Compressed sensing in data collection

Yiying Zhao
5110309484
F1103015
June 20, 2014

Abstract

Energy efficiency of data collection is one of the dominating issues of wireless sensor networks(WSNs), especially when considering data collection. Data collection will cause serious traffic load at the sink node for all the data will converge to the sink node. The newly developed technique, compressed sensing(CS), provides another method that allowing network recover the signal with high probability from far fewer samples than original dimension. This report will introduce the basic knowledge of compressed sensing and describe the model of wireless sensor network which is implied with CS.

Introduction

Wireless sensor networks have been used in many situations such as forest data collection and fire detection. Data gathering is the main function of wireless sensor networks. However, there are still many problems to be solved in data gathering in wireless sensor network. Efficiency and adaptability are two very important issues in data gathering. With the traditional data gathering approach, the sink receives one data packet from each sensor node in the typical scenario, leading to a large amount of traffic. The intensity of data traffic has a serious impact on the lifespan of WSNs when considering the battery. If the amount of the resulting traffic can be reduced, the lifespan of the whole network will be significantly prolonged.

A newly developed technique, compressed sensing promised to recover signals with high probability from far fewer samples than their original signal, as long as the original signal is sparse in some dimension. In fact, every signal is sparse in some certain dimension. This property means CS can be implied in field.

This report will introduce the CS in the second part. In the third part I wil talk about the simple model of wireless networks with CS.

Compressed sensing

The Shannon/Nyquist sampling theorem [1] specifies that to avoid losing information when capturing a signal, one must sample at least two times faster than the signal bandwidth. A new method to capture and represent compressible signals at a rate significantly below the Nyquist rate has been proposed. This method, called compressive sensing, employs nonadaptive linear projections that preserve the structure of the signal; the signal is then reconstructed from these projections using an optimization process. [5]

The compressed sensing can be divided into three steps:

1. Sparse representation

Consider a real-valued, finite-length, one-dimensional, discrete-time signal x, which can be viewed as an N×1column vector inR^N. Any signal in RN can be represented in terms of a basis of N×1 vectors {ψi}. For simplicity, assume that the basis is orthonormal. In this case, signal x can be expressed:

$$X=\sum s_i \psi_i \qquad (1)$$

Where sis the N×1column vector of weighting coefficients. $s_i =< x, \psi_i >$. We use s=$< s_0, s_2, \dots, s_{N-1} >$. Clearly, x and s are equivalent representations of the signal, with x in the time or space domain and sin the ψ domain.

The signal x is K it is a linear combination of only K basis vectors; that is, only K of the s_i coefficients in (1) are nonzero and (N−K) are zero. The case of interest is when K<<N. The signal x is also compressible if the representation (1) has just a few large coefficients and many small coefficients.[2]

2. Coding

The fact that compressible signals are well approximated by K-sparse representations forms the foundation of transform coding. Tradition coding methods is to compute the vector s by the equation s=ψ^Tx, then discard the small coefficients. This method suffers from three inherent inefficiencies. First, the initial number of samples N may be large even if the desired K is small. Second, the set of all N transform coefficients $\{s_i\}$ must be computed even though all but K of them will be remained. Third, the locations of the large coefficients must be encoded, which add the data size.

Compressive sensing get these inefficiencies by directly acquiring a compressed signal representation without going through the intermediate stage of acquiring N samples. Consider a general linear measurement process that computes M < N inner products between x and a matrix ϕ whose dimension is M*N. Then, by substituting from (1), y can be written as

$$y=\phi x=\varphi\psi s=\theta s$$

Where $\theta=\varphi\psi$ is an M×N matrix. The measurement process is not adaptive, meaning that

is fixed and does not depend on the signal x. by doing this, data is compressed by sensing the matrix θ[2].

3. Recover algorithm

According to previous discussion, if we want to get the original signal x from y, we are going to solve the problem:

Min $\{||s||_{l0}\}$, subject to y=θs (2)

$||s||_{l0}$ means the number of nonzero elements in vector s.

Obviously, the dimension of y is far less than the dimension of s, from some paper we know it is a NP-hard problem.

To solve the problem, there are many methods. Donoha[3] comes up with a method that we can use

Min $\{||s||_{l1}\}$, subject to y=θs (3)

To replace the (2)

$||s||_{l1}$ means the sum of element square in vectors.

This is largely simplify the complex of the problem for this is a linear problem which has been studied by decades. Donoha has demonstrate that as long as the signal satisfied

M >= cKlog(N/K) (4)

And the sensing matrix is a Gaussian distribution matrix. To solve this linear problem we use BP algorithm.

Another way to solve the problem is MP algorithm, which is a greedy algorithm. Mallat and Zhang *** discussed it for approximate solve the problem. The details of MP algorithm are as followed:

Input: sample y, sensing matrix θ

Initial: approximate result s^0=null, residual r=y

Step 1: find the k which minimum$||\mathrm{r} - a_k\theta_k||$,

Step 2: r=r-$a_k\theta_k$, $s^n = s^{n-1} + a_k\theta_k$

Step 3: if r=0 or n>threshold end the loop, else go to the step 1

A feature of the algorithm is that when stopped after a few steps, it yields an approximation using only a few atoms. When the dictionary is orthogonal, the method works perfectly. When the dictionary is not orthogonal, the situation is less clear. For example, if the algorithm chooses a wrong element at the first step, it will spend more time to minimum this incorrect, which means more elements in the answer than the correct one.

Network model

In this report, we only build a simple model to show the wireless sensor network combined with compressed sensing.

Firstly, we assume all the nodes are deployed intentionally, which means we have already know the connection between those nodes, we also neglect the transmission loss. So it extremely simplifies the model.

The wireless sensor network contains one sink node, which place all the data will be send to, and many sensor nodes. Constructing this model needs three steps.

Step 1

Divide the nodes set (apart from the sink node) into two sets, A and B. the sum of those nodes are N.

For those nodes in set A, they receive data less than k. for those nodes in set B, they receive data more than k-1.

Step 2

For those nodes in set A, the nodes directly send what they received to the upper node.

For those nodes in set b, nodes receive data u_r more than k-1, assuming m. Then we use a sensing matrix θ_k whose dimension is k*m to convert received data into a k vectoru_t.

$$u_t = \theta_k * u_r \qquad (5)$$

Then send the u_t to the upper node.

So data is compressed and reduce the traffic node in the network.

Note that K is not constant in the network is not constant. Actually if k is small, for the node around the sink node they may discard some information because they need to compress huge data into a small k. On the other hand, if k is large, the CS will have no advantages over the tradition methods. So the k is vary. For those nodes far away the sink nodes, data they get is small, so the k is small. For nodes near the sink nodes, k is large.

Another question is that after compressed whether the u_t is still sparse so that we can do another compress? The answer is yes. Taking a simple example

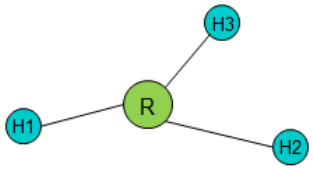

Figure 1 a simple network model

For node R, it receives data from node H1, H2 and H3. Assuming the data R get is $U_{r1}, U_{r2}, \text{and } U_{r3}$. For those vectors they are sparse in some certain dimension that $\theta_1, \theta_2 \ and \ \theta_3$, so we can express it:

$$U_r = \begin{bmatrix} U_{r1} \\ U_{r2} \\ U_{r3} \end{bmatrix} = \begin{bmatrix} \theta_1 & 0 & 0 \\ 0 & \theta_2 & 0 \\ 0 & 0 & \theta_3 \end{bmatrix} * \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} = \theta * S \qquad (6)$$

It is obvious that U is sparse. After we use the compressed sensing we can still get the U_r.

Step 3
In step 3, we are going to reconstruct the data so we will know the data from every sensor node. The reconstruction algorithm has been discussed in the previous section and we will not discuss it here

Conclusion

Data aggregation is one of the major research topics for wireless sensor networks due to its promising effect in reducing data traffic. in this report, we propose a method which combined with CS. In the second part introduce the CS and in the third part we discuss a simple network model. Although I do not do some simulation, it is obvious that CS can greatly reduce the traffic load in the network.

Reference

[1] Compressed sensing David L. Donoho

[2] a lecture on compressive sensing Richard G. Baraniuk

[3] Atomic decomposition by basis pursuit SCOTT SHAOBING CHEN, DAVID L. DONOHO, ANDMICHAEL A. SAUNDERS

[4] Compressed Data Aggregation: Energy-Efficient and High-Fidelity Data Collection Liu Xiang, Jun Luo, Catherine Rosenberg