Coded Compressed Sensing Scheme for Unsourced Multiple Access

CS754 Advanced Image Processing

Rathour Param Jitendrakumar, 190070049
Satush Parikh, 21D070062

Indian Institute of Technology Bombay
https://github.com/paramrathour/Coded-Compressed-Sensing-for-Unsourced-Multiple-Access

Spring 2023-24

Guide: Prof. Ajit Rajwade

Outline

Implementation

Unsourced Multiple Access

- Problem
  - Transmitting messages to the access point in ann uncoordinated fashion
  - All users employ a common codebook
  - The receiver decodes up to a permutation of the messages
  - Unsourced multiple access (UMAC) - the use of a unique code does not allow to distinguish the transmitters identity.
- Assumption
  - Active devices pick their information message independently and uniformly at random from the set of binary sequences $\{0, 1\}^B$.

System Model

- $\mathbf{S}_{\mathrm{a}} \subset \mathbf{S}_{\mathrm{tot}}$ - collection of devices within a network (Cardinality K_{tot})
- $\mathbf{S}_{\mathrm{a}}$ - the subset of active devices within a communication round (Cardinality K_{a})
- Every active device wishes to communicate B bits of information to a base station and, these data transfers are Decentralised (uncoordinated).
- N - number of channel uses is N
- $W = \{\underline{w}_i : i \in \mathbf{S}_{\mathrm{a}}\}$ - set of B-bit message vectors associated with active devices.
- Performance objective (per-user error probability)

$$P_{\mathrm{e}} = \frac{1}{K_{\mathrm{a}}} \sum_{i \in \mathbf{S}_{\mathrm{a}}} \Pr\left(\underline{w}_i \notin \widehat{W}(\underline{y})\right)$$

System Model

Motivation for CS

- Signal available at receiver

$$\underline{y} = \sum_{i \in \mathbf{S}_{\mathrm{a}}} \underline{x}_i + \underline{z},$$

- $\underline{x}_i$ - N-dimensional vector transmitted by device i,
- $\underline{z}$ represents additive white Gaussian noise with covariance $\sigma^2 \mathbf{I}$.
- $\widehat{W}(\underline{y})$ is an estimate the list of transmitted binary vectors based on the observed signal
-
$$\underline{y} = \mathbf{X}\underline{b} + \underline{z},$$

  where $\mathbf{X} \in \mathbb{R}^{N \times 2^B}$ denotes the common codebook and $\underline{b} \in \{0,1\}^{2^B}$ is a binary vector .
- $\|\underline{b}\|_0 = K_{\mathrm{a}}$.
- X playing the role of a sensing matrix and $\underline{b}$ being an unknown K_{a}-sparse vector.

Notation

Notation	Parameter Description
K_{tot}	Total number of users in the system
K_{a}	Number of active users
B	Message length in bits
N	Number of channel uses per round
ε	Maximum tolerable probability of error per user
n	Number of coded sub-blocks per round
J	Number of coded bits per sub-block,
M	Total number of coded bits, $M = nJ$
P	Total number of parity-check bits, $P = M - B$
$\varepsilon_{\mathrm{tree}}$	Maximum probability of error for tree decoding
C_{cs}	Computational complexity of CS sub-problem
C_{tree}	Computational complexity of tree decoding
b_j	Number of information bits in jth sub-block
l_j	Number of parity bits in jth sub-block
K	Size of output list for CS sub-problem

Figure: Notation [1]

[1] "A Coded Compressed Sensing Scheme for Unsourced Multiple Access" Vamsi K. Amalladinne IEEE TIT 2020

Big Picture

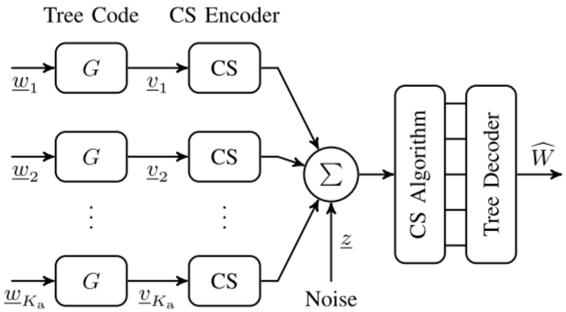


Fig. 1. This schematic diagram captures the overall architecture of the proposed scheme. The information bits are split into sub-blocks, and redundancy is added to individual components. Transmitted signals are then determined via a CS matrix, and sent over the MAC channel. A CS algorithm recovers the lists of sub-blocks, and a tree decoder reconstructs the original messages.

Figure: Architechture[2]

[2] "A Coded Compressed Sensing Scheme for Unsourced Multiple Access" Vamsi K. Amalladinne IEEE TIT 2020

Encoding

- Tree Encoding

$$\underline{p}(j) = \sum_{\ell=0}^{j-1} \underline{w}(\ell)\, G_{\ell,j-1}$$

$$\underline{v} = \underbrace{\underline{w}(0)}_{\underline{v}(0)}\, \underbrace{\underline{w}(1)\underline{p}(1)}_{\underline{v}(1)} \cdots \underbrace{\underline{w}(n-1)\underline{p}(n-1)}_{\underline{v}(n-1)}.$$

- CS Encoding

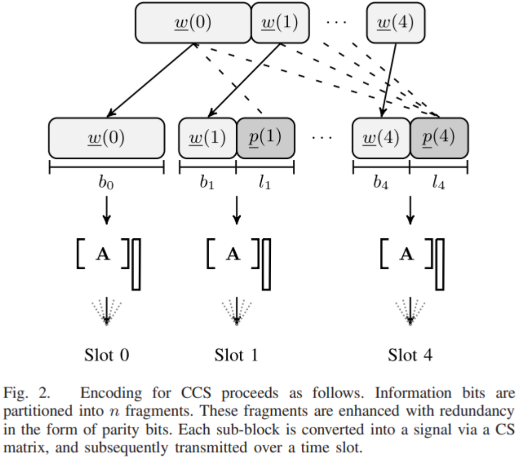


Figure: Encoding[a]

[a] "A Coded Compressed Sensing Scheme for Unsourced

Tree Encoding

Optimization Framework

- B-bit binary message partitioned into n sub-blocks, where the jth sub-block consisting of b_j message bits, with$\sum_{j=0}^{n-1} b_j = B$.
- $\underline{w} = \underline{w}(0)\underline{w}(1)\cdots\underline{w}(n-1)$.
- The tree encoder appends l_j parity check bits to sub-block j, total length of every sub-block to $b_j + l_j = J = M/n\ \bar{b}its$
- $b_0 = J\ l_0 = 0$ For subsequent subblocks, the parity bits are constructed as follows.

$$\underline{p}(j) = \sum_{\ell=0}^{j-1} \underline{w}(\ell) G_{\ell,j-1}$$

$$\begin{aligned} \min_{(p_1,\dots,p_{n-1})} \quad & \mathbb{E}[\tilde{C}_{\text{tree}}] \\ \text{subject to} \quad & \mathbb{E}\left[\tilde{L}_{n-1}\right] \leq \varepsilon_{\text{tree}} \\ & \sum_{j=1}^{n-1} \log_2\left(\frac{1}{p_j}\right) = M - B \\ & p_j \in \left[\frac{1}{2^J}, 1\right] \quad \forall j \in [1:n-1]. \end{aligned}$$

$$p_\ell = 2^{-l_\ell}.$$

Decoding

CS Decoding

- The aggregate signal received at the base station during the jth sub-block can be expressed as $y(j) = \mathbf{A}r(j) + z(j)$, where $r(j)$ is a K_a-sparse binary

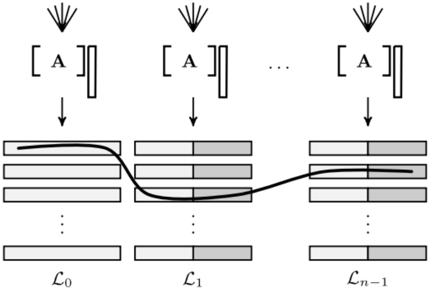
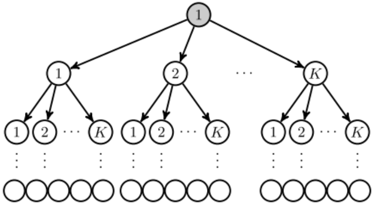


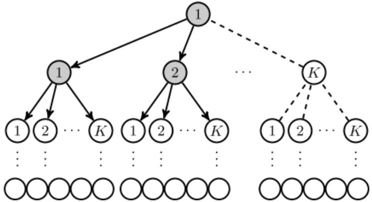
Figure: Decoding[a]

[a] "A Coded Compressed Sensing Scheme for Unsourced Multiple Access" Vamsi K. Amalladinne IEEE TIT 2020

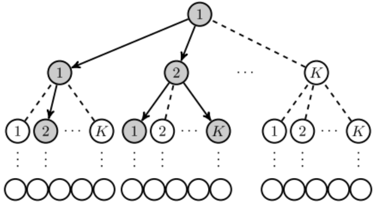
Tree Decoding



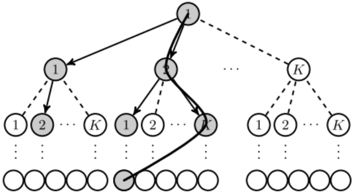
(a) Stage 0: Processing one element from $\mathcal{L}_0$ at a time, a fragment is selected as the root node of a tree.



(b) Stage 1: Fragments in $\mathcal{L}_1$ act as the children of the root node. Parity requirements are checked and only complying nodes, nodes 1 and 2 highlighted in the figure, are retained.



(c) Subsequent Stages: Candidate fragments from a subsequent stage become the children of complying nodes. Again, parity constraints



(d) Last Stage: Parity constraints are verified at the leafs. A valid message on the CS tree will survive, but decoding is successful only if no other paths meet its parity requirements. We highlight the legitimate path in black above.

Implementation Details

- Message Generation - Selection of Data Structures
- Tree Encoding - Optimisation of parity-bits length using `CVXPY` framwork
- CS Encoding - Sensing Matrix generation using BCH codes
- CS Decoding - Orthogonal Matching Pursuit
- Tree Decoding - Backtracking with pruning

Simulation Details

- According to the reference
- $K_a = 25$
- $B = 101$
- $n = 11$
- $J = 15$
- $\varepsilon_{\text{tree}} = 0.0025$
- $(2047, 23)$ BCH Codebook

Challenges

- Infeasibility of the parity length optimisation for some $varepsilon_{\text{tree}}$
- Failure of the total bits constraint due to rounding of parity lengths
- Vectorisation of the otherwise inefficient tree encoding code
- Edge cases in tree decoding due to missing data
- Slowed working of CS decoding for higher J

Results

Optimization Framework

$\varepsilon_{\text{tree}}$	$\mathrm{E}[\bar{C}_{\text{tree}}]$	Parity Length Vector
0.0001	Infeasible	
0.001	555	0 4 5 5 5 5 5 5 6 8 16
0.0012	516	0 4 5 5 5 5 5 6 6 7 16
0.0015	493	0 3 5 5 6 6 6 6 6 6 15
0.0020	477	0 4 5 5 5 6 6 6 6 6 15
0.0025	466	0 4 5 5 6 6 6 6 6 6 14
0.006	429	0 4 5 6 6 6 6 6 6 6 13
0.008	419	0 4 6 6 6 6 6 6 6 6 12
0.02	392	0 5 6 6 6 6 6 6 6 6 11

Table: Error Probability is minimized when Parity Check Bits are Pushed to End, whereas Average Computational Complexity is Least when equal parity-check bits are allocated per sub-block

Trade-Offs

- As n increases, computational complexity decreases but the error probability increases
- In parity length optimisation, allocating parity bits towards the later stages of CCS improves performance at the expense of complexity

Results

Simulation

For the mentioned parameter set,

- Complete recovery of messages wasn't possible
- 30%-40% of bits recovered partially
- Hence efficient Code design at client level needed

```
Can't recover any message fully, trying partial recovery...

Found partial messages...
[array([ 0,  2,  8, 32, 19, 10, 22, 14, 15]), array([ 1,  2, 22,  6,  0]), array([ 2, 22]), array([ 3, 11, 22, 19,
22]), array([ 4,  1, 19,  0, 21, 27]), array([ 5,  7, 25]), array([ 6, 34, 17]), array([ 7,  4, 28, 26, 22, 17,
29, 26]), array([ 8,  4,  3,  7, 13, 20]), array([ 9, 16, 32, 32,  0, 14,  6]), array([10, 17,  7]), array([11,
2, 13, 33, 33]), array([12, 31, 27, 31, 18, 18, 30]), array([13, 33, 17, 29,  8]), array([14, 26, 21, 23]),
array([15,  6, 28,  0]), array([16,  0, 15,  6, 15]), array([17, 10,  4, 11, 27, 12, 10, 30]), array([18,  1,  7,
13, 21]), array([19, 17, 19,  2,  9,  6, 12,  5, 31]), array([20, 29, 13, 20, 27,  6, 16, 21]), array([21, 18, 27,
12]), array([22, 18, 26, 30,  7, 20, 19,  5]), array([23, 19, 18, 16]), array([24, 16,  5, 18,  0, 21]),
array([25, 18]), array([26, 24,  3, 26,  2, 11, 24, 21, 25, 13]), array([27, 29, 19, 10]), array([28, 33, 27,  0,
32,  2]), array([30, 26, 14, 22, 14, 31, 14]), array([31,  1,  5,  2, 21,  2, 21, 33, 16, 24]), array([32,  2,  9,
17, 11, 19, 34]), array([33, 17, 20, 29,  8, 14,  1]), array([34,  5, 23, 20,  6,  8])]

Each message size 101

0 messages received completely []

At least 24 messages received partialy

25 messages sent
```

Future Work

- Detailed analysis of the algorithm for varying #users with a better metric such as E_b/N_0
- Comparison with other state-of-the-art techniques such as ALOHA and SIC
- Hyperparameter tuning to obtain optimal results

References

Vamsi K. Amalladinne, Jean-Francois Chamberland, and Krishna R. Narayanan.
A coded compressed sensing scheme for unsourced multiple access.
IEEE Transactions on Information Theory, 66(10):6509–6533, 2020.
doi:10.1109/TIT.2020.3012948.

Contribution

- Param: Decoder, Message Generation, overall code maintainence with vectorisation
- Satush: Encoder, Parity Length Optimisation, Sensing Matrix Generation