A distributed and adaptive signal processing approach to reducing energy consumption in sensor networks

Jim Chou, Dragan Petrovic,
Kannan Ramchandran

Department of EECS
University of California at Berkeley

IEEE INFOCOM 2003

Outline

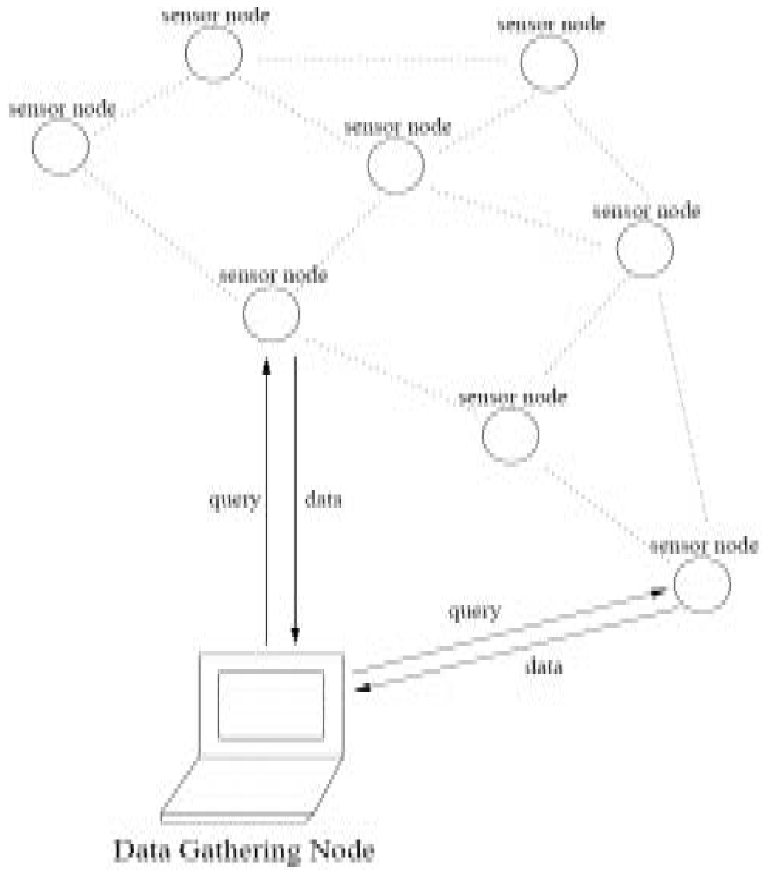
- Introduction
- Distributed Compression
- Correlation Tracking
- Querying and Reporting Algorithm
- Simulation
- Conclusion and Discussion
- Reference

Introduction

- Sensor nodes are typically of small physical dimensions and operated by battery power, making energy consumption a major concern.
- The topic of energy-aware routing to alleviate energy consumption in sensor networks has received attention recently.

Introduction

- Here we assume sensor network architectures having two types of node: sensor nodes and data-gathering nodes.



Introduction

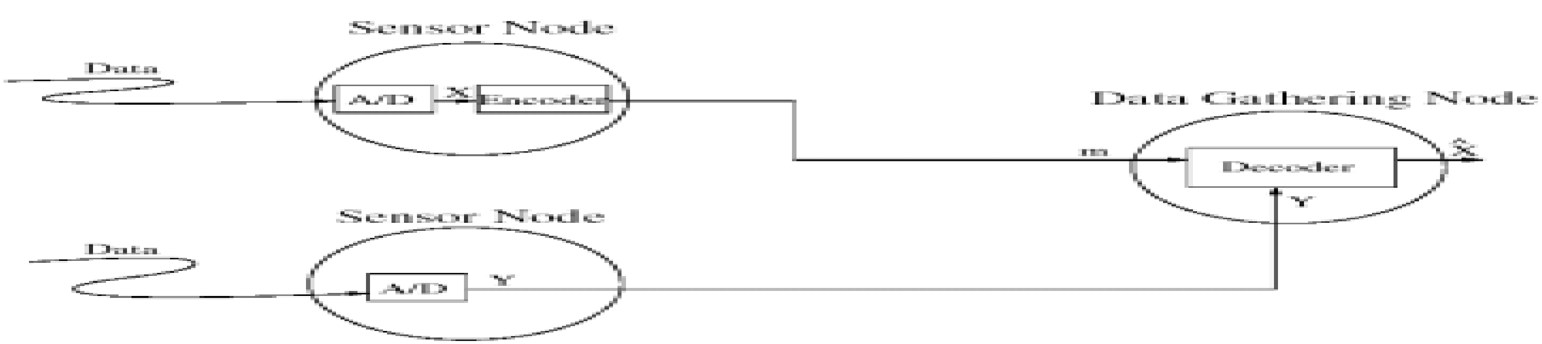
- Since dense sensor networks are particularly rich in correlations, the authors judiciously exploiting existing sensor data correlations in a distributed manner.
- Furthermore, compression can be effected in a fully blind manner without the sensor nodes ever knowing what the other correlated sensor nodes have measured.

Introduction

- Main challenges
  - Sensor nodes: devising a computationally inexpensive encoder that can support multiple compression rates.
  - data-gathering nodes: determining an adaptive correlation-tracking algorithm that can continuously track the amount of correlation that exists between the sensor nodes.

Distributed Compression

- One sensor send its data Y to the data gathering node.
- The other sensor compress it data X and then transmit compressed data m to the data gathering node.
- Data gathering node can use Y to decode m to X since Y is correlate to X.



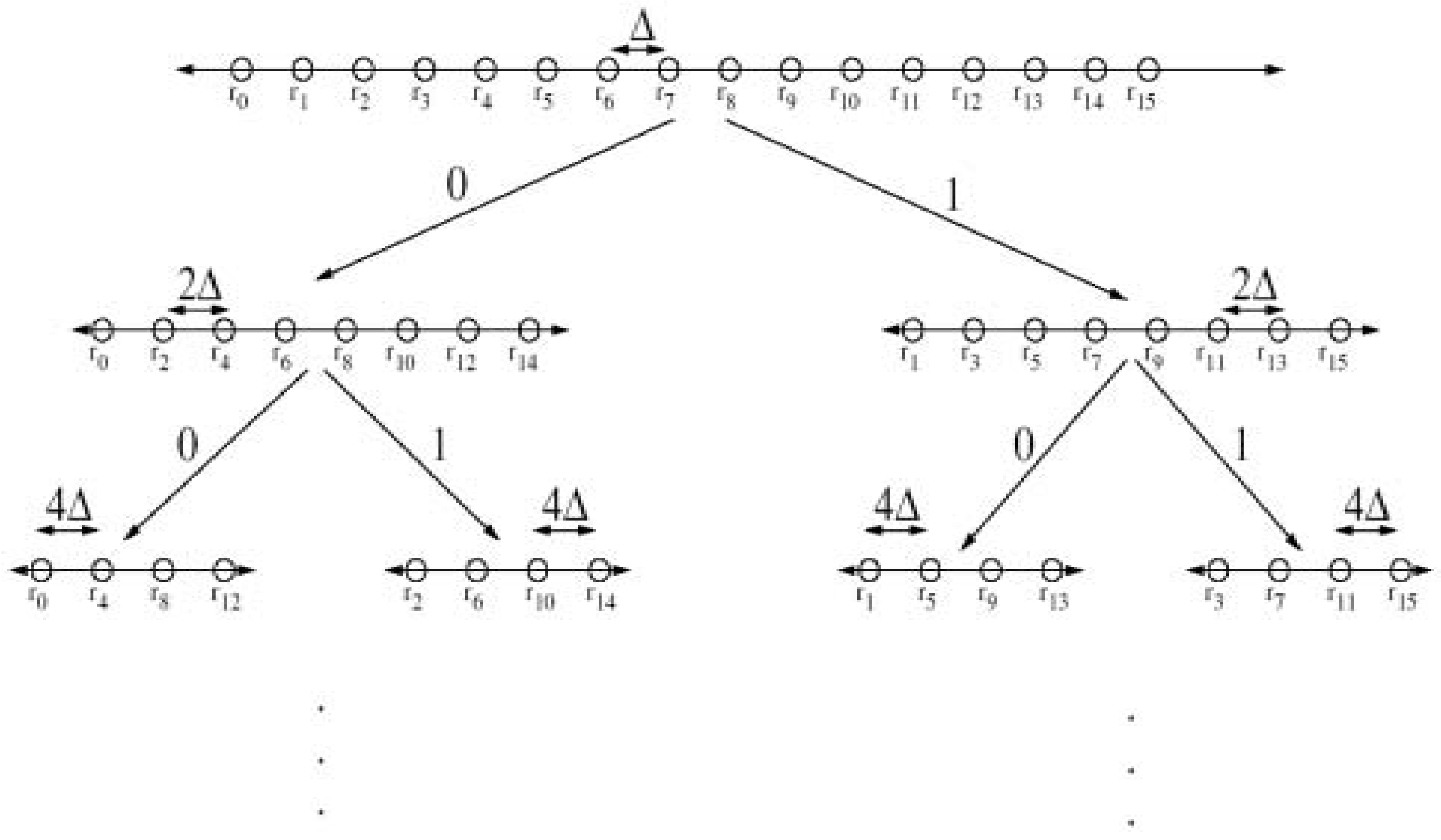
Distributed Compression

- The compression rate is directly dependent on the amount of correlation in the data, which might be time-varying.
- It is desirable to have one underlying codebook that is not changed among the sensors but can also support multiple compression rates.
- Tree-based distributed compression code was proposed.

Distributed Compression

- Assume the sensor uses an n-bit A/D converter, We start with a root codebook that contains 2^n representative values on the real axis.
- Partition the root codebook into two subsets.
- This process is repeated n times, resulting in an *n*-level tree structure that contains 2^n leaf nodes,

Distributed Compression

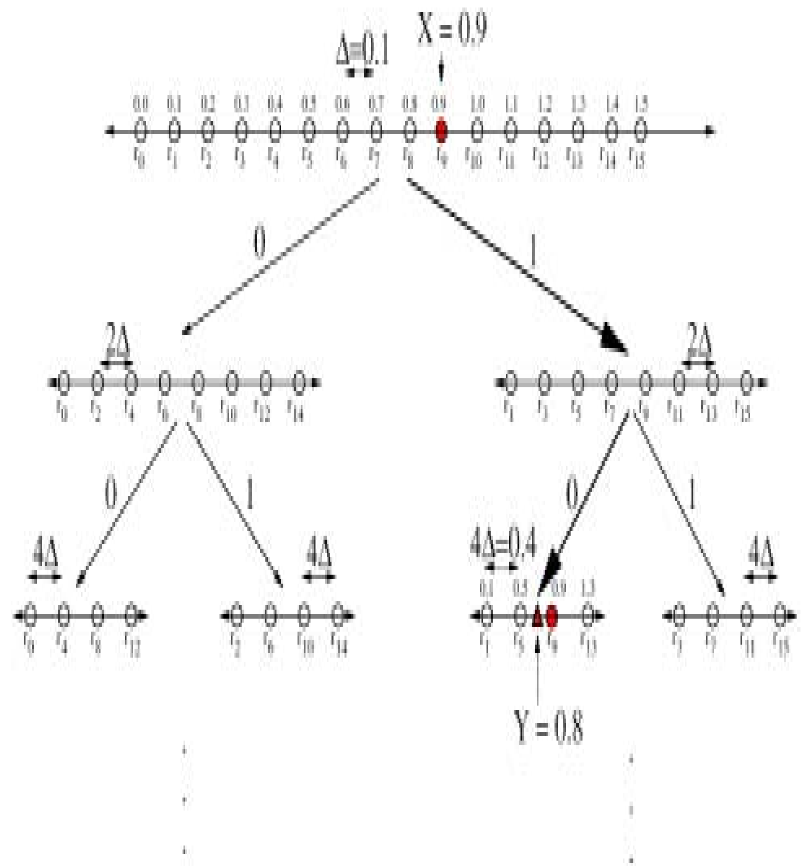


Distributed Compression

- The encoder will receive a request from the data gathering node requesting that it encode its readings using *i* bits.
- The path through the tree to level-i sub-codebook will specify the bits that are transferred to the data gathering node.
- The decoder will receive the *i*-bit value, then decode with side-information, Y.

Distributed Compression

- Since Y is less than $2^{i-1}\Delta$ away from X, where Δ is the spacing in the root codebook, decoder can recover X perfectly.



Correlation Tracking

- In practice, we choose to use a linear predictive model where $Y_k^{(j)}$ is a linear combination of values that are available at the decoder:

$$Y_k^{(j)} = \sum_{l=1}^{M} \alpha_l X_{k-l}^{(j)} + \sum_{i=1}^{j-1} \beta_i X_k^{(i)}$$

- The prediction, $Y_k^{(j)}$, determines the number of bits needed to represent $X_k^{(j)}$.
- Thus, the main objective of the decoder is to derive a good estimate of $X_k^{(j)}$.

Correlation Tracking

- we would like for the decoder to be able to find the prediction coefficient α_l ; l = 1, ...,M and β_i ; i = 1, ...,$j-1$ that minimize the mean squared error between $Y_k^{(j)}$ and $X_k^{(j)}$.
- Finally we can solve the optimal coefficient factor Γ_j as follow:

$$\vec{\Gamma_{j,opt}} = R_{zz}^{-1,j} \vec{P}_j$$

Correlation Tracking

- The coefficient factor Γ_j must be continuously adjusted to minimize the mean-squared error since the statistics of the data may be time varying.
- Thus, we use Least-Mean-Squares (LMS) algorithm and the steps in calculating the LMS solution is summarized below:

$$
\begin{aligned}
&1.\ Y_k^{(j)} = \vec{\Gamma}_j^{(k)T} \vec{Z}_{k,j} \\
&2.\ N_{k,j} = X_k^{(j)} - Y_k^{(j)} \\
&3.\ \vec{\Gamma}_j^{(k+1)} = \vec{\Gamma}_j^{(k)} + \mu \vec{Z}_{k,j} N_{k,j}
\end{aligned}
$$

Correlation Tracking

- Finally, since if $|N_{k,j}| > 2^{i-1}\Delta$, however, then a decoding error will occur, We can use Chebyshev's inequality to bound this probability of error:

$$P[|N_{k,j}| > 2^{i-1}\Delta] \leq \frac{\sigma^2_{N_j}}{(2^{i-1}\Delta)^2}$$

- Then we can caculate the value *i* by:

$$i = \frac{1}{2} log_2(\frac{\sigma^2_{N_j}}{\Delta^2 P_e}) + 1$$

Querying and Reporting Algorithm

Pseudocode for data gathering node:

Initialization:

```
for (i = 0; i < K; i++)
  for (j = 0; j < num_sensors; j++)
    Ask sensor j for its uncoded reading
  for each pair of sensors i,j
    update correlation parameters using Eqs. (16) and
    (10).
```

Main Loop:

```
for (k = K; k < N; k++)
  Request a sensor for uncoded reading
  for each remaining sensor
    determine number of bits, i, to request for using
    Eq.(14).
    Request for i bits
  Decode data for each sensor.
  Update correlation parameters for each sensor.
```

Pseudocode for sensor nodes:

```
For each request
  Extract i from the request
  Get X[n] from A/D converter
  Transmit n mod 2^i
```

Simulation

- Assumed sensor has a 12 bit A/D converter with a dynamic range of [−128, 128] in our simulations and further assumed a star topology where the data gathering node queried 5 sensor nodes directly.

Simulation

- Correlation tracking
  - top graph represents the tolerable noise
  - bottom graph represents the actual prediction noise.
  - For the simulations, zero decoding errors were made for 90,000 samples of humidity, temperature and light.

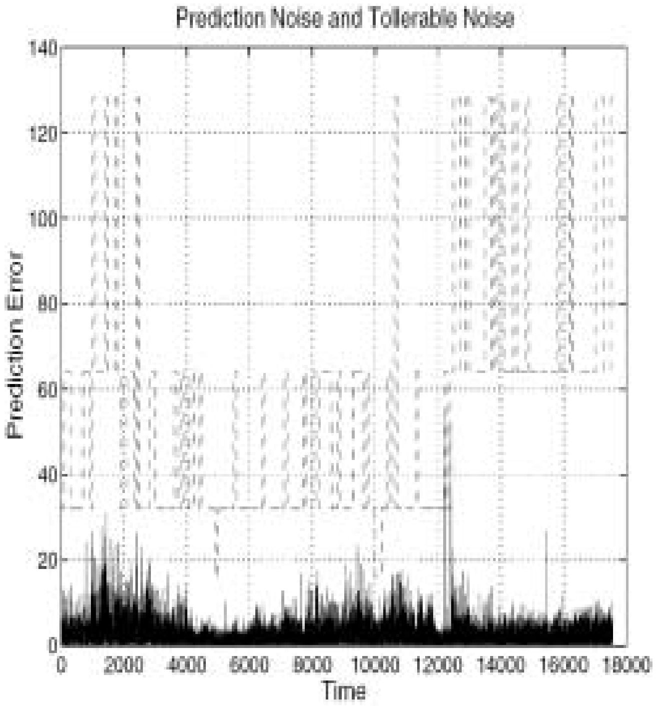


Fig. 5. Tolerable noise vs. prediction noise for 18,000 samples of humidity. The tolerable noise is the amount of noise that can exist between the prediction of a sensor reading and the actual sensor reading without inducing a decoding error.

Simulation

- Energy Saving
  - We can see that LMS algorithm is better suited for tracking correlations.

Data Set	Temperature	Humidity	Light
Ave Energy Savings	66.6%	44.9%	11.7%

TABLE I

AVERAGE ENERGY SAVINGS OVER AN UNCODED SYSTEM FOR SENSOR NODES MEASURING TEMPERATURE, HUMIDITY AND LIGHT

Conclusion and Discussion

- This paper have proposed a method of reducing energy consumption in sensor networks by using distributed compression and adaptive prediction.
- Allowing nodes to compress their readings to different levels without having the nodes know what the other nodes are measuring.
- Average energy savings per sensor node of 10–65% can be achieved using this algorithm

Reference

- R. Shah and J. Rabaey, "Energy aware routing for low energy ad hoc sensor networks," *Proc. of IEEE WCNC*, Mar 2002.
- D. E. C. Intanagonwiwat, R. Govindan, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," *Proc. of IEEE MobiCom*, Aug 2000.
- J. Chou, D. Petrovic, and K. Ramchandran, "Tracking and exploiting correlations in dense sensor networks," *Proceedings of the Asilomar Conference on Signals, Systems and Computers*, November 2002.