Energy Efficient Data Compression in Wireless Sensor Networks

Ranganathan Vidhyapriya[1] and Ponnusamy Vanathi[2]
[1]Department of Information Technology, PSG College of Technology, India
[2]Department of Electronics and Communication Engineering, PSG College of Technology, India

Abstract: *In order for wireless sensor networks to exploit signal, signal data must be collected at a multitude of sensors and must be shared among the sensors. The vast sharing of data among the sensors contradicts the requirements (energy efficiency, low latency and high accuracy) of wireless networked sensor. This paper describes our design and implementation of the two lossless data compression algorithms integrated with the shortest path routing technique to reduce the raw data size and to accomplish optimal trade-off between rate, energy, and accuracy in a sensor network. To validate and evaluate our work, we apply it to different types of datasets from different real-world deployments and show that our approaches can reduce energy consumption over other data compression schemes based on simulations.*

Keywords: *Wireless sensor networks; compression, routing, energy efficiency, lifetime, shortest path*

Received December 14, 2007; accepted March 12, 2008

1. Introduction

Advances in sensor and communication technology have focused interest on using wireless sensor networks, which are formed by a set of small untethered sensor devices that are deployed in an ad hoc fashion to cooperate on sensing a physical phenomenon, making the inferences, and transmitting the data [12]. Energy is a primary constraint in the design of sensor networks. This fundamental energy constraint further limits everything from data sensing rates and link bandwidth, to node size and weight. Large volumes of sensor data generated will make the data transmission within the network to a single information sink with minimal latency and energy is a very challenging task. Aggregation techniques such as TinyDB [5] and TAG [6] process and consume the collected data within the sensor network, forwarding only a small subset of the data to the sink. Query-based techniques such as directed diffusion aim to filter the data within the network to only what the application requires. Low-level networking techniques have been proposed to help route data within the sensor network with the hope of minimizing duplicated packets and minimizing the number of hops needed to deliver the data. Finally, data compression techniques are emerging for such sensor networks [1, 9] by compressing, the data size is reduced and less bandwidth is required for transmitting data. In the following section we survey other work relevant to our scheme. In section 3 we describe our algorithm. Section 4 presents an analysis of our experimental results. We conclude in section 5.

2. Related Work

Pradhan *et al.* [7] proposed a framework for distributed compression using joint source and channel coding. This approach minimizes the amount of inter-node communication for compression using both a quantized source and correlated side information within each individual node. Rabat *et al.* [10] propose a distributed matched source-channel communication architecture and reconstruction method from noisy random projections. A similar approach can be found in [13] which use a gossip communication scheme. Although it is claimed to be universal, there is a trade-off between power-distortion-latency. In addition, they do not consider the correlation of the data itself. Wagner [14, 11] proposed architecture for distributed wavelet analysis that removes the assumption about the regularity of the grid. Furthermore, it is not clear how to choose an optimal path for compression and the spatial correlation is not fully explored. Few other works in distributed audio and video compression in wireless sensor networks can be found at [2, 3]. Other approaches [13] try to solve multiple goals such as routing, aggregation, indexing and storage, and energy balancing with compression. However, previous work does not consider the data compression problem with any routing protocol in wireless sensor networks. The key difference between our work and these prior studies is that we focus data compression and on the deployment, e.g., network lifetime of sensor networks that identifies a shortest path in the network to transmit compressed data from the sensor nodes to the sink.

3. Proposed Method

The sensor nodes are distributed randomly in the sensing field. All nodes in the network are assigned with a unique ID. We assume each sensor node is stationary after deployment and is capable of getting its location information by the use of Global Positioning System (GPS). Each node has limited battery energy, whereas the available energy at the sink may be relatively unlimited. Data compression techniques can be classified into two methods as lossless compression method which does not tolerate any loss in data while compressing mostly suitable for applications such as health monitoring, executable programs and source code *etc.,* and lossy compression methods which overcomes small amount of data loss during compression suitable for non critical applications.

In camera sensor networks, some image file formats, notably PNG, use only lossless compression, while others like TIFF and MNG may use either lossless or lossy methods. GIF uses a lossless compression method, but most GIF implementations are incapable of representing full color, so they quantize the image (often with dithering) to 256 or fewer colors before encoding as GIF. Color quantization is a lossy process, but reconstructing the color image and then re-quantizing it produces no additional loss. In the case of camera sensor networks, images compressed losslessly occupy less space than the originals, but space-saving gains are modest, with compression ratios in the 2.5:1 range. Decompression restores the original image without loss of fidelity. Images stored in the GIF and PNG formats are compressed automatically, whereas for TIFF and BMP files, the user decides whether or not to compress the file. Lossy compression achieves higher compression ratios than lossless, but at the expense of image quality, with the degree of lossiness under user control. Lossy compression depends on the compression scheme and how it is implemented. The compression techniques which are used in this paper are based on lossless compression methods thus facilitating complete retrieval of data. We have proposed two compression techniques namely Entropy Encoded Codebook Compression (EECC) and Pipelined Codebook Compression (PCC) which are basically built over the codebook compression techniques.

3.1. Shortest Path Routing

Before sending the data to the sink, a node must start the neighbor discovery process, which is the address of all nodes that are able to transmit data to from the source. A special flooding mechanism is adopted in the neighbor discovery. The solution is to combine the broadcasting speed with the available energy on intermediate nodes. When an intermediate node receives broadcast message, it first checks its available energy. If the available energy is less than operation energy (e.g., twice the packet transmission energy), that indicates that the node has no more energy to take more transmission jobs. The node simply discards the broadcast message, if the node has sufficient energy; the node broadcasts the message to its neighboring nodes.

3.1.1. Shortest Path Algorithm

The dijkstra shortest path algorithm is used to calculate the path between the sink and the source. Given a network G = (N,E), with a positive cost D_{ij} for all edges (i, j∈N), start node *S* and a set *P* of permanently labeled nodes, the shortest path from start node *S* to every other node *j* is shown in Figure 1:

Initially P = {S}, DS = 0, and Dj = dSj for j∈N

Find i∉P such that Di = min Dj j∉P

Set P = PU{i}.

If P contains all nodes then

stop the algorithm is complete.

For all j ∉P

Set Dj = min [Dj,Di + dij]

Go to Step 1.

Figure 1. Shortest path algorithm.

3.2. Pipelined Codebook Compression

Collected sensor data packets are aggregated, combined into single packet, and redundancies in the data packets are removed to minimize data transmission. The collected data is in the form of a tuple of three values as shown in Figure 2.

Node Id	n-bit sensor measurement	Timestamp

Figure 2. Data Item produced by a sensor node.

The compression is done by checking all the most significant bits of the packets and combining the packets which have the same most significant data bits. Any two data packets can be merged only if they have the same shared most significant data bits. After the data packets are merged the resultant data is again compressed by using codebook method. Codebooks for the data packet are formed leaving out the most significant bits.

3.2.1. Dictionary Size

Codebook method of compression replaces strings of characters with single codes. At the start, the codebook is initialized to 256 entries. Our experiments focus on codebooks of 256, 512 and for comparison, an unlimited number of entries. Strings are encoded in 9 bits while the codebook has less than 512 enties, 10 bits while it has less than 1024 entries, *etc*. With small data blocks, there is almost no difference between the dictionary sizes.

To adapt codebook technique to a sensor node, three major inter-related points are balanced: the dictionary size, the size of the data to be compressed, and the protocol to follow when the dictionary fills. First, memory constraints require that the dictionary size be kept as small as possible. Additionally, as mentioned, it is required to compress and decompress relatively small, independent blocks of data, so that if a packet is lost it only affects the data that follows it in its own block. The codebook compression pseudo-code in its simplest form is as shown in Figure 3. This algorithm has no transmission overhead and is computationally simple. Since both the sender and the receiver have the initial dictionary and all new dictionary entries are created based on existing dictionary entries, the recipient can recreate the dictionary on the fly as data is received.

```
send start code
for each character {
   if new-entry appended with   character is not in
   dictionary
   {
      send code for new-entry
      add new-entry appended with  character as
      newdictionary entry
      set new-entry blank
   }
   append character to new-entry
}
send code for new-entry
send stop code
```

Figure 3. Pipelined codebook compression algorithm.

3.3. Entropy Encoded Codebook Compression

Initially the compression is done by checking the Most Significant Bits (MSB) of the packets, the packets are merged if they have the same MSB bits, to the remaining data entropy encoding method of compression is followed. Entropy encoding is a data compression scheme that assigns codes to symbols so as to match code lengths with the probabilities of the symbols. Entropy method compresses the data by replacing data's with symbols represented by equal-length codes where the length of each codeword is proportional to the negative logarithm of the probability. The technique works by creating a binary tree of nodes. These can be stored in a regular array, the size of which depends on the number of symbols (n).The algorithm as shown in Figure 4 works as follows:

- *Input*

$A = \{a_1, a_2, \ldots, an\}$ - symbol of alphabet size n
$W = \{w_1, w_2, \ldots wn\}$ - set of symbol weights.
i.e w_i = weight (a_i), $1 < i < n$

- *Output*

$C(A,W) = \{c_1, c_2, \ldots, cn\}$ - set of binary codewords where c_i is the codeword
for a_i, $1 < i < n$
Let $L(C) = {}_{i=1}\Sigma^n\, w_i \times \text{length}(c_i)$ be the weighted path length of code C.
The condition is $L(C) < L(T)$ for any code $T(A,W)$

Figure 4. Entropy encoded compression algorithm.

A node can be either a leaf node or an internal node. Initially, all nodes are leaf nodes; internal nodes contain symbol weight, links to two child nodes and the optional link to a parent node. As a common convention, bit '0' represents following the left child and bit '1' represents following the right child. A finished tree has N leaf nodes and N−1 internal nodes. The resulted compressed data is again compressed using codebook compression technique which is explained in the previous section and then the data are transmitted to the sink node wherein the data are decompressed.

4. Performance Analysis

In this section, we present our simulation results that show that the new techniques are beneficial in reducing the total transmission energy. The simulation results are generated using simulator developed using C. Network topologies are generated randomly. In case of high performance sensor networks such as structural health monitoring, disaster management, emergency response, *etc*. There exist a correlation between the energy consumption and the loss of data. As the data load increases, the key performance parameters of wireless sensor networks degrade. The data glut may result in unacceptable data loss, time delay and overall energy consumption.

4.1. Simulation Model

Sensor nodes around 100 are uniformly distributed over a 1000m×1000m area. Initially, 10 Joules of energy is assigned to every node and then we inject the network with 1000 randomly generated packets. The values of parameters used for simulations are as shown in Table 1. The source and destination of each packet are randomly chosen and the sizes of packets are drawn from a uniform distribution between 1 and 100 units. The effective radio range is 250 meters. The log-distance path loss model is used and the path loss exponent is set to 4.0. Data packets are generated at intervals of 1 second. The simulation is run for 750 seconds therefore each protocol has enough time to discover the route from the sink to the source and produce substantial amount of data traffic. The power consumption for transmitting one unit of data is 660mW, the power consumption for data reception is 35mW and the power consumption in the idle mode *i.e.,* when the sensor node is not in the sending or in the receiving and data is 35mW.

Table 1. Assumed parameters.

Parameters	Value	Parameters	Value
Bandwidth	2Mb/s	Transmission range	250 m
Transmit power	660mW	Topology Size	1000m x 1000m
Receive power	395mW	Number of sensors	100
Idle power	35mW	Packet rate	5 packets/s
Initial energy in batteries	10 Joules	Packet size	512 bytes

4.2. Energy Estimation

The power consumption model of the radio in embedded devices must take both transceiver and start-up power consumption into account along with an accurate model of the amplifier. The total energy consumed for transmitting and receiving (E_{TC}) a packet of *b* bits over a single hop wireless link of distance *d*, can be expressed as:

$$E_{TC}(b,d)=(E_T(b,d)+P_T T_{st}+E_{enc})+(E_R(b)+P_R T_{st}+E_{des}) \quad (1)$$

where E_T is the energy used by the transmitter circuitry and power amplifier, E_R is the energy used by the receiver circuitry, P_T is the power consumption of the transmitter circuitry, P_R is the power consumption of the receiver circuitry, T_{st} is the startup time of the transceiver, E_{enc} is the energy used to encode and *Edec* is the energy used to decode. Since the effect of startup component is not taken into account for multihop scenario and the encoding/decoding energies are also assumed to be negligible equation 1 can be simplified as:

$$E_{linear} = [\ n(e_{TE} + e_{RE}) - e_{RE} + (e_{TA}D^{\beta})/n^{\beta-1}] \quad (2)$$

where *n* is the number of hops between the source to the sink, e_{TE} is the energy used by transmitter, e_{RE} is the energy used by a receiver, e_{TA} is the energy used by an amplifier, *D* is the total distance between a source and a sink, and β is the path loss exponent of the channel. In this equation, the distance between any two neighboring nodes of a data path is assumed to be arbitrary. The energy used by transmitter can be further is given by equation 3:

$$e_{TA} = \frac{(S/N)_r (NF_{RE})(N_o)(BW)(4\pi/\lambda)^{\beta}}{(G_{ant})(\eta_{amp})(m_{bit})} \quad (3)$$

where $(S/N)_r$ is the signal to noise ratio at the receiver, NF_{RE} is the receiver noise figure, N_o is the thermal noise floor in a 1 Hz bandwidth , *BW* is the channel noise bandwidth, λ is the wavelength, G_{ant} is the antenna gain, η_{amp} is the transmitter amplifier efficiency, and m_{bit} is the raw bit rate. The following values are used for calculating the energy consumption in a multi-hop data transmission as shown in Table 2.

Table 2. Energy Estimation Parameters.

Parameter	Value	Parameter	Value
β	2	$(S/N)_r$	11dB
G_{ant}	-20dB	NF_{RE}	10dB
η_{amp}	0.2	N_O	4.17e-21J
m_{bit}	2.50e+05 bps	BW	2.50e+05 HZ
e_{TE}	1.45e-08 J	λ	0.125 m

4.3. Results and Analysis

In this section we compare the compression techniques EECC and PCC with the existing LZW technique. Lempel-Ziv-Welch (LZW) is a universal lossless data compression algorithm. The types of data used in the network for transmission are text, image and audio. The image data is converted into a stream of ones and zeroes by converting their pixel values into binary form. For the audio data the bit stream is obtained by sampling the signal at a required data rate. The metrics chosen to analyze the performance of our compression techniques are:

- Node Energy consumption is defined as the communication (transmitting and receiving) energy the network consumes; the idle energy is not counted.
- Packet delivery fraction is the ratio of the data packets delivered to the sinks to those generated by the sources.
- Average end-to-end delay of data packet includes all possible delays caused by queuing at the interface queue, retransmission delays at the MAC, and propagation and transfer times.
- Compression Ratio is the ratio of the number of bits saved by compression to the uncompressed file size

4.3.1. Node Energy Consumption

The node energy consumption is shown in Figure 5. The node energy consumption is an important metric to show when the first dead node appears. This graph illustrates the behavior of the network with the change in the node density. Thus with the change in the node density the positions of the node relative to each other vary resulting in the change of the shortest path, number of hops and the distance of transmission. From the graph it is inferred that out of the three techniques the EECC technique shows least energy consumption irrespective of the number of nodes followed by PCC technique.

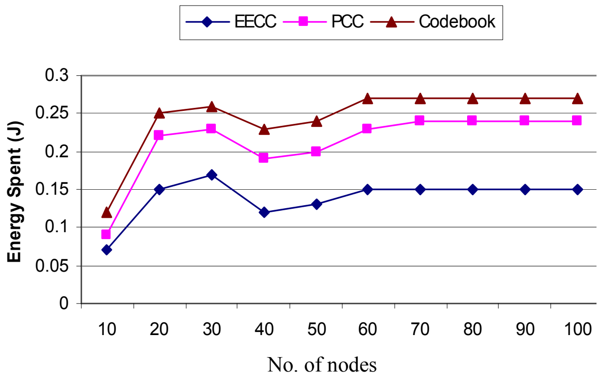

Figure 5. Node energy consumption.

The graph has been plotted keeping the source and sink node as constants. The variations in the graph over different values of the number of nodes are due to the random assignment of node positions. The hop count in the shortest path varies with each simulation due to the above factors.

4.3.2. Compression Ratio

From Figure 6, it is seen that EECC compression gives better compression ratio in comparison to others because of employing the codebook technique to entropy encoding. The compression ratio increases with the increase in packet size for all the techniques. For smaller packet sizes PCC compression technique performs better than the rest while with increase in the size of the packets EECC performs better

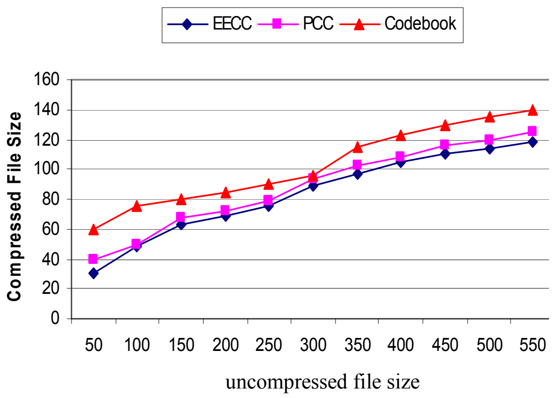

Figure 6. Compression ratio.

4.3.3. Packet Delivery Fraction

Figure 7 depicts the packet delivery fraction. It is found that for small packet sizes the compression ratio is dependent on the sequence of input stream and with the increase in the packet size EECC offers better performance and thus facilitates the transmission of more number of packets in the respective cases

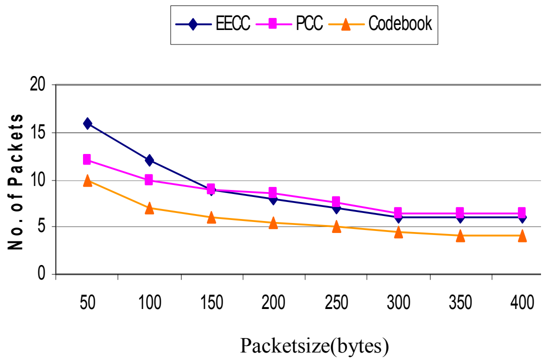

Figure 7. Delivery ratio.

4.3.4. Compression Ratio for CBR & VBR data

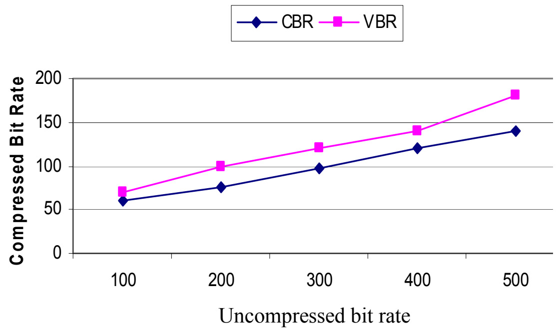

Figure 8. Compression ratio for CBR and VBR.

The compression techniques were tested for both Constant Bit Rate (CBR) and Variable Bit Rate (VBR) data types. The technique taken into consideration for the above plot is EECC since it is found to consume less energy when compared to PCC. Figure 8 shows that the compression ratio is better as the size of the packet increases which is why in the case of variable bit rate transmission, the efficiency is lesser due to transmission of a given amount of data using packets of smaller size.

4.3.5. Performance Comparison

Table 3 shows the performance of the lossless data compression algorithms. From the results it is inferred that entropy encoded compression codebook compression EECC provides a very good compression ratio of 0.18. As the compression ratio decreases, the amount of energy saved increases almost linearly with the same number of compressed packets per an original packet. This result could be inferred from Figure 4. Also, as the number of compressed data packets per original data increases, the amount of energy saved increases. The amount of energy saved can be expressed by the equation 4:

$$E_{save} = E_{linear}(X)(B_o - B) \qquad (4)$$

where *Esave* is the total amount of energy saved, *x* is the total number of compressed data packets, *B0* is the size of the original data packet in bit, and *B* is the size of the compressed data packet in bits. Since EECC is simple, there is a minimal computational overhead also this does not require any information exchange among sensor nodes, so there is no transmission overhead. By applying EECC, the amount of energy consumed for transmission is reduced largely.

Table 3. Comparison of compression techniques.

Parameter/ Technique	LZW	PCC	EECC
Compression Ratio	0.7259	0.3281	0.1868
Energy Consumption (J)	2.181835e-04	1.97893e-04	1.43785e-04
Simulation Time (ms)	7	8	10
Memory Usage (KB)	1.581	1.46	1.14

The drawbacks of entropy encoded compression technique are the processing time required for compression and the memory used for storing the compression algorithm. In the next case the pipelined codebook compression provides a higher compression ratio than EECC so the energy consumed in for data transmission increases and the network life time falls below the lifetime of ECC technique. In the case of LZW the compression ratio is high because the compressed file size is greater than PCC and EECC and so more amount of energy is required in sensor nodes for compressed data transmission and the network lifetime decreases. The advantage of this technique is that the amount of memory occupied by the compression is less and the processing time required is also less. As a result the overall delay of the data packet transmitted is reduced.

5. Conclusion

The performance of the Wireless Sensor Network (WSN) for various compression schemes has been analyzed in this paper. A compression technique which works well for a given wireless sensor network might not suit another WSN with different requirement. The further enhancements that can be imbibed in the WSNs is to embed numerous distributed devices to monitor and interact with physical world phenomena, and to exploit spatially and temporally dense sensing and actuation capabilities of the sensing devices.

References

[1] Bajwa W., Haupt J., Sayeed A., and Nowak R., "Compressive Wireless Sensing," *in Proceedings of Information Processing in Sensor Networks*, USA, pp. 134-142, 2006.

[2] Gehrig N. and Dragotti L., "Distributed Compression in Camera Sensor Networks," *in Proceedings of International Conference on Image Processing*, UK, pp. 82-85, 2001.

[3] Gehrig N. and Dragotti L., "Distributed Sampling and Compression of Scenes with Finite Rate of Innovation in Camera Sensor Networks," *in Proceedings of Data Compression Conference, pp*. 83-92, 2006.

[4] Karvonen H., Shelby Z., and Pomalaza-Raez C., "Coding for Energy Efficient Wireless Embedded Networks," *in Proceedings of International Workshop on Wireless Ad Hoc Networks*, Finland, pp. 300-304, 2004.

[5] Madden S., Franklin M., Hellerstein J., and Hong W., "TinyDB: An Acquisitional Query Processing System for Sensor Networks," *Computer Journal of ACM Transaction Database Systems,* vol. 30, no. 1, pp. 122–173, 2005.

[6] Madden S., Franklin M., and Hellerstein J., "TAG: A Tiny Aggregation Service for Ad-Hoc Sensor Networks," *in Proceedings of Annual Symposium on Operating System Design and Implementation (OSDI),* USA, pp. 131-1462, 2002.

[7] Magli E., Mancin M., and Merello L., "Low Complexity Video Compression for Wireless Sensor Networks," *in Proceedings of International Conference on Multimedia and Expo*, vol. 3, USA, pp. 585- 590, 2003.

[8] Petrovic D., Shah C., Ramchandran K., and Rabaey J., "Data Funneling: Routing with Aggregation and Compression for Wireless Sensor Networks," *in Proceedings of the IEEE International Workshop on Sensor Network Protocols and Applications*, USA, pp. 156-162, 2003.

[9] Pradhan S., Kusuma J., and Ramchandran K., "Distributed Compression in a Dense Micro-sensor Network," *IEEE Signal Processing Magazine,* vol. 19, no. 2, pp. 51-60, 2002.

[10] Rabbat M., Haupt J., Singh A., and Nowak D., "Decentralized Compression and Pre-distribution via Randomized Gossiping," *in Proceedings of International Conference on Information Processing in Sensor Networks,* USA, pp. 51-59, 2006.

[11] Roy O. and Vetterli M., "Distributed Compression in Acoustic Sensor Networks Using Oversampled A/D Conversion," *in Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing,* vol. 4, France, pp. 165-168, 2006.

[12] Tilak S., Abu-Ghazaleh N., and Heinzelman W., "A Taxonomy of Wireless Microsensor Network Models," *Computer Journal of ACM Mobile Computing and Communication Review* (MC2R), vol. 6, no. 2, pp. 1-8, 2002.

[13] Wagner R., Choi H., Baraniuk R., and Delouille V., "Distributed Wavelet Transform for Irregular Sensor Network Grids," *in Proceedings of IEEE Workshop on Statistical Signal Processing*, USA, pp. 1196-1201, 2005.

[14] Wagner S., Baraniuk G., Du S., Johnson B., and Cohen A, "An Architecture for Distributed Wavelet Analysis and Processing in Sensor Networks," *in Proceedings of International Conference on Information Processing in Sensor Networks*, USA, pp. 243-250, 2006.

Ranganathan Vidhyapriya has a master in applied electronics from the PSG College of Technology, India. Working as a senior lecturer in the Department of Information Technology, PSG, College of Technology. She is pursuing research in the field of Ad Hoc networks. Her research interests include energy optimization in heterogeneous wireless networks.

Ponnusamy Vanathi holds a doctorate in electronics and is with the Department of Electronics and Communication Engineering of the PSG College of Technology, India as an assistant professor. Her areas of research interests include speech signal processing, wireless networks, and information security.