§ 3 Effiziente Decodierung von Reed-Solomon-Codes

Def.: Nearest Codeword Problem (NCP)

gegeben: Erzeugermatrix $G \in \mathbb{F}_q^{k \times n}$ eines $[n,k]$-Codes C, $y \in \mathbb{F}_q^n$

gesucht: Finde Codewort $x \in C$, das zu y am nächsten liegt, d.h. $d(x,y) = \min\{d(x',y) : x' \in C\}$.

NCP ist für allg. Codes NP-schwer

→ folgt aus dem Resultat von Berlekamp, McEliece, van Tilborg (1978)

→ hier: einfache Reduktion von MAXCUT.

Def.: Sei $\Gamma = (V,E)$ ein ungerichteter Graph.
Sei $G \in \mathbb{F}_2^{V \times E}$ die Inzidenzmatrix von Γ, d.h.

$$G_{v,e} = \begin{cases} 1, & \text{falls } v \in e \\ 0, & \text{sonst}. \end{cases}$$

Dies definiert einen linearen Code C_Γ:
G ist (nach dem Löschen von linear abhängigen Zeilen) die Erzeugermatrix des binären Codes C_Γ.

Lemma Sei $x \in C_\Gamma$. Dann gibt es $U \subseteq V$, so dass

$$x = \sum_{v \in U} x_v,$$ wobei $x_v \in \mathbb{F}_2^E$ die zu v gehörige Zeile von G ist. Dann ist

$$w(x) = |\delta(U)| = |\{e \in E : |e \cap U| = 1\}|$$

die Kardinalität des Schnittes, der durch $U \subseteq V$ definiert wird.

Bew.: Hinschauen. $\boxtimes$
(Wir rechnen modulo 2.)

Also: Wenn man das NCP für C_Γ für $y = 1 \dots 1$ lösen möchte, dann muss man einen Schnitt in Γ mit maximaler Kardinalität finden. (MAXCUT). Dies ist NP-schwer. ($\rightarrow$ VL Convex Optimization).

<u>Jetzt</u>: Effizienter Algo. zur Decodierung von Reed-Solomon-Codes $RS_{q,n,k}$

$k \le n \le q$, $\alpha_1, \dots, \alpha_n \in \mathbb{F}_q$

$RS_{q,n,k} = \{ (f(\alpha_1), \dots, f(\alpha_n)) : f \in \mathbb{F}_q[x], \deg f \le k-1 \}$

ist $[n, k, n-k+1]$-Code.

Welch-Berlekamp-Algorithmus (US-Patent 1983), nach Gemmel-Sudan: Löse NCP für $RS_{q,n,k}$, wenn y eindeutig decodiert werden kann, d.h. $\exists x \in RS_{q,n,k}$: $d(x,y) \le e$ und $e < \frac{d}{2} = \frac{n-k+1}{2}$.

<u>Algorithmus</u>

<u>Eingabe</u> $y \in \mathbb{F}_q^n$

<u>Ausgabe</u> $x \in RS_{q,n,k}$ mit $d(x,y) \le e$

oder "Fehler", falls $\forall x \in RS_{q,n,k}$: $d(x,y) > e$.

1. Bestimme Polynome $E \in \mathbb{F}_q[x] \setminus \{0\}$ mit $\deg E = e$ und $Q \in \mathbb{F}_q[x]$ mit $\deg Q \leq e+k-1$, so dass

$$y_i E(\alpha_i) = Q(\alpha_i) \qquad i \in \{1, \dots, n\}$$

(Löse linearen Gleichungssystem)

2. If $E \nmid Q$ then
   fail
   else
   $P = \frac{Q}{E}$

3. If $d(y, (P(\alpha_1), \dots, P(\alpha_n))) > e$ then
   fail
   else
   return $x = (P(\alpha_1), \dots, P(\alpha_n))$.

<u>Klar</u> a) Algorithmus läuft in Polynomialzeit

b) Falls x zurückgegeben wird, dann ist die Ausgabe korrekt.

Zugrunde liegende Ideen

1. Idee: "error locating polynomial"

Ang. $\exists\, x \in RS_{q,n,k}$ mit $d(x,y) \leq e$.

Sei $x = (P(\alpha_1), \dots, P(\alpha_n))$ für $P \in \mathbb{F}_q[x]$, $\deg P \leq k-1$.

Definiere das "error locating polynomial"

$$E = X^{e-r} \prod_{\substack{i=1 \\ P(\alpha_i) \neq y_i}}^{n} (X - \alpha_i) \quad \in \mathbb{F}_q[x], \quad \deg E = e,$$

mit $r = |\{ i : P(\alpha_i) \neq y_i \}|$.

Dann gilt

$$\boxed{y_i E(\alpha_i) = P(\alpha_i) E(\alpha_i), \quad i = 1, \dots, n} \qquad (*)$$

Weil: $y_i = P(\alpha_i) \Rightarrow E(\alpha_i) \neq 0$

$y_i \neq P(\alpha_i) \Rightarrow E(\alpha_i) = 0.$

2. Idee: Löse Gleichung $(*)$, um E und P zu finden. ($\rightsquigarrow$ ist leider nicht-linear)

3. Idee: Linearisiere

Setze $Q = PE$. Dann $\deg Q \le e + k - 1$.

und $P = \frac{Q}{E}$.

Mögliches Problem: Es kann passieren, dass das LGS im 1. Schritt keine eindeutige Lösung besitzt.

Ist aber kein Problem: Jede Lösung ist gut.

Lemma Ang. $\exists x \in RS_{q,n,k} : d(x,y) \le e$.

Ang. $\exists E, E' \in \mathbb{F}_q[x]$, $\deg E = \deg E' = e$, $E, E' \neq 0$

$\exists Q, Q' \in \mathbb{F}_q[x]$, $\deg Q, \deg Q' \le e + k - 1$,

$y_i E(\alpha_i) = Q(\alpha_i)$ und $y_i E'(\alpha_i) = Q'(\alpha_i)$, $i = 1, \dots, n$.

Dann $\frac{Q}{E} = \frac{Q'}{E'}$,

Bew.: Betrachte QE' und $Q'E$. Beide Polynome haben Grad $\le 2e + k - 1$.

Definiere

$$R = QE' - Q'E. \qquad \deg R \leq 2e+k-1.$$

Setze ein:

$$\begin{aligned} R(\alpha_i) &= Q(\alpha_i)E'(\alpha_i) - Q'(\alpha_i)E(\alpha_i) \\ &= y_i E(\alpha_i)E'(\alpha_i) - y_i E'(\alpha_i)E(\alpha_i) \\ &= 0. \end{aligned}$$

Also hat R wenigstens n verschiedene Nullstellen.
Andererseits ist $2e+k-1 < n$ $\left(e < \frac{n-k+1}{2} \right)$.
Also ist R das Nullpolynom. $\Rightarrow$ Beh. $\boxtimes$