BCS: Compressive Sensing for Binary Sparse Signals

Ukash Nakarmi and Nazanin Rahnavard
School of Electrical and Computer Engineering
Oklahoma State University
Stillwater, OK, 74078
Emails: {ukash.nakarmi, nazanin.rahnavard}@okstate.edu

***Abstract*—Model-based compressive sensing (CS) for signal-specific applications is of particular interest in the sparse signal approximation. In this paper, we deal with a special class of sparse signals with binary entries. Unlike conventional CS approaches based on l_1 minimization, we model the CS process with a bi-partite graph. We design a novel sampling matrix with *unique sum property*, which can be universally applied to any binary signal. Moreover, a novel binary CS decoding algorithm (BCS) based on graph and unique sum table, which does not need complex optimization process, is proposed. Proposed method is verified and compared with existing solutions through mathematical analysis and numerical simulations.**

I. Introduction

The paradigm of sparse sampling and reconstruction, called compressive sensing (CS), has been the state of art research in recent years. The gist of CS lies in determining the sparse solution of an under-determined system of linear equations. CS [1], [2] approaches have opened up many new research avenues in the field of under-determined systems and have found many practical applications in image processing, wireless communication, genetic and molecular analysis, data-streaming, medical resonance imaging, spectrum sensing etc.

For illustration, let us consider a K sparse, N length source $\underline{X} = \{x_1, x_2, \ldots, x_N\}$ be defined by $M << N$ linear equations given by (1)

$$\underline{Y} = \Phi \underline{X}. \tag{1}$$

In terms of compressive sensing, $\Phi \in \mathbb{R}^{M \times N}$ is called sampling matrix or measurement matrix and $\underline{Y} \in \mathbb{R}^M$ is linear functionals of sparse source $\underline{X}$ and is called compressed measurements. In general, the system in (1) is ill-posed, but CS theory asserts that under the conditions that the source X is sparse and sampling matrix Φ satisfies the Restricted Isometry Property (RIP) [3], [4], the approximate solution to (1) is obtained by that l_1 minimization given by

$$\widehat{\underline{X}} = argmin||\underline{X}||_{l_1} \quad s.t.\ \underline{Y} = \Phi \underline{X}. \tag{2}$$

In this paper, we introduce a novel compressive sensing approach for a *special class of signals with binary entries*. We first design a sampling matrix for binary signals. Next, we present a novel compressive decoding algorithm for binary sparse signals.

The rest of the paper is organized as follows: In Section II, we formally define the binary compressive sensing problem and provide a quick review on the existing studies on binary compressive sensing. In Section III, we introduce a novel sampling matrix and compressive decoding algorithm for binary signals. Section IV presents numerical analysis of the proposed scheme. Section V verifies our proposed scheme with numerical simulations and comparison with the existing methods, and finally the Section VI concludes the paper.

II. Related Work and Contribution

In binary CS, instead of considering real valued signals, we have the prior information that the source signal $\underline{X}$ is binary. The system is thus redefined as:

$$\underline{Y} = \Phi \underline{X}, \tag{3}$$

where, $\underline{X} \in \mathbb{B}^N$ and $\mathbb{B} = \{0, 1\}$. Binary sparse signals come into account in many practical applications such as event detection in wireless sensor networks, group testing, spectrum hole detection for cognitive radios, etc [5]. Linear programming based solution to (3) have already been proposed and discussed. In [6], the author modified the solution (2) for the binary system in (3) by limiting the reconstructed signal in range $0 \leq x_i \leq 1$. The author has also provided the recovery threshold for l_1 reconstruction of the binary signal. Though this method improves the performance for the binary sparse signal reconstruction, the reconstructed signal $\hat{x}_i \in [0, 1]$, whereas, the original $x_i \in \{0, 1\}$. The solution is not able to exactly reconstruct the original signal in $\underline{X} \in \mathbb{B}^N$. In a very recent work [7], the authors employ integer programming model to solve the under-determined binary systems of linear equations. The work is basically the solution for the general binary systems of equations. In compressive sensing, we have an added advantage of *having control over the sampling matrix* Φ, the elements of which are the coefficients of the linear equations. In [8], [9], the authors proposed the use of bipartite graphs to represent the CS and have explained the rate distortion performance of binary CS based upon the edge evolution in low-density parity-check codes [10]. The authors have provided an interesting closed form solution for edge evolution using large deviation probability theory and the martingales [11]. However, the edge evolution process halts after some iteration and thus the reconstruction solution is incomplete.

In this paper, we first design a universal sampling matrix Φ for binary signals, suitable for graph based recovery. Next, we propose a novel graph and sum-based CS decoding algorithm for binary sparse signals. Moreover, we provide analysis of our scheme and discuss the measurements required, error floor, and complexity and verify our scheme using both numerical analysis and simulations.

III. BCS: Compressive Sensing for Binary Sparse Signal

Let us represent the binary compressive sensing system in (3) by a bipartite graph as in Fig. 1. The coefficients of the

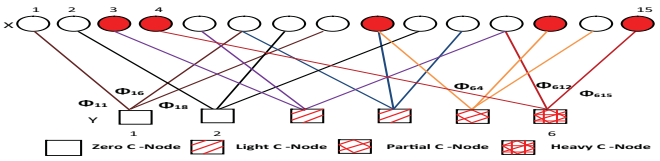

Fig. 1. Representation of Compressive Sensing with Bipartite Graph

binary sparse signal $\underline{X}$ is represented by the circular nodes and the compressive sensing measurements, i.e. elements of $\underline{Y}$ are represented by the square nodes. These nodes are termed as Variable nodes ($V-nodes$) and the Check nodes ($C-nodes$), respectively. The i^{th} $V-node$ and j^{th} $C-node$ are represented by v_i and c_j and their corresponding values are represented by x_i and y_j, respectively. The edge E_{ij} between v_i and c_j corresponds to the j^{th} row and i^{th} column element of sampling matrix, i.e., $(\Phi)_{j,i}$. A node v_i is said to be a neighbor of c_j if E_{ij} exists, i.e. $(\Phi)_{ji} \neq 0$.

In Fig. 1, the filled V-nodes represent $x_i = 1$ and the empty V-nodes represent $x_i = 0$. Similarly, the $C-nodes$ are divided into 4 different types as shown in the Fig.1 (they will be defined later). In previous binary compressive sensing studies using bipartite graph [8], [9], the sampling matrix Φ is a constant row weight binary matrix with row weight L. In [8], [9], the process of recovering $V-nodes$ associated with $C-nodes$ with values $y_i = 0$ and $y_i = L$ is termed as *First Phase Recovery (FPR)*. After the first phase recovery, the edge recovery is performed by corresponding edge removal process (for details on edge recovery please refer to [8], [9], [10]). However, this approach has two major setbacks. First, the edge recovery process does not recover all the $V-nodes$ and second, the work does not discuss the effect of row-weight L on the overall recovery of the $V-nodes$. In the following section, we first design the sampling matrix Φ and then discuss the consequence of row weight L on the FPR.

A. The Sampling Matrix Φ

Let us consider an $M \times N$ sampling matrix Φ of row weight L. Each row of Φ contains L non-zero elements placed randomly. Let Γ_i denote the set of indices of non-zero elements in the i^{th} row and Γ_{i1} and Γ_{i2} be any two arbitrary subsets: Γ_{i1} , $\Gamma_{i2} \subseteq \Gamma_i$ and $\Gamma_{i1} \neq \Gamma_{i2}$ for $i = 1, 2, \ldots M$. We say Φ has *Unique Sum Property* if the following condition is satisfied for $i = 1, 2, \ldots M$:

$$\sum_{j \in \Gamma_{i1}} \phi_{ij} \neq \sum_{j \in \Gamma_{i2}} \phi_{ij}. \tag{4}$$

In words, Φ has *Unique Sum Property* if all the possible sums of non-zero elements in each row of Φ are unique. For a finite constant row-weight L, when the elements of Φ are sampled from *continuous* random distributions such as Gaussian or Uniform, (4) is easily satisfied [12]. Hence, for $i = 1, 2. \ldots . M, the$, the i^{th} row of sampling matrix Φ is constructed by following steps:

First, Γ_i is formed by generating L random positions from $1\ to\ N$. Next, we set the value of $\Phi_{i,j}$ to be a random number generated from a continuous distribution such as Gaussian or Uniform if $j \in \Gamma_i$, else we set $\Phi_{i,j} = 0$.

Hence, the sampling matrix in our method is a sparse constant row weight matrix whose non-zero elements are drawn from the Gaussian or uniform distribution and each row of Φ satisfies *Unique Sum Property* in (4).

B. Compressed Measurements

The compressed measurements $(C-nodes)$ are weighted sums of the L random $V-nodes$ as represented by (3) and the Fig.1. We divide the $C-nodes$ into the four groups: Zero $C-node$, Light $C-node$, Partial $C-node$, and Heavy $C-node$, whose definitions follow.

Definition 1: *A $C-node$, c_j is said to be a Zero $C-node$, if $y_j = 0$. In Fig. 1, c_1, c_2 are examples of Zero $C-nodes$.*

Definition 2: *A $C-node$, c_j is said to be Light $C-node$, if, $y_j = \phi_{j,k}$, where $k \in \Gamma_j$. In other words, c_j s said to be a Light $C-node$ if y_j is equal to any of the non-zero element of the j^{th} row of Φ. This happens if and only if one of the neighboring V-nodes of c_j is one and all other neighbors are zeros (This is due to the Unique Sum Property of Φ). In Fig. 1, c_3, c_4 are examples of Light $C-nodes$.*

Definition 3: *c_5 is an example of Partial $C-node$. A $C-node$, c_j is said to be a Partial $C-node$ if $y_j \neq 0 \neq \phi_{j,k} \quad \forall k \in \Gamma_j$. However, during iterative edge recovery process, the node ultimately turns to be either Zero $C-node$ or Light $C-node$.*

Definition 4: *A Heavy $C-node$, c_j occurs when c_j has at least two neighboring $V-nodes$ having value of one and c_j as their sole neighbor (these nodes cannot be recovered by edge removal process). c_6 is an example of a Heavy $C-node$.*

C. Compressive Sensing Decoding for Binary Sparse Signals

Our proposed node recovery or decoding process is divided into three steps.

1) ***First Phase Recovery***: At the First Phase Recovery (FPR), the $V-nodes$ that are neighbors of Zero $C-nodes$ and Light $C-nodes$ are recovered as follows:

$$\begin{aligned} &if\ c_j\ is\ a\ Zero\ C-node,\ \ i.e.\ y_j = 0, \\ &\quad x_i = 0; \quad \forall\, i \in \Gamma_j; \end{aligned} \tag{5}$$

$$\begin{aligned} &if\ c_j\ is\ a\ Light\ C-node\ , i.e.\ y_j = \phi_{j,k},\ k \in \Gamma_j, \\ &\quad x_k = 1, \\ &\quad x_i = 0, \quad \forall i \in \Gamma_j, i \neq k. \end{aligned} \tag{6}$$

2) ***Edge Removal, Check Node Update and Iteration***: After the variable nodes are recovered from Zero and Light $C-nodes$ at the first phase recovery, edges incident to the corresponding $V-nodes$ are removed from the graph and subsequently the values of the neighboring $C-nodes$ are updated. We have $\forall j$ s.t. c_j is Zero or Light C-node and $\forall i \in \Gamma_j$

$$\begin{aligned} &Remove\ E_{ji}, \\ &Remove\ E_{qi},\ where,\ q \neq j,\ q = 1, 2 \ldots M \end{aligned}$$

$$\begin{aligned} &if\ x_i = 0 \Rightarrow y_q = y_q. \\ &elseif\ x_i = 1 \Rightarrow y_q = y_q - \phi_{q,i}. \end{aligned}$$

After edge removal and check node update processes, new Zero $C-nodes$ and Light $C-nodes$ may occur. Therefore, the FPR process, edge removal and check node update processes are iterated repeatedly until no more Zero $C-nodes$ and Light $C-nodes$ occur. The process of FPR, edge removal and check node update for Fig.1 is illustrated graphically below.

For the graph in Fig.1:

Zero $C-node$: c_1: $\Gamma_1 = \{1, 6, 8\}$

$y_1 = 0 \quad \rightarrow x_1 = x_6 = x_8 = 0.$
Edge Removal: Remove: $E_{1,1},\ E_{1,6},\ E_{1,8}$
Remove: $E_{4,6}$
Check Node Update: $x_6 = 0 \quad \rightarrow y_4 = y_4.$

Similar process is carried out for all Zero $C - nodes$.

For Light $C - node:\ c_3 : \Gamma_3 = \{3, 5, 12\}$
$y_3 = \phi_{3,3} \quad \rightarrow x_3 = 1,\ x_5 = x_{12} = 0.$
Edge Removal: Remove: $E_{3,3},\ E_{3,5},\ E_{3,12}$
Remove: $E_{6,12}$
Check Node Update: $x_{12} = 0 \quad \rightarrow y_6 = y_6.$

Similarly,

For Light $C - node:\ c_4 : \Gamma_4 = \{6, 9, 11\}$
$y_4 = \phi_{4,9} \quad \rightarrow x_9 = 1,\ x_6 = x_{11} = 0.$
Edge Removal: Remove: $E_{4,6},\ E_{4,9},\ E_{4,11}$
Remove: $E_{5,9}$
Check Node Update: $x_9 = 1 \quad \rightarrow y_5 = y_5 - \phi_{5,9}.$

It should be noted that at this point the Partial $C - node, c_5$, has changed to a Light $C - node$. The reduced graph after these steps is shown in Fig.2. The processes of edge removal, check

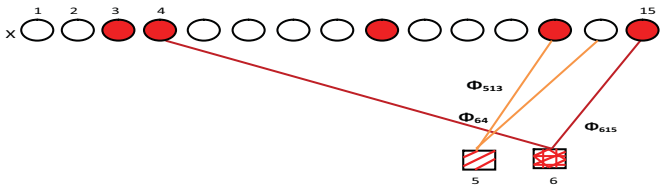

Fig. 2. Reduced Graph after the First cycle of First Phase Recovery and Edge Removal and Check Nodes Update

node update and Zero and Light $C-nodes$ recovery are continued as long as there exist single Zero or Light $C - nodes$. However, all $V - nodes$ may not be recovered by these processes. The $V-nodes$ which are neighbor of Heavy $C-nodes$ (e.g. c_6) have to be yet recovered. These nodes are recovered by the following method.

*3) **Check Sum Method for Heavy** $C - nodes$:* We use the *Unique Sum Property* of the sampling matrix Φ designed in Section (III-A) to recover the Heavy nodes and $V-nodes$ that are neighbors of Heavy nodes. The Check Sum method is described by following steps in Algorithm 1.

Algorithm 1 Check Sum Method for Heavy Nodes Recovery

1. $\forall$ Heavy $C - nodes\ c_j$,
2. Find Updated Γ_j
3. Find The numbers of elements in Γ_j, i.e. $|\Gamma_j| = \beta$
4. Create Vector $V_\phi = [\phi_{j,k}], \forall k \in \Gamma_j$.
5. Create Binary Matrix (BT) of size 2^β,
where the rows of BT correspond to the binary representation of 0 to $2^\beta - 1$.
6. Compute Check-Sum (SUM): $SUM = BT \times V'_\phi$
7. Find l s.t. $y_j = SUM(l)$
8. Assign element wise the l^{th} row of BT to the $V - nodes$ that are neighbors of c_j.

To illustrate this process for Fig.1, we can see from Fig. 2 that the c_6 is a Heavy $Check - node$. Following the steps described in Algorithm 1 we have updated $\Gamma_6 = \{4, 15\}$ and $\beta = 2$. Accordingly,

$$V_\phi = [\phi_{6,4} \quad \phi_{6,15}], \quad BT = \begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}, \text{ and}$$

$SUM = [0 \quad \phi_{6,15} \quad \phi_{6,4} \quad (\phi_{6,4} + \phi_{6,15})]'$.
Clearly, $y_6 = SUM(4) \rightarrow [x_4\ x_{15}] = [1\ 1]$.

IV. Analysis and Discussion

A. Choice of Row-Weight (L)

In our proposed BCS, the number of Zero $C - nodes$ and Light $C - nodes$ at the beginning of the first phase of recovery determines how many edges will be removed from the bi-partite graph at the first iteration. The probabilities of having Zero $C - nodes$ and Light $C - nodes$ are both functions of the row weight L and the sparsity $S = \frac{K}{N}$ of the binary sparse signal. From this point of view, it is desirable to choose the row weight L such that the probability of check nodes being Zero or Light is maximized to facilitate FPR and edge recovery process. However, it is not feasible to maximize both simultaneously (due to space limit, we skip the proof). However, we note that a Light $C - node$ is more effective and preferred than a Zero $C - node$. The reason is that Light $C - nodes$ facilitates in creating new Light $C - nodes$ and Zero $C - nodes$ during edge removal process. A Partial $C - node$ which has two non-zero neighbors and one of the non-zero neighbors is neighbor of a Light $C - node$ turns into a new Light $C - node$ after edge removal and check node update processes. Therefore, we maximize the probability of having Light $C - nodes$.

Lemma 1: The optimal row weight L for the maximum number of $Light\ C - nodes$ is $1/S$.

Proof: Let P_1 denote the probability of a $C - node$ being a Light $C - node$. We have

$$P_1 = (1 - S)^{L-1} \cdot S \cdot \binom{L}{1} \tag{7}$$

Our goal is to maximize P_1. Therefore, $\frac{d}{dL}(P_1) = 0$ or

$$(1 - S)^{L-1} \cdot S + L \cdot S \cdot (1 - S)^{L-1} \cdot ln(1 - S) = 0,$$

$$L \cdot ln(1 - S) = -1.$$

Expanding $-ln(1 - S) = S + \frac{S^2}{2} + \frac{S^3}{3} + \ldots$ and for small S, $-ln(1 - S) \approx S$. Therefore,

$$L \approx 1/S. \tag{8}$$

■

B. The Number of Measurements (M)

It should be noted that the proposed BCS scheme can always recover the signal coefficients if each signal coefficient (V-node) is included in at least one check node. Based on this we can find the order of the required measurements.

Lemma 2: For successful decoding of binary sparse source from compressed measurement using graph and sum based decoding algorithm, the number of measurements M is of $\mathcal{O}(Klog(N))$.

Proof: For a bipartite graph with $N\ V - nodes$ and $M\ C - nodes$ with the $C - node$ degree of L, we need $M \times L$

of order $Nlog(N)$ for all $V-nodes$ to be sampled in the graph [13]. Hence,

$$M = \mathcal{O}\left(\frac{N}{L} logN\right).$$

Given that $L = \mathcal{O}(1/S)$, we have

$$M = \mathcal{O}\left(KlogN\right).$$

■

Lemma 3: The encoding requires the computational complexity of $\mathcal{O}(ML) = \mathcal{O}(NlogN)$.

C. The Error Rate (E.R) of Proposed BCS

Let the error rate of the proposed binary compressive compressive sensing algorithm for binary sparse signals be defined as the ratio of number of unrecovered variable nodes to the total number of variable nodes. Provided (4) is satisfied, in our scheme, a variable node cannot be recovered if and only if it is not sampled by any check nodes (compressed measurements). In other words, if the V-node is isolated in the corresponding bi-partite graph.

Lemma 4: Error rate of BCS is given by $E.R = \left(1 - \frac{L}{N}\right)^M$

Proof: The probability that a variable node is not sampled in one measurement is $1 - \frac{L}{N}$. Therefore, error rate i.e., the probability that a variable node is not sampled in any of the M measurements is given by

$$E.R = \left(1 - \frac{L}{N}\right)^M. \tag{9}$$

■

V. Simulation and Results

For simulation purpose, following parameters are considered unless stated otherwise. We take the number of variable nodes $N = 1000$ and the sparsity of the binary source $S = K/N = 0.1$. In edge recovery, the number of check nodes recovered in the first phase is an important factor. A check node is said to be recovered when we recover all the variable nodes associated with it (neighbors) and in turn remove all the edges associated with it. The more the number of $C - nodes$ recovered in first phase, the easier and quicker the overall recovery scheme. In Figs. 3 and 4 we compare the first iteration recovery of $C - nodes$ in BCS with that of in Binary Tree scheme in [8], [9]. Fig. 3 shows the

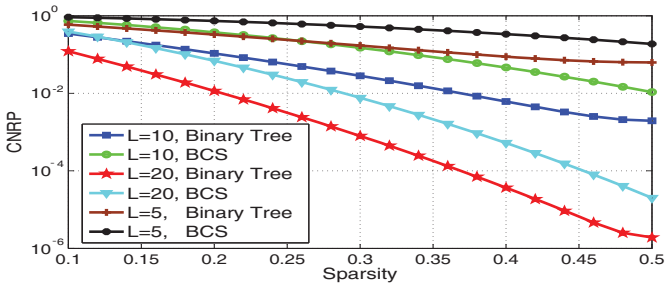

Fig. 3. Probability of Check Nodes Recovered in First Iteration (CNRP)

probability of check nodes recovered (CNRP) in first iteration for different sparsity and different row weight (L). It also shows that for the given L, the CNRP decreases as the sparsity increases. So it is necessary to decrease the row weight in high sparsity signal to increase the CNRP. However, we can clearly see that at each L and sparsity the CNRP of BCS is greater than that of in the Binary Tree scheme. To address the low CNRP for large sparsity it is desirable to take the row weight L of the sampling matrix of the order $\left(\frac{1}{S}\right)$ as discussed in IV-A. In Fig. 4, we show the number of check nodes recovered in first iteration for different sparsity rates. From Fig. 4 we can clearly see that in BCS, out

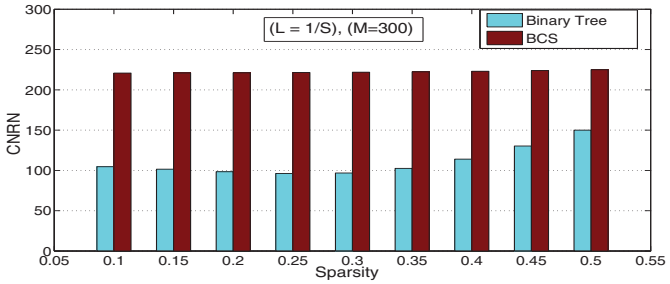

Fig. 4. Number of Check Nodes Recovered in the First Iteration (CNRN)

of 300 $C - Nodes$, about 225 are recovered in the first iteration itself whereas in Binary Tree scheme the number is as low as 100.

In Fig. 5(A), we can clearly see that for the given number of measurements, the Error Rate decreases as the row weight increases. However, this increase in performance is achieved by some cost in decoding time as shown in the Fig. 5(B). Fig.

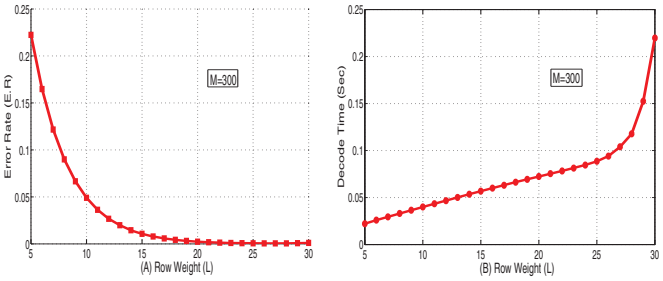

Fig. 5. Error Rate and Decode Time for different Row Weight

6 shows the performance of BCS for various row weights and sampling rates $(S.R = M/N)$. It is clearly seen that the Error Rate decreases as the sampling rate increases. We can also see that for the same sampling rate, the error rate decreases as the row weight is increased. We can clearly see from Fig. 6 that the

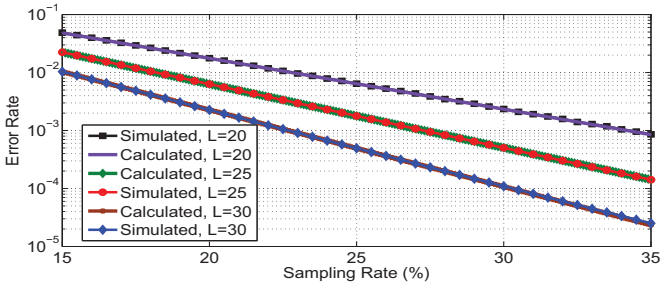

Fig. 6. Error Rate for different Sampling Rate and Row weight

simulated error rate is very close to the error rate calculated in the section IV-C.

In [6] the general l_1 minimization method is modified for the binary signal source. A new reconstruction constraint is used to bound the reconstructed signal values within, $0 \leq \hat{x} \leq 1$. However, in that scheme, the reconstructed binary signal values lie in the continuous range of $0 \leq \hat{x} \leq 1$ instead of giving discreet values 0 or 1. In Fig. 7 the original binary signal and reconstructed signal using the binary l_1 and BCS is shown. Fig. 7**(A)** is the original binary signal, **(B)** is the reconstructed signal using the binary l_1 method and **(C)** is reconstructed signal using the BCS scheme. We can clearly see that the BCS scheme has

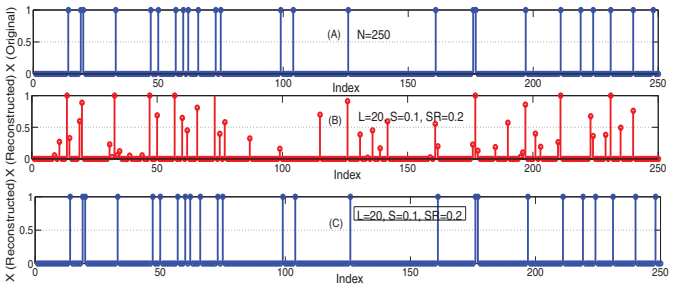

Fig. 7. Original and reconstructed signal using Binary l_1 and BCS.

better reconstruction than the l_1 binary scheme. Fig. 8 shows the error rate comparison of our scheme and the Binary l_1 scheme with threshold of 0.5 for decision making. We can clearly see that

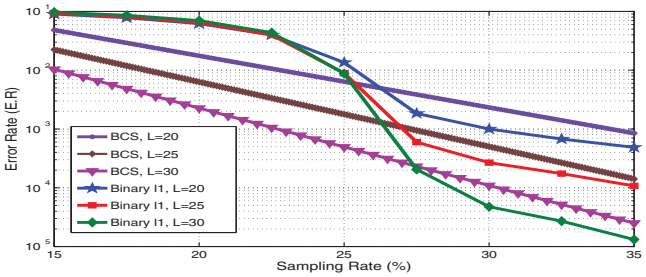

Fig. 8. Error Rate comparison between Binary l_1 scheme with threshold and BCS.

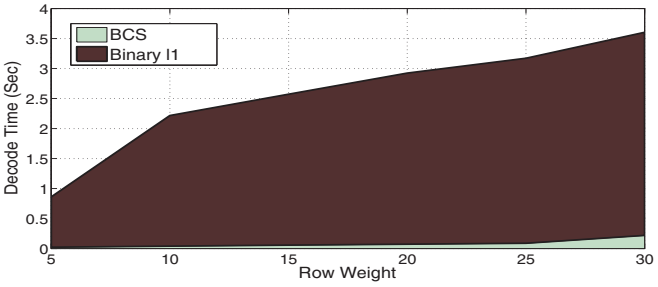

Fig. 9. Decoding time comparison between Binary l_1 scheme with threshold and BCS.

for row weight of 30 at low sampling rate of 25%, the error rate of BCS is in the order of 10^{-4} whereas, the error rate in binary l_1 method is in the order of 10^{-2}. At very low sampling rate BCS has very good performance compared to the binary l_1 scheme. When the sampling rate is increased binary l_1 has slightly better performance than BCS, however, the error rate in both schemes are in the same order. It should be noted that the error rate in BCS is solely because of the un-covered (isolated) $V-nodes$ unlike in Binary l_1, which is due to the wrong decoding. Moreover, Fig. 9 shows that BCS takes comparatively very less time for decoding. The small gain in error rate performance in binary l_1 is achieved at very high cost of decoding time.

VI. Conclusion

In this paper, we presented a novel compressive sampling and decoding method for Binary Sparse Signal using Graph and unique sum method called Binary Compressive Sensing (BCS). We modeled the binary compressive sensing method as the bipartite graph and implemented the edge recovery scheme. Edge recovery scheme in itself cannot guarantee the complete decoding. To overcome this drawback, we designed a unique sampling matrix for the binary sparse source and implemented unique sum method to completely recover all variable nodes and provided mathematical analysis of the proposed BCS scheme. We compared our scheme with Binary tree recovery method and Binary l_1 method and confirmed the superiority of BCS. BCS can be used in many practical applications such as event detection in wireless sensor networks, spectrum occupancy decision making, binary images and other applications with binary sparse signal.

VII. Acknowledgement

This material is based upon work supported by the National Science Foundation under Grants ECCS-1056065 and CCF-0915994.

References

[1] D. Donoho, "Compressed sensing," *IEEE transaction on information theory*, vol. 52, pp. 1289–1306, April 2006.

[2] R. Baraniuk, "Compressive sensing," *Lecture Notes in IEEE Signal Processing Magazine*, vol. 24, July 2007.

[3] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory,*, vol. 52, no. 2, pp. 489 – 509, 2006.

[4] R. Baraniuk, M. Davenport, R. Devore, and M. Wakin, "A simple proof of the restricted isometry property for random matrices," *Constr. Approx*, vol. 2008, 2007.

[5] J. Meng, W. Yin, H. Li, E. Houssain, and Z. Han, "Collaborative spectrum sensing from sparse observations using matrix completion for cognitive radio networks," in *,2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pp. 3114 –3117, march 2010.

[6] M. Stojnic, "Recovery thresholds for l1 optimization in binary compressed sensing," *International symposium in information technology (ISIT).*, pp. 1593–1596, June 13-18 2010.

[7] O. L. Mangasarian and B. Recht, "Probability of unique integer solution to a system of linear equations," *European Journal of Operational Research, doi:10.1016/j.ejor.2011.04.010*, 2011.

[8] F. Wu, J. Fu, Z. Lin, and B. Zeng, "Analysis on rate-distortion performance of compressive sensing for binary sparse source," in *Proceedings of the 2009 Data Compression Conference*, (Washington, DC, USA), pp. 113–122, IEEE Computer Society, 2009.

[9] J. Fu, Z. Lin, B. Zeng, and F. Wu, "Tree structure based analyses on compressive sensing for binary sparse sources," in *Proceedings of the 2010 Data Compression Conference*, DCC '10, (Washington, DC, USA), pp. 530– , IEEE Computer Society, 2010.

[10] M. Luby, M. Mitzenmacher, M. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," *IEEE Transactions on Information Theory,*, vol. 47, pp. 569 –584, feb 2001.

[11] N. C. Wormald, "Differential equations for random processes and random graphs," *The Annals of Applied Probability*, vol. 5, pp. 1217 –1235, Nov 1995.

[12] S. Sarvotham, D. Baron, and R. G. Baraniuk, "Sudocodes – fast measurement and reconstruction of sparse signals," in *IEEE International Symposium on Information Theory – ISIT*, (Seattle), Jul. 2006.

[13] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory,*, vol. 52, pp. 2551 –2567, June 2006.