1. INTRODUCTION

Distributed Arithmetic Coding (DAC):
- A variant of the arithmetic coding (AC) that can be used to perform lossless distributed source coding

Open Problems in DAC:
- DAC’s decoding complexity
- How fast the complexity of the full-search DAC decoder grows with respect to code

Previous Work (Fang et al. 13, Fang et al. 14, Fang et al. 15):
- Codebook Cardinality Spectrum (CCS)
- Hamming distance (H-distance) spectrum (HDS)
- Breadth-First Decoder of DAC (BFD)

Drawbacks of BFD:
- There is a risk that the optimal path is mis-pruned when its partial metric is inferior to other paths
- To achieve good performance, a large amount of paths must be maintained during the decoding, which imposes a heavy burden on the decoder

Contribution:
- First realization of depth-first DAC decoder
- Experiments show that under the same complexity constraint, the depth-first decoder (DFD) outperforms the BFD, if the code is not too long and the SI quality is not very poor.

2. Review on Breadth-First DAC Decoder

Problem Formulation:
- Assume that the source emits $X^n = x^n$, which is encoded at rate $R$ to get $M^n$. If $R < 1$, the SI $Y^n$ is correlated with $x^n$ and is necessary at the decoder for the lossless recovery of $x^n$. On receiving $m$, the decoder tries to find the binary vector best matching $y^n$ from all solutions to $2^m(1^n) = m$, where $s^n \in \mathbb{B}^n$. Then DAC decoding can be formulated as

$$\hat{x} = \arg \min_{s^n} d_H(s^n, y^n), \text{ s.t. } |2^m L(s^n)| = m \quad (1)$$

Construction of DAC Tree:
- We define the following vector

$$u_i(s) = (u(s'), \ldots, u(s^{i-1})) \quad (2)$$

where $s_{ij}$. If $i \neq 0$, the subscript is dropped for simplicity. For boy nodes, i.e., $i \in \{0, (n-i)\}$, we have

- If $u(s') = 0$ or $(1-2^{-j})$, node $s$ has only 0-child
- If $u(s') = 0$ or $(1-2^{-j})$, node $s$ has both 0-child and 1-child, which causes branching
- If $u(s') = 2^{-j} \ldots, 1$), node $s$ has only 1-child

for $i \in \{0, (n-i)\}$, if $u(s') \in [0, 0.5]$ node $s$ has only 0-child; otherwise, node $s$ has only 1-child. So there is no branching at tail nodes.

3. Depth-First DAC Decoder

Principle of Depth-First DAC Decoder:
- The principle of DFD can be illustrated by above figure

The SI is assumed to be 000010. The initial pass proceeds along the black full path 111010. After the initial pass, the decoder records the path-SI H-distance: $d_{min} = d_{H}(100010, 111010) = 2$. There are 3 fork nodes along the path 111010, so 3 unequal-length paths (10, 111, and 11010) are suspended, whose end nodes are marked with red color. Note that, branch picking/storing at fork nodes is based on overall path metrics rather than SI, because the last three nodes (marked with green color) have only 0-child; otherwise, node $s$ has only 1-child. So there is no branching at tail nodes.

4. Experimental Results

Experimental Results Demonstrate How Tall Length, Code Length, and SI Quality Impact the DFD and the BFD that are Subject to Equivalent Constraints.

5. CONCLUSION AND SUMMARY

This research work presents a depth-first decoding algorithm for distributed arithmetic codes under uniform binary sources.

The DFD’s complexity can be lowered by enhancing the SI quality: the better SI, the lower complexity.

Compared with the BFD, the DFD performs better for short and medium code lengths and in situations when SI quality is not too poor.