Advanced Algos 4/30.

Error Correcting Codes

Berlekamp-Welch

Source $\longrightarrow$ destination

message $\in \Sigma^k$ ← alphabet.

Code: encode $E : \Sigma^k \to \Sigma^n$

k = orig. # char.

n = length of msg.

$n > k$.

decode: $D : \Sigma^n \to \Sigma^k$

property: $\forall (f_0, \ldots, f_{k-1}) = f$

$$D(E(f) + \underline{\text{corruption}}) = f.$$

1) erasure: replace e chars with ? $0 \to ?$

today

2) error: e chars got changed. don't know which $0 \to 1$

3) deletion / insertion a message $\Sigma^n \to$ into Σ^{n+1}

$000 \to 0100$
$\to 00$

Motivations

- communication
- storage: fault-tolerant

storing k files on k hard drives

naive way to make f.t.: repetition (code):

repeat each file on $\alpha > 1$ hard drives.

$n = \alpha \cdot k$. (# hd)

if $\alpha - 1$ hd fail, still ok.

erasure corruption

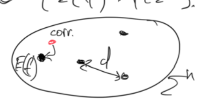
Can we do better?

yes: want $n = \alpha \cdot k$, α = const.

to be resilient to many more errors?

Def: code $\mathcal{C} \subset \Sigma^n$

$$\mathcal{C} = \{E(f) : f \in \Sigma^k\}.$$



Natural decoding: to map $m \in \Sigma^n$ into closest code $\in \mathcal{C}$ and then decode f.

when can we decode (how many coord. can be affected?)

1) erasure

erasure errors can tolerate is $\leq d - 1$ ← min dist in $\mathcal{C}$

$$d = \min\{\text{dist}(x, y) : x, y \in \mathcal{C}, x \neq y\}$$

2) error (substitution)

decoding: map the corrupted message $m \in \Sigma^n$ into closest codeword $\in \mathcal{C}$

correct decoding iff $e \triangleq$ # errors $\leq \frac{d-1}{2}$.

A code $\mathcal{C}$ has params:

(k, n, d) ← min. dist.

k: length orig. msg

n: len. of codew.

Rep. code: $(k, n = \alpha k, \alpha - 1)$

Thm (Shannon '48): can obtain code $\mathcal{C}$ with

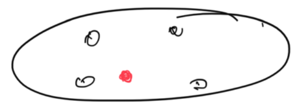
$n = \alpha \cdot k$ for $\alpha = O(1)$

$d = \beta \cdot n$ $\qquad \beta = \Omega(1)$.

$\Rightarrow$ can tolerate a const. frac. of failures

Pf: take $\mathcal{C}$ = set of 2^k random vectors in $\{0,1\}^n$ (for $\Sigma = \{0,1\}$).

E any map $\{0,1\}^k \xrightarrow[1-1]{} \mathcal{C}$



D: decode to closest codeword. $\boxtimes$

Issue: how to compute E, D.

namely: D ?

naively: time $\sim 2^k$.

want: time poly(k).

Reed-Solomon codes

for Σ "large".

$\Sigma = \mathbb{F}_p$ $\leftarrow$ integer arithmetic modulo p = prime.

Def: of the RS code:

$$\mathcal{C} = \{ (f(\alpha_1), f(\alpha_2), \dots f(\alpha_n)) : f(x) = \sum_{i=0}^{k-1} f_i x^i, \; f_0, \dots f_{k-1} \in \mathbb{F}_p \}$$

where $\alpha_1 \dots \alpha_n$ are distinct in $\mathbb{F}_p$.

an input message is encoded into a polynomial with coeff. = message

$$(f_0, f_1, \dots f_{k-1}) \xrightarrow{E} (f(\alpha_1), \dots f(\alpha_n)).$$

Params of the RS code?

think of $n = \Theta(k)$.

Lem: min dist in RS code is $d = n-k+1$.

Eg: $n = 2k \Rightarrow d = k+1$

$\Rightarrow$ can tol. k eras. $k/2$ errs.

(count frac of transm. msg.)

Pf: take 2 distinct codewords

$A(x) = \sum f_i x^i$ $\quad B(x) = \sum g_i x^i$

with $\{f_i\}_i \neq \{g_i\}_i$.

Dist $E(f)$ and $E(g)$:

\# $\underline{\alpha_i}$'s s.t. $A(\alpha_i) = B(\alpha_i)$.

A, B are 2 poly's of deg $\leq k-1$ distinct.

$\rightarrow$ at most k! (fundam. thm. of algebra)

$C(x) = A(x) - B(x) \neq 0$ poly.

$\Rightarrow C(x)$ can have $\leq \deg(C)+1$ zeros

$\Rightarrow$ at most k α_i's where $A(\alpha_i) = B(\alpha_i)$. $\boxtimes$

RS codes: $(k, n, \underset{d}{\underbrace{n-k+1}})$-code.

Algos for E, D?

E: simple: directly compute

<u>Decoding Algos</u>

<u>Erasure errors:</u> $(01011 \Downarrow 0??11)$

want dec. algo to decode as long as \# errors $\underline{e \leq n-k} = d-1$.

<u>Comp. Prob.</u>: we get a vector

$(y_1, y_2, \ldots y_n)$, where

for $i \in S$, $|S| \leq e$, $y_i = ?$

otherwise $y_i = f(\alpha_i)$.

Recover $\underline{f}(x) = \sum_{i=0}^{k-1} \underline{f_i} x^i$.

orig. msg $= (f_0 \ldots f_{k-1}) \Leftrightarrow f$.

time: poly(k).

$\rightarrow$ Input: $(\alpha_1 \ldots \alpha_n)$ (fixed)

$(y_1, \ldots, y_n)$ (corr msg)

Out: $f(x)$.

<u>Algo:</u> <u>polynomial interpolation</u>

given eval of $f(x)$ in $n-e \geq k$ values

Setup as a linear system of eq.:

$$\left[\sum_{i=0}^{k-1} f_i \alpha_j^i = y_j, \ j \notin S \right.$$

k unknowns (f_i)

$\geq k$ eqn's $(n - |S| \geq k)$.

matrix is Vandermonde matrix:

$$V = \begin{bmatrix} \alpha_1^0 & \alpha_1^1 & \alpha_1^2 & \cdots & \alpha_1^{k-1} \\ \alpha_2^0 & \alpha_2^1 & \alpha_2^2 & \cdots & \alpha_2^{k-1} \\ & \vdots & & & \\ \alpha_n^0 & \alpha_n^1 & & \cdots & \alpha_n^{k-1} \end{bmatrix}$$

with $\leq e$ rows missing.

$\det(V) \neq 0$ as long as k diff. α_j's.

<u>Subst. errors?</u> $\quad 00110 \downarrow 01010$

Challenge: we don't know S (pos. of errors)

trying all poss. for S too costly:

$$\binom{n}{|S|} \approx \binom{n}{n/2} \approx 2^{\Theta(n)}.$$

<u>Berlekamp-Welch algo</u> (decoding)

$$E(x) = \prod_{y_i \neq f(\alpha_i)} (x - \alpha_i) \quad \text{error poly.}$$

Don't know $E(x)$.

(*) $\quad \underline{E(\alpha_i)} \cdot (f(\alpha_i) - y_i) = 0 \quad \forall i \in [n]$.

$N(x) = E(x) \cdot f(x)$.

<u>Claim:</u> there exist poly's:

$N(\alpha_i) = y_i E(\alpha_i)$

where $N(x)$ has deg $e+k-1$

$E(x)$ — " — e

<u>Pf:</u> follows (*).

<u>Algo:</u> learn/compute:

poly $N(x)$ deg $e+k-1$

$E(x)$ deg e

s.t. $N(\alpha_i) = y_i E(\alpha_i)$.

<u>Pf:</u> setting up a system of lin. equations.

\# unknowns is $(e+k-1)+1$

$+ e + 1$

\# eqn's is n

can do $e+k+e+1 \leq n$

$\Rightarrow e \leq \frac{n-k-1}{2} = \frac{d}{2} - 1$.

RS: Best <u>possible</u> code, for large Σ.

trade-off b/w (k, n, d)