number of iterations and generally more than AR(3) model. Because of this, we have not calculated values of the model parameters at SNs. We performed experiments with Intel sensor dataset (http://db.lcs.mit.edu/labdata/labdata.html) and observed that for a reasonable time, values of these parameters remain same. So SNs do not calculate these parameters but rather these parameters are calculated and broadcasted by CHs or base station to these SNs at regular intervals. This reduces the overall computational complexity of the predictor algorithm.
Algorithm 3 Prediction-based data filtering algorithm

For any cluster member (i):
begin
1: \( x_i = \text{sense}() \);
2: \( \hat{x}_i = \text{predict}() \);
3: if (abs \( (x_i - \hat{x}_i) < \varepsilon \))
4: store the predicted value to sensor reading queue;
5: else delivered the sensed value to its cluster head;
6: store the sensed value to sensor reading queue;
7: end if
end

For cluster head (j):
begin
1: if (receive a sensed value from cluster member i)
2: update the sensor reading queue for cluster member i ;
3: else
4: execute predictor to predict data value;
5: update the sensor reading queue for cluster member i by predicted value;
6: end if
8: if (parameter re-estimation timer \( t_{\text{reestimate}} \) expires)
9: for (each cluster member i) do
10: re-estimate model parameters ;
11: forward these parameter lists to cluster member i ;
12: end for
13: end if
end

6 Performance evaluation

In this section, we present the performance evaluation of our EEDGPF protocol and compare its performance with the AR(3) model-based protocol of Jiang et al. (2011). Both protocols are simulated using a Castalia version 3.2 (http://castalia.npc.nicta.com.au/) WSN simulator. To reflect the real application effects of WSNs, we have used the real world sensor dataset from Intel Berkeley Research lab (http://db.lcs.mit.edu/labdata/labdata.html). Our experiments demonstrate that the proposed EEDGPF data gathering protocol can notably reduce redundant data transmission, as discussed below.
6.1 Experiment setup

We conducted our experiments using real sensor dataset which was collected by the Intel Berkeley Research lab during one month period. This sensor dataset consist of environmental data such as temperature, humidity and voltage readings collected from 54 SNs deployed around their lab. We observed that this dataset has some missing data values for various nodes at different epochs due to unreliable wireless transmissions. We have adopted the same method used in Jiang et al. (2011) for filling the missing sensor data values with the average of the values during the previous and subsequent epochs at the same node. In our experiments, we assume that \( m = 0.2 \) the fraction of the CHs. We used the following metrics to evaluate the performance of our protocol:

- **Number of data packets** reported is the number of data packets transmitted by a node after prediction-based filtering.
- **Number of successful predictions** is defined as the number of predicted readings that are within the error bound.
- **Transmission ratio (TR)** is defined as ratio of the number of data packets transmitted with prediction-based filtering to the number of data packets transmitted without prediction-based filtering (Raza et al., 2012).
- **Suppression ratio (SR)** is calculated as \( 1 - TR \). This metric measures the number of data packets whose transmission cannot be filtered by the predictor (Raza et al., 2012).
- **Average energy consumption** is defined as the average amount of energy consumed by a SN.

6.2 Impact of error bound

In this experiment, we have used a topology similar to that of the sensor network deployed in the Intel Berkeley Research lab and used only 1,000 temperature records of each SN for performance analysis. The simulation parameters are listed in Table 4.

<table>
<thead>
<tr>
<th>Simulation parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sink</td>
<td>1 (placed at centre)</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>54</td>
</tr>
<tr>
<td>m</td>
<td>0.2</td>
</tr>
<tr>
<td>Sensor field</td>
<td>200 × 100</td>
</tr>
<tr>
<td>Communication radius</td>
<td>25 m</td>
</tr>
<tr>
<td>Initial battery power</td>
<td>18,720 J</td>
</tr>
<tr>
<td>Simulation time</td>
<td>6,000 sec</td>
</tr>
<tr>
<td>Sensing interval</td>
<td>5 sec</td>
</tr>
<tr>
<td>MAC</td>
<td>T-MAC</td>
</tr>
</tbody>
</table>
Figure 3(a) shows the impact of the error bound on the number of packets transmitted by a node. In this simulation scenario, the error bound was varied from 0.1 to 2.1, to evaluate the number of packets transmitted by a node. Simulation results reveal that the value of the error bound notably impacts the number of packets transmitted by a node to CH. As described in Section 5.2, the SN is not required to send the actual sensor reading to the CH when the condition $|x_i - x'_i| < \varepsilon$ is satisfied, thereby reducing redundant data transmissions and saving energy. As shown in Figure 3(a), when the error bound ($\varepsilon$) increases, the number of data packets reported to CH decreases. This is due to the fact that prediction accuracy of GARCH (1, 1) model is higher than that of AR(3).

Figure 3(b) shows the impact of the error bound on the number of successful predictions. The result shows that the GARCH (1, 1) model-based predictor has a much higher success rate of prediction than AR(3).

Figure 3 Performance with a varying error bound, (a) number of data packet (b) number of successful prediction (c) TR (d) SR (see online version for colours)
Figures 3(c) and 3(d) show the impact of the error bound on the TR and SR, respectively. It is evident from Figure 3(c) that TR of GARCH (1, 1) model-based predictor is lower than that of the AR(3) based predictor. From Figure 3(d), it is observed that GARCH (1, 1) model and AR(3) model-based prediction suppresses up to 98% and 44% of the data reports on an average, respectively. This suggests that GARCH (1, 1) model-based predictor performs better than AR(3) based prediction in reducing redundant data transmissions.

Figure 4 shows the impact of error bound on the average energy consumption. The error bound is again varied from 0.1 to 2.1 to evaluate the average energy consumption. From Figure 4, it is observed that average energy consumption decreases with increase in error bound. This is due to the fact that a higher error bound causes less data packets reported by nodes, thereby reducing average energy consumption.

**Figure 4  Average energy consumption vs. error bound (see online version for colours)**

6.3 Impact of network size

In order to study the scalability of proposed clustering protocol, we use the software toolkit provided in Jindal and Psounis (2004) to generate large trace of spatially correlated data. By using this toolkit, we extract the model parameters from Intel sensor datasets and generate large-scale synthetic datasets. Figure 5 shows the performance comparison of our proposed SCDCP with clustering algorithm (ASAP) proposed in Gedik et al. (2007). It is evident from Figure 5(a) is that SCDCP generates approximately 30% less control message compared to ASAP. This is due to the fact that CHs are selected randomly in ASAP. Figure 5(b) shows that the performance of SCDCP is better than ASAP in terms of total energy consumption. This is because of less number of control message transmissions in spatial clustering compared to ASAP.

Figure 6(a) shows the impact of number of nodes on the average energy consumption. The number of nodes was varied from 50 to 300, for the error bounds 0.5 and 0.9. It is observed from Figure 6(a) that the GARCH (1, 1) model-based protocol performs better than the AR(3) model-based protocol in terms of average energy consumption, because it provides higher prediction accuracy. With error bound $\varepsilon = 0.9$, AR(3) performs better
than with $\varepsilon = 0.5$. This is due to the fact that higher the error bound allows less data packet transmissions from SN to CH.

Figure 6(b) shows the average energy consumption for GARCH (1, 1) based prediction filtering protocol with different error bounds. It can be observed that average energy consumption decreases as error bound increases. This is because a higher error bound allows less data packet transmission from nodes, thereby decreasing communication cost, resulting in reduced average energy consumption. Thus, our GARCH (1, 1) model-based protocol is more scalable and energy efficient than the AR(3) model-based protocol.

Figure 5  Performance with a varying number of nodes, (a) number of cluster construction message (b) total energy consumption (see online version for colours)

Figure 6  Performance with a varying number of nodes, (a) average energy consumption (b) average energy consumption (see online version for colours)
7 Conclusions

We have presented an energy efficient data gathering protocol for WSNs which exploits spatial and temporal correlation of sensor readings in order to reduce redundant data transmissions, thereby reducing energy consumption. Spatial correlation is exploited by organising nodes around CHs in clusters, based on the similarity measure of sensor readings. However, temporal correlation is exploited by our GARCH (1, 1) model-based prediction approach that builds a more accurate model than existing approaches. The simulation results show that the proposed protocol effectively reduces redundant data transmission and provides an energy efficient solution for periodic data gathering applications.

References


