קורס: קודים לתיקון שגיאות ושימושיהם במדעי המחשב
מרצה: ד״ר קלים יפרמנקו
סמסטר: סתיו תשפ״א
תאריך: 05/01/2020

הרצאה 21

List Decoding Folded RS

קודי Folded RS, מתקבלים ע״י מיפוי r קודי Reed-Solomon על אלפ-בית גדול יותר ע״י איגוד מדוקדוק של סמלי קוד.

קודים אלה, הינם מקרה מיוחד של קודי Parvaresh-Vardy.

בעזרת שימוש בפרמטרים אופטימליים עבור Folded RS, ניתן לפענח בקצב R ולהשיג רדיוס פענוח של 1-R.

המונח "Folded Reed-Solomon Codes" נטבע במאמר של V.Y. Krachkovsky עם אלגוריתם שהציג קודי RS עם שגיאות אקראיות רבות. אלגוריתם list decoding עבור קודי Folded RS מתקן מעבר לחסם עבור קודי RS, שהושג ע״י אלגוריתם Guruswami-Sudan עבור שגיאות אקראיות כאלה.

הגדרה:

עבור $\Sigma_1 = F_q$,

$\Sigma_2 = \left(F_q\right)^r$,

$x_1, x_2, \dots, x_n \in F_q$ שונים,

עבור $\lambda \in F_q$ מסדר גדול מ r(*),

ופולינום p מדרגה $k-1$,

קוד Folded RS $C: (\Sigma_1)^k \to (\Sigma_2)^n$ מוגדר באופן הבא:

$$C(p) = \begin{pmatrix} p(\lambda x_1) & p(\lambda x_2) & \cdots & p(\lambda x_n) \\ p(\lambda^2 x_1) & (\lambda^2 x_2) & \cdots & p(\lambda^2 x_n) \\ \vdots & \vdots & \ddots & \vdots \\ p(\lambda^r x_1) & p(\lambda^r x_2) & \cdots & p(\lambda^r x_n) \end{pmatrix}.$$

כלומר, $C(p) \in \left(F_q\right)^{r \times n}$ הינה מטריצה כאשר $C(p)_{i,j} = p(\lambda^i x_j)$.

(*)$\lambda^i \neq 1$ לכל $i \in \{1, \dots, r\}$.

הערות:

1. הסימבולים של הקוד הם עמודות המטריצה. כלומר, שני סימבולים זהים אמ״ם כל r הכניסות שלהם זהות.
2. לכל $j \in \{1, \dots, n\}$ ולכל $i \neq i' \in \{1, \dots, r\}$ מתקיים כי $\lambda^i x_j \neq \lambda^{i'} x_j$.

תחילה נראה אלגוריתם שמפענח $n(1 - \frac{1}{r}n - rR)$ שגיאות, ולאחר מכן אלגוריתם שמפענח $n(1 - \epsilon n - R)$ שגיאות ובעצם מגיע ל "List Decoding Capacity".

עבור $D=\frac{n}{r+1}-\frac{k}{r+1}$, אלגוריתם ה List Decoding הבא מצליח להוציא רשימה בגודל q^r שמכילה את ההודעה הנכונה, אם יש $D+k$ מקומות נכונים:

נסמן $\beta_{i,j}=p(\lambda^i x_j)$.

1. מצא $r+1$ פולינומים $A_0(x),A_1(x),\dots,A_r(x)\in F_q[x]$ מדרגה מינימאלית המקיימים:
   א. $\deg(A_0)\le D+k$
   ב. $\deg(A_i)\le D$ לכל $i\in\{1,\dots,r\}$
   ג. $A_0(x_j)+\sum_{i=1}^r A_i(x_j)\cdot\beta_{i,j}=0$ לכל $j\in\{1,\dots,n\}$
   ד. קיים $i\in\{0,\dots,r\}$ כך ש $A_i(x)\not\equiv 0$
2. מצא את כל הפולינומים $p(x)$ כך ש $\Lambda_p(x)=A_0(x)+\sum_{i=1}^r p(\lambda^i x)\cdot A_i(x)\equiv 0$

<u>ניתוח האלגוריתם:</u>

1. מספר הנעלמים בשלב 1 הינו $r(D+1)+D+k+1$
(עבור A_0 קיימים $D+k+1$ מקדמים, ועבור A_i לכל $i\in\{1,\dots,r\}$ קיימים $D+1$ מקדמים).
מספר המשוואות הינו n.
כלומר, ע"פ עקרונות של מערכת משוואות מאלגברה ליניארית, אם $r(D+1)+D+k+1>n$ קיים פתרון. מכאן, נציב את D ונקבל כי קיים פתרון אם $r>-1$. כלומר, תמיד קיים פתרון.

2. - <u>פתרון יעיל:</u> נבחין כי על מנת למצוא פתרון $\Lambda_p(x)\equiv 0$, נדרש כי כל המקדמים של $\Lambda_p(x)$ הינם 0.
נסמן $A_i=\sum a_{i_j}x^j$, $p(x)=\sum c_j x^j$.
המקדם החופשי של $\Lambda_p(x)$ הינו $a_{0_0}+c_0\sum_{i=1}^r a_{i_0}$.
כלומר, קיבלנו כי $a_{0_0}+c_0\sum_{i=1}^r a_{i_0}=0$, וניתן למצוא את c_0 אם $\sum_{i=1}^r a_{i_0}\ne 0$.
נניח כי $\sum_{i=1}^r a_{i_0}\ne 0$ ומצאנו את c_0.
המקדם של x בפולינום $\Lambda_p(x)$ הינו $a_{0_1}+c_0\sum_{i=1}^r a_{i_1}+c_1\sum_{i=1}^r\lambda^i a_{i_0}$.
כלומר, קיבלנו כי $a_{0_1}+c_0\sum_{i=1}^r a_{i_1}+c_1\sum_{i=1}^r\lambda^i a_{i_0}=0$, וניתן למצוא את c_1 אם $\sum_{i=1}^r\lambda^i a_{i_0}\ne 0$.
נניח כי $\sum_{i=1}^r\lambda^i a_{i_0}\ne 0$ ומצאנו את c_1.
באופן הנ"ל נמצא את $c_0,c_1,\dots$ וכך בעצם נמצא את הפולינום p. במידה ובאחד השלבים המקדם של c_k הינו 0, נקח את כל q האפשרויות עבור מקדם זה.

- <u>מספר הפולינומים p לכל היותר q^r:</u> לא ניתן למצוא c_k כלשהו אם $B(\lambda^i)=\sum_{i=1}^r(\lambda^i)^k a_{i_0}=0$. לפולינום B לכל היותר r שורשים, כלומר לא ניתן למצוא c_k לכל היותר r פעמים, וקיבלנו כי מספר הפולינומים שבנינו הינו לכל היותר q^r.

- <u>הפולינום הנכון ברשימה:</u> נתבונן בפולינום $\Lambda_p(x)$.

נראה כי ל $\Lambda_p(x)$ קיימים יותר שורשים מהדרגה שלו, ומכך נסיק כי $\Lambda_p(x) \equiv 0$.

ראשית נבחין כי $\deg\left(\Lambda_p(x)\right) = \max\left\{\deg(A_0(x)), \deg\left(p\left(\lambda^i x\right) \cdot A_i(x)\right) : i \in \{1, \dots, r\}\right\}$.

מכיוון ש $\deg(A_0(x)) \leq D + k$, $\deg\left(p\left(\lambda^i x\right)\right) \leq k$ וגם $\deg\left(A_i(x)\right) \leq D$ נקבל כי $\deg\left(\Lambda_p(x)\right) \leq D + k$.

אם העמודה ה j לא משובשת אז $\beta_{i,j} = p(\lambda^i x_j)$, ועפ"י הבנייה של A_i נקבל כי $A_0\left(x_j\right) + \sum_{i=1}^{r} \beta_{i,j} \cdot A_i\left(x_j\right) \equiv 0$. כלומר קיבלנו $A_0\left(x_j\right) + \sum_{i=1}^{r} p\left(\lambda^i x\right) \cdot A_i\left(x_j\right) \equiv 0$.

במילים אחרות, קיבלנו כי $\Lambda_p\left(x_j\right) = 0$ לכל $j \in \{1, \dots, n\}$.

כלומר, אם העמודה ה j אינה משובשת, מתקיים כי x_j הינו שורש של Λ_p:

מספר השגיאות הינו לכל היותר $n - (D + k + 1) \Leftarrow$

יש לפחות $D + k + 1$ עמודות j שאינן משובשות $\Leftarrow$

לפולינום Λ_p יש $D + k + 1$ שורשים $\Leftarrow$

ל $\Lambda_p(x)$ קיימים יותר שורשים מהדרגה שלו $\Leftarrow$

$\Lambda_p(x) \equiv 0$.

<u>רעיון שיפור האלגוריתם:</u> עבור $r = 3$ לדוגמא, ניתן לבנות מהסימבול $\begin{pmatrix} p\left(\lambda x_j\right) \\ p\left(\lambda^2 x_j\right) \\ p\left(\lambda^3 x_j\right) \end{pmatrix}$ שני סימבולים $\begin{pmatrix} p\left(\lambda^2 x_j\right) \\ p\left(\lambda^3 x_j\right) \end{pmatrix}, \begin{pmatrix} p\left(\lambda x_j\right) \\ p\left(\lambda^2 x_j\right) \end{pmatrix}$. בעזרת הרעיון הזה עבור r מסדר גדול יותר ניתן לבנות קודים שמגיעים ל $(1 - R - \epsilon)$ שגיאות, ובעצם להשיג את ה "List Decoding Capacity".