סיכום הרצאה 14:

פיענוח RS-read Solomon

Reed–Solomon code: הוא קוד תיקון שגיאות ליניארי נפוץ ושימושי ביותר, המבוסס על אינטרפולציה באמצעות פולינומים מעל שדות סופיים .הקוד מיועד להעברה אמינה של מידע בערוץ תקשורת רועש שעלולות להיווצר בו שגיאות.

עד כה בקורס דיברנו על הבניה של הקודים לתיקון שגיאות, ומה שעניין אותנו בעיקר היה מרחק הקוד.

כתוצאה מכך, בנינו קודי RS אשר הראינו כי הם אופטימליים מעל א'-ב' גדול.

כעת נתחיל לענות על השאלות החשובות הבאות :

כמה זמן לוקח לקודד הודעה בעזרת קוד RS ?

כמה זמן לוקח לפענח הודעה המקודדת בקוד RS?

גוסטב סולומון (מימין) ואירווינג ריד

סיבוכיות זמן של קידוד הודעה:

ניזכר כי קידוד RS מתבצע באופן הבא :

RS מקבל כקלט את המקדמים $\{m_0, m_1, \dots, m_{k-1}\}$ כאשר m_i הינן ההודעות להצפנה,

בונה מהם פולינום $\sum_{i=0}^{i=k-1} m_i x^i = m(x)$

ומחשב את ערכי $m(x)$ בנקודות $\alpha_1, .., \alpha_n$ אשר הן פרמטר של האלגוריתם.

כעת נחשב את סיבוכיות הזמן של האלגוריתם :

בחישוב נאיבי ניתן להציב בערכי x אשר בהגדרת הפולינום את ערכי α ונחשב את חזקתם בזמן $O(k)$

מכאן שחישוב n הנקודות יתבצע בזמן $O(kn)$

ניתן לחשב בצורה יעילה יותר את ערכי הנקודות בפולינום בעזרת אלגוריתם FFT - Fast Fourier transform

בזמן $O(n * log^2(k))$

הרחבה – אלגוריתם FFT:

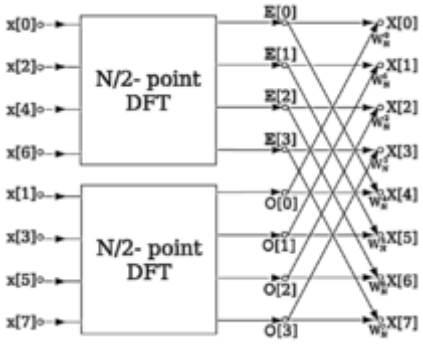
מכפלת הפולינומים מתבצעת על ידי חישוב ערכי כל פולינום בנקודות שונות, חישוב ערכי פולינום המכפלה ואינטרפולציה לקבלת הפולינום מערכי נקודותיו, בעזרת FFT נייעל את חישוב ערכי נקודות הפולינומים, ואת ביצוע האינטרפולציה :

קלט האלגוריתם הוא וקטור מקדמים $(\alpha_0, \dots, \alpha_{n-1})$ של פולינום $p(x) = \sum_{i=0}^{i=n-1} \alpha_i x^i$

פלט האלגוריתם הוא הווקטור $(p(\omega_n^0), (\omega_n^1), \dots, (\omega_n^{n-1}))$ כלומר וקטור הערכים בשורשי היחידה, המכונה DFT של וקטור המקדמים.

לאחר חישוב ערכי הנקודות ומכפלתן, ניתן לבצע אינטרפולציה בעזרת מכפלת הערכים ב-DFT ההופכית.

האלגוריתם עובד בשיטת ״הפרד ומשול״ הוא מחלק את הבעיה ל־2, פותר כל תת בעיה באופן רקורסיבי, ומחבר את שני תתי הפתרונות לכדי פתרון לבעיה הכללית. פירוק והרכבת הבעיה יוצא יעיל יותר מאשר פתרון ללא פירוק, הסיבה שניתן לפרק ולהרכיב בצורה יעילה נעוצה בכך שבוחרים את שורשי היחידה כנקודות שבהן נייצג את הפולינום לפי ערכים.



<u>סיבוכיות זמן של פיענוח הודעה:</u>

<u>פיענוח מחיקות:</u>

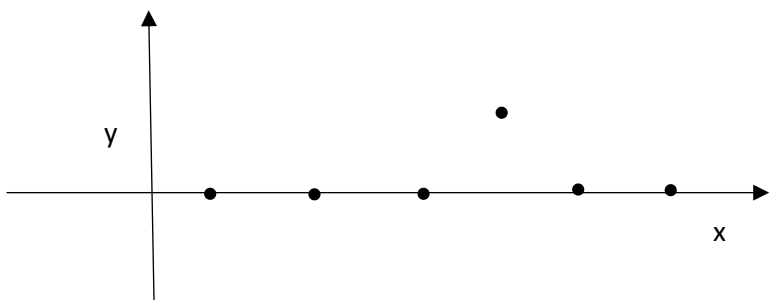
במקרה זה מקבלים כפלט מאלגוריתם RS את הערכים $p(\alpha_1),?,?,p(\alpha_4)\dots,p(\alpha_n)$ כלומר, אנו יודעים את הערך בהרבה מקומות, אבל בתאים בהם התבצעה מחיקה נקבל סימון מיוחד – "?"

על מנת למצוא את הפולינום נבנה מטריצה בה לכל i כך ש $p(\alpha_i)$ ידוע, השורה ה-i תהיה ההעתקה הלינארית עבור α_i

$$\begin{pmatrix} p(\alpha_1) \\ p(\alpha_3) \\ \vdots \\ p(\alpha_n) \end{pmatrix} = \begin{pmatrix} m_0\alpha_1^0 & \cdots & m_{k-1}\alpha_1^{k-1} \\ m_0\alpha_3^0 & \dots & m_{k-1}\alpha_3^{k-1} \\ \vdots & \ddots & \vdots \\ m_0\alpha_n^0 & \cdots & m_{k-1}\alpha_n^{k-1} \end{pmatrix}$$

נציב את ערכי α_i בכל שורות המטריצה ונקבל (מספר המחיקות $-n$) משוואות לינאריות. מרחק הקוד הוא $n-k+1$ ולכן עבור מספר מחיקות $\geq n-k$ יש פתרון יחיד למערכת המשוואות. ומכאן נסיק כי ניתן לפענח את הקוד באופן חד משמעי. נחשב את מערכת המשוואות בעזרת Gauss elimination בזמן של $O(k^3)$.

כיוון שאנו יודעים את ערכי פולינום מדרגה k ב- k מקומות נוכל לשחזר את הפולינום בעזרת אינטרפלציית לגראנג', אשר אותה נתאר כעת.



נבנה פולינום באופן הבא :

$$p_j = \prod_{i=1,i\neq j}^{i=k} \frac{x-\alpha_i}{\alpha_j-\alpha_i} \Leftarrow$$

נבחין כי מתקיים :

$$i \neq j \rightarrow P_j(\alpha_i) = 0$$
$$P_j(\alpha_j) = 1$$

נסמן את ערכי α_j בפולינום להיות y_j, נגדיר $p(x) = \sum_{j=1}^{j=k} y_j p_j(x)$

$p(\alpha_j) = y_j$ – מתקיים כיוון שלפי הגדרה $p(\alpha_j) = \sum_{i=1}^{i=k} y_i p_i(\alpha_j)$ ושווה אפס מלבד $i=j$ כנדרש.

חישוב כל פולינומים p_j יעלה $O(n^2 log^2(n))$ כיוון שיש $\log(n)$ מכפלות.

חישוב $p(x)$ מתבצע בזמן לינארי, ולכן סך הכל ניתן לחשב בסיבוכיות זמן של $O(n^2 log^2(n))$

כפי שהראינו, ניתן לחשב זאת בעזרת FFT בזמן יעיל יותר של $O(nlog^2(n))$.

בנוסף, ניתן לחשב את הפולינומים p_j מראש, ואז זמן הריצה בפועל יהיה לינארי ב – n , כלומר $O(n)$

פיענוח שגיאות:

כאמור אלגוריתם RS מקבל פולינום מדרגה $k-1$ ומחזיר n ערכים : $RS: p(x) -> \ p(\alpha_1)..,p(\alpha_n)$

אם האורך הוא n והמימד k אז מרחק הקוד הוא $d = N-k+1$ ולכן אפשר לפענח $\tau = \left\lfloor \frac{d-1}{2} \right\rfloor$ שגיאות.

אנו מקבלים ערכים $y_1, y_2, \dots, y_m$ כקלט, וידוע שלכל y_i מלבד τ כאלו, מתקיים $y_i = p(\alpha_i)$

כאשר ערכי α_i ידועים, אבל הפולינום $p(x)$ ו- τ המקומות בהם $y_i \neq p(\alpha_i)$ אינם ידועים.

נשתמש באלגוריתם Berlekamp – welch (הצגה של אלגוריתם זה ע"י (Gemmel-Sadan

נסמן $E' = \{i: y_i \neq p(\alpha_i)\}$ כלומר המקומות של השגיאות. את הקבוצה E' אנחנו כמובן לא יודעים

נגדיר את פולינומים באופן הבא :

$E(x) = \prod_{i \in E'} x - \alpha_i$ מתקיים $\deg E(x) \leq \tau$

$N(x) = E(X).p(X)$ מתקיים $\deg N(x) = \tau + k - 1$

<u>טענה</u>: לכל $1 \leq i \leq n$ מתקיים $E(\alpha_i) * y_i = E(\alpha_i) * p(\alpha_i)$

<u>הוכחה</u>: אם מתקיים ש $y_i = p(\alpha_i)$ סיימנו, אם $i \in E$ מההגדרה נובע ש : $E(\alpha_i) = 0$

מכאן מתקיים : $E(x)y_i = p(\alpha_i)E(x)$ כנדרש.

קיבלנו שיש שני פולינומים $E(x), N(x)$ המקיימים :

- $DegE(X) \leq \tau$
- $DegN(X) \leq k - 1 + \tau$
- $E(\alpha_i)y_i = N(\alpha_i)$ לכל $1 \leq i \leq n$

אם נסמן :

$$N(X) = \sum_{i=0}^{i=\tau+k-1} n_i x^i$$

$$E(X) = \sum_{i=0}^{i=\tau} e_i x^i$$

$E(\alpha_i)y_i = N(\alpha_i)$ לכל $1 \leq i \leq n$ נותן לנו n משוואות על המקדמים של N ו E

יש $2\tau + k + 1$ נעלמים, $\tau = \left\lfloor \frac{d-1}{2} \right\rfloor$ אז סך הנעלמים שווה $d + k$ השווה $n + 1$

לכן תמיד ניתן למצוא פתרון למערכת משוואות, הבעיה שיכול להיות שיהיו לנו הרבה פתרונות, לכן נמצא

פתרון כלשהו. נוכיח כי עבור פתרון כלשהו $E_1(x), N_1(x)$ מתקיים ש: $p(x) = \frac{N_1(x)}{E_1(x)}$

<u>תיאור האלגוריתם:</u>

מקבל כקלט $\{\alpha_1, \dots, \alpha_n, y_1, \dots, y_n\}$ כך שלכל y_i מלבד τ כאלו, מתקיים $y_i = p(\alpha_i)$

מוצא פולינום השונה מפולינום ה-0 שמקיים:

- $Q(x,y) = E_1(x)y - N_1(x)$
- $deg\ E_1 \le \tau, Deg\ N_1 \le \tau + k - 1$
- $Q(\alpha_i, y_i) = 0$

מחזיר: $\frac{N_1(x)}{E_1(x)}$

$Q(\alpha_i, y_i) = 0$ זה משוואה לינארית על מקדמי Q, וניתן למצוא את הפולינום השונה מ-0 ב- $O(nlog^2(n))$ בעזרת אינטרפולציה של פונקציה רציונלית.

כעת נוכיח כי $p(x) = \frac{N_1(x)}{E_1(x)}$:

נתבונן בפולינום הבא:

$$R(x) = E_1(x)p(x) - N_1(x)$$

מדרגות הפולינומים E_1, P, N_1 נקבל $Deg\ R(x) \le \tau + k - 1$

<u>למה</u>: $R(x) = 0$

<u>הוכחה</u>: נוכיח זאת ע"י כך שנראה כי יש לפולינום יותר שורשים מאשר דרגתו.

$$\forall i \notin E' \quad R(\alpha_i) = E_1(\alpha_i)p(\alpha_i) - N_1(\alpha_i) = E_1(\alpha_i)y_i - N_1(\alpha_i) = 0$$

קיבלנו כי $\forall i \notin E'$ מתקיים $R(\alpha_i) = 0$.

לכן ל R יש $n - \tau$ שורשים ומכאן שאם $n - \tau > \tau + k - 1$ נוכיח את הטענה.

$$n - \tau > \tau + k - 1 \iff n > 2\tau + k - 1 \iff n > d - 1 + k - 1 = n - 1$$

ולכן $E_1(x)p(x) - N_1(x) = 0$.

מהלמה קיבלנו כי $E_1(x)p(x) - N_1(x) = 0$, נסדר אגפים ונקבל $p(x) = \frac{N_1(x)}{E_1(x)}$ כנדרש

<u>הערה</u>: צריך להוכיח $E_1(x) \neq 0$

נניח בשלילה, נקבל ש $N_1(\alpha_i) = 0 \quad 1 \le i \le n$

לכן בגלל ש $deg\ N_1 < N$ נקבל שגם $N_1 = 0$ וזו סתירה כי אז $Q(x,y) = 0$ בניגוד להגדרתו.