Coded Modulation with Lattices and Reed-Solomon Codes

Propose coded modulation system using lattices and a systematic (N,K) Reed-Solomon code over GF(2^m)

- Each (GF(2^m)) symbol corresponds to one group of flash cells.
- Only encode mod 2 data values (increases the rate) — lattice Euclidean decoding.
- Lattice decoding errors are buried, so Reed-Solomon codes are well suited.
- For flash memories, Reed-Solomon codes have lower decoding complexity than BCH codes [Chen et al., 2008].

Rewriting Codes Using Lattices

Rewriting codes allow writing flash memories two or more times without erasing. Lattices can be used to construct rewriting codes.

The main idea is to use a code-to-many mapping from information to lattice points.

Cookbook mapping
• Choose parameter M ≤ V,
• Information is encoded in V bits, B = {0, 1, ..., 2^M - 1}
• "Codebook" colored in V bits C, C = {0, 1, ..., 2^M -1}
• If C < M, then mapping is one-to-one.
• The following encoding mapping is needed:

Dirty Paper Coding” for Rewriting Flash

Dirty paper (DPC) encoding:
- u is colored in base codebook.
- Known interference s, is current state of memory.
- "Transmitted" codeword is u = mod B = which is positive-valued.
- The value in memory is: u = u - mod B = u - "+
- Decoding in absence of noise: u = mod B

Numerical Results

DPC system with E8 lattice:
- BASE code only: V = M
- DPC: V = 2M
- Interested in high-rate codes suitable for applications.

Soft-Input Architecture

Conventional flash memory architecture:
- Hard decisions made internally, ECC performed externally.
- Typical values: 4 values.

On-chip soft-input decoding (e.g., LPCD) is difficult to perform on-chip.
- Soft-input lattice decoding is more powerful than simple hard decisions.
- Lattice decoding is less complex than LPDC, can be performed on-chip.
- External ECC can operate on hard decision values.

Soft-Input Architecture Model

On-chip soft-input decoding (e.g., LPCD) is difficult to perform on-chip.
- Soft-input lattice decoding is more powerful than simple hard decisions.
- Lattice decoding is less complex than LPDC, can be performed on-chip.
- External ECC can operate on hard decision values.

Storing Lattice Values in Flash

The values of an n-dimensional lattice are stored in a flash cell:
- Assume signal between 0 and V (i.e., other systems quantize to k levels)
- Codebook C is the lattice points inside side length M cube
- If G is lower triangular, then mapping B = {0, 1, ..., 2^k} → Z^k is efficient

Complexity of E8 Lattice Decoding

Two algorithms exist to find the E8 lattice point closest to u ∈ k.

Closest Decoding (about 304 steps) / (x, y) is a rotated to nearest integer: (x, y) has least reliable position rounded "wrong way."

y_1 = floor(x) + floor(y) + 1 if (x + y) is even otherwise.

y_2 = floor(x) + floor(y) - 1 if (x + y) is even otherwise.

Construction A" Decoding (about 72 steps)
1. Find y and r ∈ Z^2 such that x ≤ y + 1 and 1 ≤ y ≤ 1
2. Set the sign of r, for which x + 0 < x ≤ 3. For x ≤ 0, replace x by 2 - x.
3. Decode y as a first-order Reed-Muller code of length 5. Output x.
4. For r = 0, change x, r ∈ Z^2, Output r + 2a.

More work has been done on "The E8 Lattice and Error Correction in Flash Level Flash Memory," is appear in Proceedings of the IEEE 2011.

Summary

This paper proposes using lattices to encode data in flash memories:
• For error correction, lattices combined with Reed-Solomon codes form a coded-modulation system that have about 1.7 to 1.9 dB lower SNR than existing BCH code systems.
• For rewriting flash memories, rewriting codes can be constructed from lattices at high rates.

Lattices for Error Correction and Rewriting in Flash Memories

Testbed allows to use lattices in the Flash modeling based upon lattices, and an example using the E8 lattice. --