4th IEEE Conference on Automation Science and Engineering
Key Bridge Marriott, Washington DC, USA
August 23-26, 2008

Collaborative data reduction for energy efficient sensor networks

Chiwoo Park
Industrial and Systems Engineering
Texas A&M University
College Station, Texas 77843-3131
chiwoo.park@tamu.edu

Yu Ding
Industrial and Systems Engineering
Texas A&M University
College Station, Texas 77843-3131
yuding@iemail.tamu.edu

Eunshin Byon
Industrial and Systems Engineering
Texas A&M University
College Station, Texas 77843-3131
esbyon@tamu.edu

Abstract— **When we consider the limited power of wireless sensors, it is necessary to reduce the dimension of data conveyed between sensors, because high dimensional data transmission requires much power consumption of sensors. For data reduction in a network, in-network data aggregation methods and collaborative compression methods were reported. However, the in-network data aggregation methods typically lead to time-delay while performing data aggregation and compression, whereas the current collaborative methods primarily consider data redundancy between two sensors, not among multiple sensors. This paper proposes a new collaborative data reduction method to remove the redundancy existing in the data coming from mutiple sensors as well as from a single sensor. Our method uses a tree-based data propagation model to characterize the collaboration structure among multiple sensors. Our method also seperates the data aggregation process from the collaboration process in order to relieve the time-delay problem during aggregation processes. Thus, the time-delay is occured only during the periodic collaboration phase but not during the real-time data transmission. Our experimental results show that our method can reduce data transmission more effectively than the in-network aggregation without losing important information and without causing delay in aggregation.**

I. Introduction

Wireless sensor nodes are distributed over the area of interest and they form an autonomous network to gather important information. In many applications, the network shoud be able to operate for a long period of time without human's intervention. However, the power resource available on wireless sensors limits the operation time. Frequent power outages of sensors cause significant maintenance cost of battery replacement and increase down-time of the sensors. Therefore, implementing a power efficient sensor network is very important. According to an analysis on power consumption of wireless sensors [1], the sensors consume power mainly for transmitting data via ZigBee or Bluetooth and for processing data on their micro-CPUs. Especially, consumption per unit data transmission is about ten times as much as consumption per unit processing. Therefore, if we can reduce the dimension of data gathered by a sensing unit on a sensor's CPU, we can reduce much power consumption for data transmission at the expense of less power required for data processing on the CPU. For this reason, on-sensor data reduction is generally considered a good practice to increase the power efficiency.

However, if we mean to address data reduction in the entire network, doing solely the on-sensor reduction may not be sufficient. In a wireless network, sensors do not work only as information collectors, but also as intermediate nodes routing data from different sensors to a central repository. In this configuration, anchor nodes, functioning as hubs, receive a lot of routing packets from other sensors. Therefore, even though the size of the packet is reduced on each sensor, the anchor nodes may still need to transmit a lot of data accumulated from many sensors. Consequently, the anchor nodes consume their power fast and the whole network could lose its function.

A simple idea to relieve the problem is to apply data reduction on intermediate nodes before transmission. This technique is referred to as in-network data aggregation, also known as data fusion. The data aggregation techniques define a tree model of sensor nodes describing the hierarchical structure for data transmission. A central repository is modeled as a root node and the techniques aggregate data from leaf nodes to the root node; data reduction is applied to each node in the tree model. The techniques commonly involve a timing model, which decides how much time the intermediate nodes wait for to aggregate data from their child nodes. Arici et. al. [2] proposed PINCO (Pipelined In-Network COmpression) algorithm, in which an intermediate node stores data from different sensors for a specific time delay and reduces all the data accumulated for the delay of period. The timing model using such a predefined delay as PINCO is called Periodic Simple. Directed Diffusion [3] and LEACH [4] fall in the same category. Another type of the timing model is Periodic Per Hop. In this model, the intermediate nodes reduce and transmit data just after all data from child nodes arrive. TAG [5] and Convergecasting [6] are in this category.

In-network data aggregation reduces data transmission on all sensors in a network, but the time delay required for data aggregation causes latency of transmission. If the latency is large, urgent data would not be avaliable by time for use. Another problem is its inefficiency in terms of data reduction capability on a leaf sensor node. Data from leaf nodes are collected to an intermediate node and the intermediate node further reduces the data, which means there are still redundancy among the data from the leaf nodes. This redundancy is the between-sensor redundancy, which is difficult to be removed on individual sensors.

Between-sensor redundancy is mainly caused by spatial

978-1-4244-2023-0/08/$25.00 ©2008 IEEE.

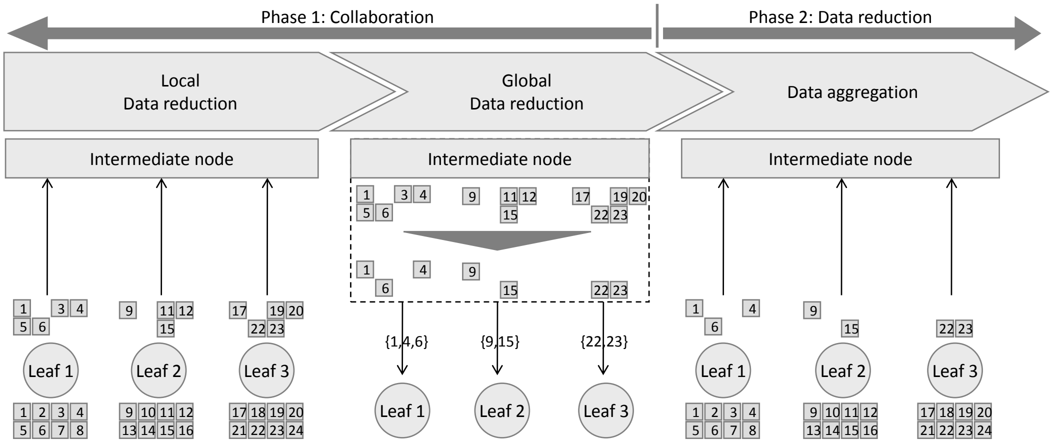

Fig. 1. Overall process of our method

proximity; sensors at proximity generates highly correlated data. If we can remove the between-sensor redundancy at leaf nodes, we get two benefits: 1) leaf nodes can reduce more dimension and transmit less data, 2) intermediate nodes just receive data from child nodes without performing another data reduction. The second benefit leads to saving the power consumption at the intermediate nodes as well as alleviating data latency. Our research is looking for a new method that can reduce both between-sensor redundancy and within-sensor redundancy on leaf nodes.

To remove between-sensor redundancy, each leaf node should know about data from the other sensors. In this context, collaborative compression algorithms are proposed to remove both types of the redundancies on leaf nodes. Collaboration involves communication between sensors and it consumes energy. Thus, collaborative compression methods need to strike a balance between power consumption by collaboration and data reduction performance. Hoang et al. [7] developed collaborative compression algorithm using a broadcating feature of wireless communication. When a sensor transmits data to a target sensor, the sensor broadcasts the data, not sending it only to the target sensor. Thus, all neighboring nodes within the radio connectivity of the sender can receive the data. However, the sensors except the target sensor usually discard the data. Hoang et al. [7] utilized the broadcasting feature to share data among neighboring nodes. Therefore, a sensor can identify the correlation of its data with its neighboring sensors' data without any additional collaboration efforts. Its limitation is that only the data redundancy within a single-hop distance can be removed.

Pradhan [8] proposed a distributed coding algorithm on the framework of Slepian.Wolf source coding theorem. By the theorem, a sensor can obtain as high compression rate as it knows of the data from its neighbors, even if the sensors cannot communicate with each other. As such, the method ideally does not require any collaboration. However, the coding algorithm was applied only to binary relations involving two sensors. Additional research is necessary to extend the sensor collaboration to multiple-sensor environments. In addition, the algorithm assumes that a codebook needed to compress data is known for every sensor in a network, but making a good codebook requires collaboration between sensors and may not be easy to create.

In this paper, we propose a method to reduce the between-sensor redundancy as well as the within-sensor redundancy through collaboration between an intermediate node and its child sensor nodes. Our method is a general data reduction that removes redundancy among multiple sensors, not necessarily limited to binary relations as the previous distributed compression algorithms did. It is also different from the in-network aggregation method in the sense that it removes both the between-sensor redundancy and the within-sensor redundancy on leaf nodes. The main idea and our contribution are as follows:

- **Collaborative data reduction:** our method is essentially a collaborative data reduction method. Different from in-network aggregation, each sensor can reduce both between-sensor redundancy and within-sensor redundancy on itself through collaboration with other sensors. There were two collaboration models: a peer-to-peer(P2P) model like collaborative compression and a tree-based model. A P2P model consists of pair-wise collaboration, so the number of pairs increases exponentially as the number of sensors increases. Thus, we choose a tree-based collaboration model where sensors cooperate through their common parent nodes to reduce data transmission.
- **Separating collaboration from data reduction:** For the exisitng collaborative compressions, collaboration among sensors occurs every time data reduction occurs. It means that energy required for collaboration is consumed frequently. Assuming that the between-sensor redundancy due to spatial proximity does not change very frequently, we seperate the collaboration processes from the data reduction prcesses. In our method, the data reduction processes are frequent events, while the collaboration processes are periodic yet much less frequent. Through the periodic collaboration, our method

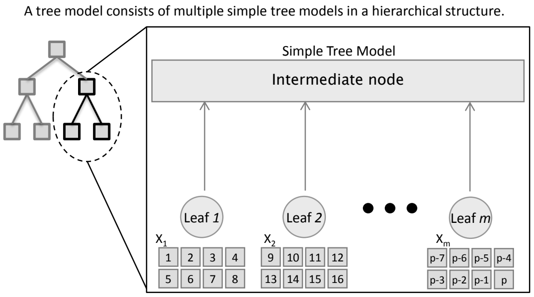

Fig. 2. Simple network collaboration

defines the minimally required data components to be transmitted from each sensor. In the data reduction processes, sensors just follow the pre-defined reduction schemes to fulfill a real-time data reduction. Consequently, data transmission required for collaboration decreases. Moreover, there is no time-delay in the data reduction processes.

We start with Section II by describing our methods. In Section III, an experimental result will be provided to show the overall performance of our method. Finally, we conclude the paper in Section IV.

II. COLLABORATIVE DATA REDUCTION METHOD

Our method has two phases: a collaboration phase and a data reduction phase as shown in Fig. 1. In the collaboration phase, each sensor identifies which part of its data is essential or non-redundant when it considers the other sensor's data. An intermediate node will take the data from the leaf nodes and perform a global data reduction. The parts result in the minimal set of data parts that need to be transmitted. These data parts are labeled by a set of indices. A sensor keeps the indices of the parts in their memory until the next collaboration phase. In the data reduction phase, the sensors simply send over the parts of their data, corresponding to the indices. More details for both phases are described in the subsequent sections.

A. Collaboration phase

Suppose that a sensor network has a tree model through which sensors transmit data to a central repository, where collaboration works between an intermediate node and its child nodes to remove data redundancy. Now, we explain our method in a simple network (Fig. 2), consisting of one intermediate node and m sensors. Since the network consists of multiple simple network-based collabrations in a hierarchical structure, the case can be easily extended to a general tree model.

For data reduction, suppose that we have n observations from m sensors. We denote the ith observation of the jth sensor by $\underline{x}_{ij}$ and let X_j be a set of all the observations of the jth sensor as:

$$X_j = \{\underline{x}_{1j}, \underline{x}_{2j}, \ldots, \underline{x}_{nj}\}, j = 1, 2, .., m \tag{1}$$

where $\underline{x}_{ij} = (x_{ij1}, \ldots, x_{ijp_j})$ is a $1 \times p_j$ vector.

The collaboration phase consists of the two steps as Fig. 1. First, the j^{th} sensor collects data $X_j = \{\underline{x}_{ij} : i = 1, 2, \ldots, n\}$ and reduces it to Y_j by certain data reduction techniques. Next, the j^{th} sensor sends Y_j to the intermediate node. The intermediate node finds the redundant columns by analyzing $\{Y_j, j = 1, 2, ..m\}$ and removes them so that it produces an index set of the non-redundant data. The intermediate node will then notify the correspoding leaf nodes about this index set.

1) Local data reduction: The objective of this step is to reduce local data redundancy as much as possible under the limitation of computational capability of each sensor. In this step, each node has access to all the data it collects but not the data from its neighbors. In our procedure, we used the Principal Component Analysis (PCA) since it is a very efficient reduction technique and simple enough to be implemented on sensors with limited computing power. In real practice, to reduce computational and memory burden, we can use the incremental PCA [9]. The computational complexity of PCA is known as $O(p^3)$ or $O(p^2)$ [10].

We start with the jth sensor data, $X_j = \{\underline{x}_{ij} : i = 1, .., n\}$. We would like to find a projection matrix P_j that projects p_j dimensional data onto q_j dimensional space ($q_j < p_j$) as

$$\underline{y}_{ij} = \underline{x}_{ij} P_j \tag{2}$$

We would like to make q_j as small as possible, but $\underline{y}_{ij}$ should capture the pattern of the original dataset. The pattern is described by the covariance matrix of X_j. Thus, our objective is to reduce the dimension of the data while retaining as much as possible variation present in the original dataset.

Given the n observations in X_j, define a data matrix $\mathcal{X}_j$ with $\underline{x}_{ij}$ as its ith row. Let Σ_j be the sample covariance matrix of $\mathcal{X}_j$. Let $\mathcal{Y}_j$ be the projection of $\mathcal{X}_j$ by P_j. Then, the sample covariance of $\mathcal{Y}_j$, $\Sigma_{\mathcal{Y}_j}$, is

$$\begin{aligned}
\Sigma_{\mathcal{Y}_j} &= \frac{1}{n - q_j}(\mathcal{Y}_j - \overline{\mathcal{Y}}_j)'(\mathcal{Y}_j - \overline{\mathcal{Y}}_j) \\
&= \frac{1}{n - q_j} P_j'(\mathcal{X}_j - \overline{\mathcal{X}}_j)'(\mathcal{X}_j - \overline{\mathcal{X}}_j) P_j \\
&= \frac{n - p_j}{n - q_j} P_j' \Sigma_j P_j
\end{aligned} \tag{3}$$

Then, we would like to find P_j^* to maximize $tr(\Sigma_{\mathcal{Y}_j})$ with smaller dimensionality as:

$$\begin{aligned}
P_j^* = \arg \max_{\text{all } P_j} \quad & tr(P_j' \Sigma_j P_j), \\
\text{s.t.} \quad & q_j \leq UB_q
\end{aligned}$$

where q_j is the number of the columns that P_j has and UB_q is the upper bound of q_j. That is, we do not want the reduced dimension, namely q_j, to exceed UB_q.

Since P_j is a projection matrix, $tr(P_j' \Sigma_j P_j) = tr(\Sigma_j)$, and $tr(\Sigma_j)$ is equivalent to the total sum of the eigen-values of Σ_j. Thus, the problem is to find P_j^* so that the UB_q principal components of Σ_j forms its column vectors.

The appropriate UB_q value is determined by a well known simple rule [14]:

$$UB_q = \min\{s : \sum_{k=1}^{s} \lambda_{(k)} \geq r\}, \tag{4}$$

where $\lambda_{(k)}$ is the kth biggest eigen-value of Σ_j and r corresponds to the ratio of the variance explained by s leading eigen-values and the total variance.

To sum up, the local projection matrix for the jth sensor, P_j^* is formed by the UB_q principal components of Σ_j. $\mathcal{X}_j$ $(n \times p_j)$ is reduced to $\mathcal{Y}_j$ $(n \times q_j)$ by P^* and the jth sensor sends $\mathcal{Y}_j$ to a central repository.

$$\mathcal{Y}_j = \mathcal{X}_j P_j^* \tag{5}$$

For better understanding, look at the example under the local data reduction step of Fig. 1. The leaf 1 originally has data $\{1, 2, 3, 4, 5, 6, 7, 8\}$ that corresponds to $\mathcal{X}_j$ here. The data is reduced to the parts $\{1, 3, 4, 5, 6\}$ that corresponds to $\mathcal{Y}_j$.

2) Between-sensor redundancy reduction: After the local data reduction, a central repository has the reduced dataset $\{\mathcal{Y}_j, j = 1, 2, .., m\}$ from m sensors. The objective of this step is to drop some parts of $\{\mathcal{Y}_j, j = 1, 2, .., m\}$, which may be highly redundant to the data from other sensors. For that, we cannot use factorial data reduction techniques or orthogonal transformation methods such as PCA and wavelet shrinkage. That's because those methods use projection matrices described in the local reduction step but data reduction using the projection matrices requires availability of all the original variables. At the data reduction phase after the collaboration phase, we do not have all the original variables on an intermediate node. Thus, we need to identify the parts of $\mathcal{Y}$ with less redundancy without using any projections.

This become a subset selection problem. Regression methods such as ridge regression and LASSO regression can be considered. However, in this case we have only a data matrix without a dependent variable for regression. Cumming et al. [11] proposed an iterative subset selection algorithm based on partial correlation which requires a data matrix only. We adopted his algorithm and modified its objective function by adding a penalty term related to the data reduction rate in order to achieve more aggressive data reduction. The stopping rule for the algorithm was also modified in accordance with the new objective function.

Let $\mathcal{Y} = [\mathcal{Y}_1, \mathcal{Y}_2, .., \mathcal{Y}_m]$, which is a column-concatenated matrix of the reduce datasets from the m sensors. Then, it is an $n \times q$ matrix where $q = \sum_{j=1}^{m} q_j$. Suppose that we partition $\mathcal{Y}$ into S and T as:

$$\mathcal{Y} = [T|S] \tag{6}$$

where S is a subset of $\mathcal{Y}$ to be retained and T is a subset of $\mathcal{Y}$ to be eliminated due to its high redundancy of S.

The sample partial covariance of T given S is a measure of the covariance structure that cannot be captured by S [11]. Denote the partial covariance matrix by $\Sigma_{T|S}$. The matrix is defined as [12]:

$$\Sigma_{T|S} = \Sigma_{TT} - \Sigma_{TS}\Sigma_{SS}^{-1}\Sigma_{ST} \tag{7}$$

where Σ_{TT}, Σ_{TS}, Σ_{SS} and Σ_{ST} are parts of the covariance matrix generated by $\mathcal{Y}$ as

$$\Sigma_{\mathcal{Y}} = \begin{bmatrix} \Sigma_{TT} & \Sigma_{TS} \\ \Sigma_{ST} & \Sigma_{SS} \end{bmatrix} \tag{8}$$

We want to minimize the covariance structure not captured by S. That is, the covariance structure remaining in $\Sigma_{T|S}$ should be minimized. In the meantime, we would like to get higher data reduction rate. Since the column size of S is the number of the selected variables, the column size of S should be small. Then, our objective is to find T and S to minimize the covariance structure remaining in $\Sigma_{T|S}$, while the column size of S is kept small. The covariance structure remaining in $\Sigma_{T|S}$ is measured by the total sum of its eigen values, so the problem is formulated as:

$$\begin{aligned} (T^*, S^*) &= \arg \min_{\forall (T,S)} tr(\Sigma_{T|S}^2) + \alpha(q - t) \\ &= \sum_{i=1}^{t} \lambda_i^2 + \alpha(q - t) \end{aligned} \tag{9}$$

where $\alpha \geq 0$, t is the column size of $\Sigma_{T|S}$, $q - t$ is the column size of S and λ_i is the i^{th} eigen value of $\Sigma_{T|S}$. The formulation is constructed by adding a penalty, $\alpha(q - t)$, to McCabe's criteria [13] for attaining more aggressive reduction.

A simple way to find (T^*, S^*) is an exhaustive search to evaluate equation (9) for every possible partition. It is very computationally expensive. Considering the limitation of power resource in sensors, we use a computationally efficient greedy search (forward selection) [14] to find (T^*, S^*) in equation (9).

Start with $T = \mathcal{Y}$ and $S = \emptyset$ an empty matrix. The details are as follows:

1. (Initialization) Calculate the initial objective function. First, perform a singular value decomposition on $\Sigma_{T|S}$. The result is expressed as $\Sigma_{T|S} = ADA'$ since $\Sigma_{T|S}$ is symmetric, where D is a diagonal matrix with the eigen values λ_i's of $\Sigma_{T|S}$ as its diagonal elements. Then,

$$tr(\Sigma_{T|S}^2) = \sum_{i=1}^{t} \lambda_i^2 = \sum_{i=1}^{t} \lambda_i^2 \sum_{j=1}^{t} a_{ji}^2 = \sum_{j=1}^{t} \sum_{i=1}^{t} t_{ij}^2 \tag{10}$$

where a_{ij} is the (i, j)th element of A and t_{ij} is the (i, j)th element of $\Sigma_{T|S}$.

2. (Iteration) Evaluate the possible changes of $[T|S]$. For all $j = 1, ..., t$, we move the jth column, namely t_j, from T to S. Let the new partition be $[T'|S']$. Then, the decrease of $tr(\Sigma_{T|S}^2)$ due to this change, h_j is as follows [13] :

$$h_j = tr(\Sigma_{T|S}^2) - tr(\Sigma_{T'|S'}^2) = \sum_{i=1}^{t} t_{ij}^2 \tag{11}$$

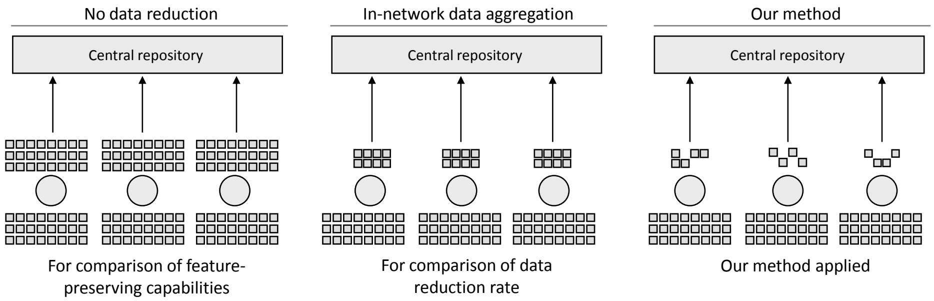
No data reduction
Central repository
For comparison of feature-preserving capabilities
In-network data aggregation
Central repository
For comparison of data reduction rate
Our method
Central repository
Our method applied

Fig. 3. Experimental configurations

Then, the improvement of the objective function due to this change is

$$\Delta_j = h_j - \alpha \tag{12}$$

3. (Check optimality) Update $[T|S]$ or stop the process. Let $k := \arg\max_{j=1,..,t} \Delta_j$. If $\Delta_k > 0$, we update $T = T \backslash t_k$, $S = S \vee t_k$, $t = t - 1$. Then, go to Step 2. Otherwise, we stop at $(T^*, S^*) = (T, S)$. Note that $T \backslash t_k$ removes the kth column from T and $S \vee t_k$ appends the k column of T to S.

After determining (T^*, S^*), we discard T^* from $\mathcal{Y}$ and use only variables corresponding to S^* for further investigation or decision making processes. For that, a central repository notifies the indices of data belonging to S^* to each corresponding sensor. For the example in Fig. 1, $\{1, 4, 6\}$ is the index set sent back to leaf node 1 after the global data reduction.

B. Reduction phase

After the collaboration phase, each sensor has the updated indices from the central repository. In this phase, when new data are collected, each sensor first reduces the new data by its local reduction scheme as in equation (5). Next, the sensor extracts the parts corresponding to the indices received from the central repository and sends them over.

III. Evaluation

An experiment is performed to evaluate the performance of our method in comparison with the in-network data aggregation method as well as with the case without in-network data reduction on sensors.

A. Experimental setup

There are a number of sensors attached to a forging machine. The sensors read vibration signals on the machine during a fixed period. The vibration signals are used to identify whether the forging process is normal or out of control. In the experiment, we used the vibration signals from a sensor and simulated the multi-sensor environment by segmenting the signals. An original signal is a 224 dimensional vector. It is randomly divided into 14 segments with the same size. Each segment is a 16-dimensional data vector, which represents data from a single sensor. Using the segments, we tested the following approaches. For better understanding, some illustrations are presented at Fig. 3.

1) No in-network data reduction: There are 14 sensors. Each sensor collects signals and sends it to a central repository without any in-network data reduction. In the central repository, we applied PCA to all the data from the 14 sensors. In this case, the central repository has all original data, so it can easily detect redundancy among the data. Thus, it is expected to show the highest data reduction than the other approaches.
2) In-network aggregation: There are 14 sensors. The jth sensor applies the local data reduction (PCA) to its data and sends the reduced data to the central repository. Then, we applied again PCA to the reduced data from the 14 sensors.
3) Our method: We use the proposed method with $\alpha = 1$ in equation (9).

There are 528 signals collected under the normal process and 378 signals collected under some process faults. We randomly selected 350 signals out of the 528 ones from the normal condition and 248 signals out of the 378 ones from the abnormal condition. The selected ones were used as a training dataset to build data reduction schemes for each of the three approaches. The remaining signals were used as the test dataset to estimate the fault detection errors.

B. Performance criteria

For all the three approaches, we would like to test the following: the data reduction capability, the amount of data transmission, and the information that the reduced data preserves.

1) Data reduction (DR): This measure is defined as the dimension of the final reduced data. It shows how much redundancy can be removed by each data reduction approach.
2) Amount of data transmission (DT): This measure indicates how many data should be transmitted from a leaf node to the central repository. It reflects the degree of communication intensity.
3) Fault detection error (FA,MD): Fault detection error is used as an indirect measure for how much the reduced dataset captures the information contained in the original dataset. If the fault detection error is low, it

means the reduced dataset represents the whole dataset very well. Otherwise, it means the reduced dataset loses some valuable information. This measure consists of two sub-measures: the miss-detection (MD) and the false-alarm (FA). MD is defined as the ratio of the number of miss detections and the total number of signals representing process faults. FA is defined as the ratio of the number of detections among the signals under the normal process.

C. Results

For each approach, the following procedure is repeated 50 times to obtain the performance measures:

1) Randomly select the training dataset X_t and the test dataset X_v as described in Section III.A.
2) Find the optimal data reduction procedure, $Rd(\cdot)$, given the training dataset randomly selected. For the first approach, the optimal data reduction is a projection matrix generated by PCA on the whole training dataset. For the second one, it consists 14 projection matrices generated by PCA on each of the 14 segments. For the last one, it consists of the same 14 projection matrices as in the second approach, plus the 14 index sets generated by the global data reduction step.
3) Apply $Rd(\cdot)$ to the training dataset and get the reduced training dataset, $Rd(X_t)$. Then, build the linear discriminant analysis (LDA) boundaries based on $Rd(X_t)$.
4) Apply $Rd(\cdot)$ to the test dataset and get the reduced test dataset, $Rd(X_v)$. Then, classify $Rd(X_v)$ by the linear boundaries established in 3).
5) Evaluate DR, DT, FA and MD.

Table I shows the average of DR, DT, FA and MD over the 50 repetitions. For the first and second approaches, the dimension of the reduced data (DR) after PCA at the central repository became one. Nevertheless, the down side of the first approach is that it needs to transmit all the original data of 224 dimensions. That is a high communication burden. The second approach improves the situation and it only transmits, on average, 15.6 data. Our method achieves the best data reduction in terms of the data transmission measure (DT); it is only one-third of the DT of the second approach or 2.2% of the original data dimension. The final data reduction rate of our approach is 5.1 (on average), higher than the other approaches. However, this data dimension can be further reduced. When we think of the central repository in the simple network as an intermediate node in the entire network, redundancy among the intermediate nodes is removed at the higher level intermediate nodes. On the other hand, since it does not achieve the highest possible data reduction rate, there could be potentially more capable methods that may make furthur improvements.

In terms of fault detection errors, all the three approaches are similar. That means the data reduced by the three approaches kept similar important information from the original data.

TABLE I
OVERALL PERFORMANCE

Configuration	DR	DT	FA	MD
No in-network data reduction	1	224	0.0534	0.0417
In-network data aggregation	1	15.6	0.0522	0.0440
Our method(collaborative reduction)	5.1	5.1	0.0343	0.0588

IV. Conclusion

We introduced a novel method for reducing within-sensor and between-sensor data redundancy through a collaborative approach. We demonstrated through the experimental results that doing so could capture sufficient information for fault detection while substantially reducing the data dimension. There are a couple of extensions that may be worth consideration. Our proposed method uses PCA and partial correlation so that it can detect only linear data redundancy. To remove non-linear data redundancy, one may need to apply our method in a kernel space. Additionaly, in the experimental study, we fixed α to 1 in equation (9). However, further studies are much needed regarding how to decide the optimal α.

References

[1] J. Pottie, and W. J. Kaiser, 2000, “Wireless integrated network sensors,” Communications of the ACM, vol.43, pp.51-58.

[2] T. Arici, B. Gedik, Y. Altunbasak, and L. Liu, 2003, “PINCO: a pipelined in-network compression scheme for data collection in wireless sensor networks,” In Proceedings of the 12th International Conference on Computer Communications and Networks, pp.539-544.

[3] C. Intanagonwiwat, R. Govindan, D. Estrin, and J. Heidemann, 2003, “Directed Diffusion for Wireless Sensor Networking,” IEEE/ACM Transactions on Networking, vol.11, no.1, pp.2-16.

[4] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, 2000, “Energy-efficient communication protocol for wireless microsensor networks,” In Proceedings of the Hawaii Conference on System Science, pp.1-10.

[5] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, 2002, “TAG: a tiny aggregation service for ad-hoc sensor networks,” In Proceedings of the Symposium on Operating Systems Design and Implementation, OSDI, vol.36, pp.131-146.

[6] L. Schwiebert, V. Annamalai, and S.K.S Gupta, 2003, “On tree-based convergecasting in wireless sensor networks,” In Proceedings of the IEEE Wilress Communications and Networking Conference, vol.3, pp.1942-1947.

[7] A. T. Hoang, and M. Motani, 2007, “Collaborative Broadcasting and Compression in Cluster-Based Wireless Sensor Networks,” ACM Transactions on Sensor Networks, vol.3, no.3, article 17.

[8] S. S. Pradhan, 2003, “Distributed Source Coding Using Syndromes (DISCUS): Design and Construction,” IEEE Transactions on Information Theory, vol.49, no.3, pp.626-643.

[9] H. Zhao, P. C. Yuen, and J. T. Kwok, 2006, “A Novel Incremental Principal Component Analysis and Its Application for Face Recognition,” IEEE Transactions on Systems, Man and Cybernetics.PART B: CYBERNETICS, vol.36, no.4, pp.873-886.

[10] A. Sharma, and K. K. Paliwal, 2007, “Fast principal component analysis using fixed-point algorithm,” Pattern Recognition Letters, vol.28, no.10, pp.1151-1155.

[11] J.A. Cumming, and D.A. Wooff, 2007, “Dimension reduction via principle variables,” Computational Statistics and Data Analysis, vol.52, pp.550-565.

[12] F. A. Graybill, 2000, “Theory and Application of the Linear Model,” Duxbury Press, pp.115-117.

[13] G. P. McCabet, 1984, “Principle variables,” Technometrics, vol.26, no.2, pp.137-144.

[14] T. Hastie, R. Tibshirani, and J. Friedman, 2001, The Elements of Statistical Learning, Springer, pp.59-75.