Today

Start of Algorithmic Phase

- Algorithmic tasks
- Easy tasks for linear codes
- Decoding Reed-Solomon Codes

Algorithmic Tasks: Broadly

- Encoding
- Decoding
- Construction

Encoding

- Given message compute its encoding.
- More or less straightforward.
- Only issue how is a code specified?

Specification/Construction of Code

- Asked this question when we asked what is explicit construction.
- What is a construction? Won't debate here, since most results are reasonable. Possibilities include:
  - Given n, k, q spit out Encoding circuit (implicitly). (More in line with Shannon theory).
  - Given n, k, q spit out a recognizer circuit for codewords. (More what Hamming theory would like?)
  - For linear codes, above are polytime equivalent.

Decoding

Much more tricky issue: What is the target? What is acceptable? When/How is code specified?

- Problems too hard if code is specified as part of the input. Don't know of any efficient algorithm.
- Know of many hardness results.
- So will only focus on specific codes.

Popular definitions in decoding

Maximum Likelihood Decoding (MLD) (Shann theory)

- Given Channel (bipartite graph/Markov chain) and some distribution on messages (say uniform)
- Compute most likely message/codeword, given received vector.

Nearest Codeword Problem (NCP) More combi

- Given received vector $r \in \Sigma^n$ find $c \in C$ nearest to r.
- Corresponds to MLD for q-ary symmetric channel.
- What happens with ties?

Popular definitions in decoding (contd.)

Soft-decision Decoding
- Given $n \times q$ matrix R of non-negative reals, (rows indexed by elements of Σ), compute codeword $c \in C$ that maximizes $\sum_{i=1}^{n} M_{i,c_i}$.
- What? Think M as being 0/1 with one 1 per column. Then, get the NCP problem.
- Name comes from process generating this matrix. Given MLD problem, with i.i.d. channel, for each rec'd symbol compute $P_{i,\alpha}$, probability that we received what we received given the transmitted symbol on ith coordinate is α. Now let $M_{i,\alpha} = -\log\{P_{i,\alpha}\}$. Maximizing codeword in the one that is maximum likelihood.

- Soft-decision decoding solves MLD for i.i.d. channel.

Problems with definitions so far: All too hard even for reasonable codes.

Reasonable decoding problems

Try to decode small number of errors. Less than d, for sure. Less than $d/2$?

Unambiguous/Unique decoding Given $r \in \Sigma^n$ compute $c \in C$ such that $\Delta(r, c) < d/2$ if such c exists (where $d = \Delta(C)$). Sometimes called Bounded Distance Decoding.

Relatively Near Codeword (RNC) Parameterize Problem with Parameter $\gamma > 0$. Given $r, e < \gamma d$, find c with $\Delta(r, c) \leq e$ if one such exists. (Ties? Any such c!)

List-decoding problem Similar to RNC, but now want a list of all codewords c such that $\Delta(r, c) < \gamma d$.

Reasonable decoding problems (contd.)

- Unambiguous decoding is RNC/List-decoding for $\gamma = \frac{1}{2}$.
- Complexity of RNC increases with γ. $\gamma = \infty$ becomes NCP (so not reasonable). In my opinion, problem is reasonable up to $\gamma = 1$.
- For every γ, RNC reduces to list-decoding. List-decoding is clearly hard if list-size exponential. So should only attempt to solve this up to the list-decoding radius of a code. Don't quite know this quantity for any reasonable code, so we try up to known lower bounds on the LDR (Johnson bound).

So many problems ...

- Main point: More problems than we can deal with. Must be careful when trying to interpret literature.
- Our focus simple
  - Unambiguous decoding
  - List decoding typically up to Johnson bound.

Some observations for Linear Codes

Assume Generator matrix is given, and "easy" is poly time.

- Encoding easy.
- Error-detection easy.
- Erasure correction easy.
  - Given $r \subseteq (\Sigma \cup \{?\})^n$, compute $c \in C$ such that $r_i \neq ? \Rightarrow r_i = c_i$.
  - Amounts to solving linear system. If system has no solution, can tell. If it has one solution, can find. If it has many solutions, can list all implicitlt in the form $Ax + b$.

– Food for thought: How many erasures can you correct with code of distance d?

Syndrome Decoding

- Commonly used term.
- Term meaningful but not necessarily useful.
- Given vector $r = c + e$, $rH = eH$ is called the "syndrome" of the error, since it depends only on the error and not on codeword/message.
- "Syndrome decoding" could mean one of

1. The Problem: Compute e from eH.
2. The Brute Force Algorithm: Build table of map $eH \mapsto e$, and look up table to compute e.

- Uselessness of Problem: Any decoding algorithm for linear code implies syndrome decoding algorithm.
- Uselessness of Algorithm: Exponential time.

Brute-Force Decoding Algorithms

- Enumerate all codewords and measure distance.
- Enumerate all error patterns and check if it leads to codeword.
- Syndrome decoding/Table lookup.
- All run in exponential time for asymptotically good families of codes.

Unambiguous Decoding of Reed-Solomon Codes

Problem Statement

Given Distinct points $\langle(\alpha_i, r_i) \in \mathbb{F} \times \mathbb{F}\rangle_{i=1}^n$, and parameter k

Task Compute (coefficients of) polynomial p of degree at most k such that $p(\alpha_i) = r_i$ for at least $(n+k)/2$ values of $i \in [n]$.

Convoluted history

- No a priori reason why brute-force is not best!
- Polynomial time solution found essentially concurrently with definition of Reed-Solomon codes (or even earlier!).
- How?
  - 1958/59: BC + H discover binary BCH codes.
  - 1960: Peterson discovers polytime decoding algorithm for binary BCH codes.
  - 1960: Reed-Solomon discover RS codes. (But no connection known to BCH codes).

  - 1963: Gorenstein-Zierler discover q-ary BCH codes and "note" RS codes are a special case of q-ary BCH codes, and that Peterson's algorithm generalizes to q-ary BCH codes.
  - Much later: BCH codes described as Subfield subcodes of RS codes.
- Note: Peterson's algorithm discovered before definition of the class P. One line justification by Peterson. (Edmonds/Cobham 1965).
- Good news: RS codes have decoding algorithm. Bad news: Decoding algorith too complicated to explain. Have to take duals of codes twice!
- Fortunately, cured by Welch-Berlekamp.

We'll follow their proof as described in Gemmell-Sudan.

- Many speedups of the Peterson-Gorentstein-Zierler algorithm known, including the famed Berlekamp-Massey algorithm.

Key concept: Error-locator Polynomial

- Given $\langle(\alpha_i, r_i)\rangle_i$ s.t. $\exists p$ of deg. k agreeing with seq. on $(n+k)/2$ points, a polynomial $E(x)$ is an error-locating polynomial if:
  - $p(\alpha_i) \neq r_i$ implies $E(\alpha_i = 0)$.
  - E is not zero "too often" (at least $k+1$ non-zeroes).

- Simple Fact: Given an error-locator polynomial E, can compute p efficiently.

- Simple fact: Such an E of degree e (# errors) exists $E(x) = \prod_{i|r_i \neq p(\alpha_i)}(x - \alpha_i)$.

- Question: How to find E?

- Grammatical aside: "Key" is an adjective, not a noun.

Key equation & Algorithm

- Grammatical aside: "Key" is an adjective, not a noun.

- Fix E of degree e and p and let $N(x) = p(x).E(x)$.

- Then (the key equation)

$$\forall i, \quad N(\alpha_i)(= p(\alpha_i)E(\alpha_i)) = r_i E(\alpha_i).$$

- Algorithm:

1. Find (N, E) with $(N, E) \neq (0, 0)$ and $\deg(N) \leq k + e$ and $\deg(E) \leq e$ satisfying key equation.

2. Output N/E if it is a polynomial satisfying the right conditions, else say none exists.

- Over time, key equation became Key equation.

Analysis

- Why can we find such a pair (N, E)?
  - Substitute unknowns for coefficients.
  - Solve linear system!
- Why does a solution exist? We just argued it!
- Why is it unique?
  - It is NOT!
  - But any solution will do.

Analysis (contd.)

- Claim: If (N, E) and (M, F) are both solutions to Step 1, then $N/E \equiv M/F$.
- Proof:
  - $\forall i, \quad r_i N(\alpha_i)F(\alpha_i) = r_i M(\alpha_i)E(\alpha_i)$.
  - If $r_i \neq 0$ then can cancel from both sides above to get $N(\alpha_i)F(\alpha_i) = M(\alpha_i)E(\alpha_i)$.
  - If $r_i = 0$ then $N(\alpha_i)F(\alpha_i) = M(\alpha_i)E(\alpha_i) = 0$.
  - So for n values, we have $N \cdot F = M \cdot E$.
  - If $n > k + 2e$ then $N/E \equiv M/F$.

Summary

- Gives polytime algorithm for decoding up to error-correction capacity of code.
- Highly non-trivial result - no reason to exist!
- Algebra often has non-trivial solutions to seemingly hard problems. Have to be very careful when basing cryptography on it.