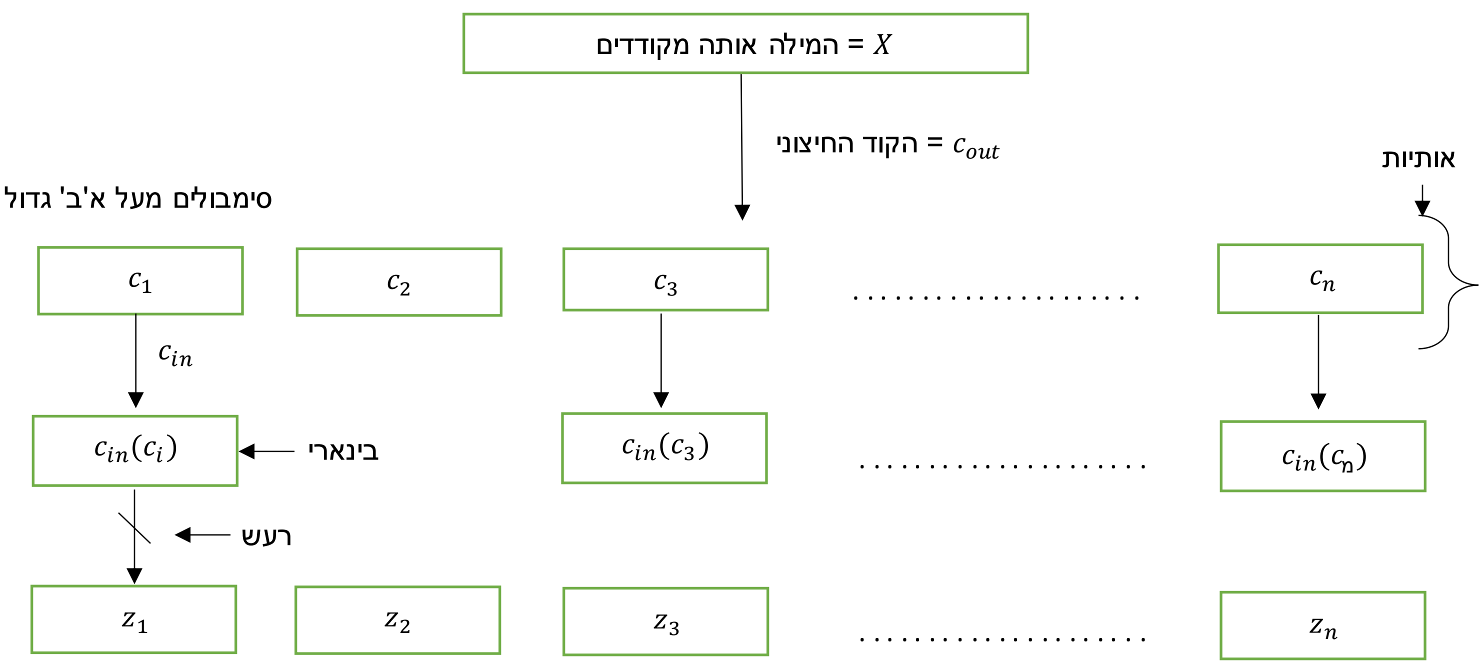
סיכום הרצאה 15:

concatenated codes decoding

שיעור שעבר ראינו אלגוריתם המפענח קוד RS עם $\left\lfloor\frac{d-1}{2}\right\rfloor$ שגיאות הרץ בסיבוכיות זמן של $O(n\log^2 n)$.
הבעיה היא שאלגוריתם RS אינו לקודים בינאריים אלא קודים מעל א״ב גדול, הדרך להתמודד עם א״ב גדול היא באמצעות שרשור של קודים.

היום נרצה להראות אלגוריתם לפיענוח של קודים משורשרים:

קוד משורשר עובד בצורה הבאה:



אם מרחק הקוד של $c_{out} = D$ ומרחק הקוד של $c_{in} = d$ אז מרחק הקוד המשורשר: $\mathbf{c_{out} \circ c_{in} = Dd}$.

ולכן ניתן לתקן $\left\lfloor\frac{dD-1}{2}\right\rfloor$ שגיאות.

השיעור נענה על השאלה איך עושים זאת באופן יעיל.

בהנחה שאנו יודעים איך עושים את קוד c_{out} וקוד c_{in} נתאר אלגוריתם נאיבי ואלגוריתם יעיל לפתרון:

<u>אלגוריתם נאיבי :</u>

מקבל כקלט את $z_1, \dots, z_{n_{out}}$

<u>שלב ראשון :</u> לכל $1 \le i \le n_{out}$ מצא את a_i כך ש $\Delta(c_{in}(a_i), z_i)$ מינימלי, כלומר כך שהמרחק בין הפעלת c_{in} על a_i ל- z_i הינו מינימאלי.

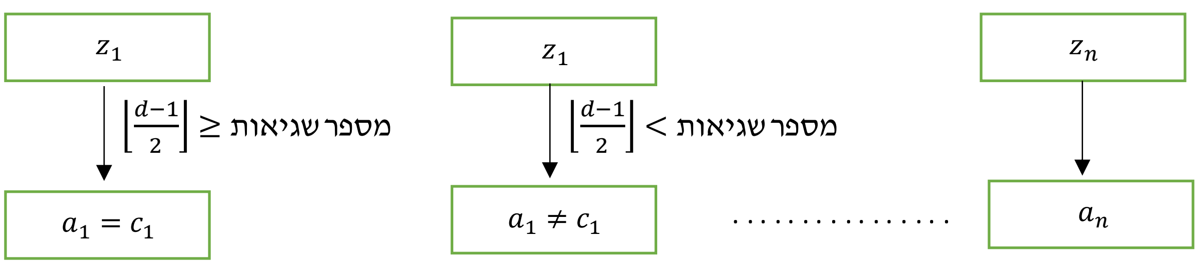
האלגוריתם מקבל את האותיות המשובשות ומנסה עבור כל אות לזהות את הסימבול עבורה, ובכך לזהות את הסימבולים : $c_1, \dots, c_n$

<u>שלב שני :</u> נפעיל אלגוריתם פיענוח של c_{out} על ערכי $a_1, \dots, a_{n_{out}}$ שקיבלנו בשלב הקודם, ונחזיר את מה שקיבלנו.

האלגוריתם הנאיבי לוקח כל z_i (אות משובשת) ומוצא את מילות הקוד המתאימות : $a_1, \dots, a_n$ ומפעיל עליהן את הפיענוח של c_{out} לקבלת ההודעה המקורית.

אם עבור z_i מספר השגיאות $\ge \left\lfloor \frac{d-1}{2} \right\rfloor$ אז אלגוריתם הפיענוח של c_{in} יחזיר את הדבר הנכון : $a_i = c_i$

ואם עבור z_i מספר השגיאות $< \left\lfloor \frac{d-1}{2} \right\rfloor$ אז אלגוריתם הפיענוח של c_{in} יחזיר : $a_i \neq c_i$, לדוגמה :



אם מספר ה - a_i הלא נכונים $> \left\lfloor \frac{D-1}{2} \right\rfloor$ האלגוריתם יחזיר תשובה נכונה.

<u>משפט</u> : אם מספר השגיאות $> \frac{dD}{4}$ אז האלגוריתם הנאיבי יחזיר תשובה נכונה

<u>הוכחה</u> : נניח כי מספר השגיאות $> \frac{dD}{4}$.

נתבונן בשלב הראשון ונסתכל על הקבוצה $S = \{i : a_i \neq c_i\}$

לכל $i \in S$ מכיוון שהאלגוריתם מוצא a_i כך שהמרחק בין הפעלת c_{in} על a_i ל- z_i הינו מינימאלי אז מתקיים $\Delta(c_{in}(a_i), z_i) < \Delta(c_{in}(c_i), z_i)$

אנו יודעים כי לא פענחנו נכון את a_i, מכאן שמספר השגיאות הינו לפחות d. כלומר:

$d \le \Delta(c_{in}(a_i), c_{in}(c_i))$.

כמו כן מאי שוויון המשולש: $\Delta(c_{in}(a_i), c_{in}(c_i)) \le \Delta(c_{in}(a_i), z_i) + \Delta(c_{in}(c_i), z_i)$.

מכאן קיבלנו כי $d \le (c_{in}(a_i), z_i) + \Delta(c_{in}(c_i), z_i)$

המרחק בין הפעלת c_{in} על a_i ל- z_i הינו מינימאלי אז מתקיים
$\Delta(c_{in}(a_i), z_i) < \Delta(c_{in}(c_i), z_i)$

ולכן קיבלנו כי $d \le 2\Delta(c_{in}(c_I), z_i) \Leftarrow \frac{d}{2} \le \Delta(c_{in}(c_I), z_i)$

כיוון שהשגיאה לכל $i \in S$ היא לפחות $\frac{d}{2}$, מספר השגיאות בקוד הוא לפחות $\frac{d}{2}|S|$.

בגלל שמספר השגיאות $< \frac{dD}{4}$ נקבל: $|S| < \frac{D}{2}$, ו- $|S|$ מספר שלם לכן: $|S| < \left\lfloor \frac{D-1}{2} \right\rfloor$.

כיוון שיש ב- $a_1, \dots, a_n$ יש לכל היותר $\left\lfloor \frac{D-1}{2} \right\rfloor$ שגיאות, אלגוריתם הפיענוח של השלב השני יחזיר תשובה נכונה,

ומכאן שהאלגוריתם החזיר תשובה נכונה.

ניתוח זמן ריצה:

ראינו כעת אלגוריתם שמחזיר תשובה נכונה, על מנת שיהיה יעיל אנו רוצים שזמן הריצה יהיה פולינומיאלי.

c_{in} מקודד מספר קטן של ביטים $O(\log n)$ וכיוון שזמן הריצה של c_{in} הוא כמספר ההודעות שיכולות להיות,
ניתן לחסום ב $O(n)$.
כלומר מספר ה- a_i האפשריים פרופורציונאלי ב-n.
עבור על כל a_i ניתן לחשב את $c_{in}(a_i)$ ואת המרחק של זה ב $\log(n)$

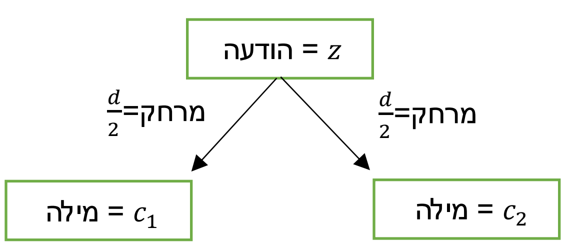
ולכן סך הכל השלב הראשון, בו עוברים כל ה- a_i מתבצע בסיבוכיות זמן של $O(n\log(n))$.

האלגוריתם הנאיבי מפענח מספר $\frac{dD}{4}$ שגיאות, נטען כי **ניתן גם לפענח $\frac{dD}{2}$ שגיאות**.

על מנת ש- $c_{in}(c_i)$ יטעה, מספיקות $\frac{d}{2}$ שגיאות.

נניח שיש שתי מילים במרחק $\frac{d}{2}$,

במקרה כזה שגיאה קטנה תגרום לפיענוח לא נכון.



באלגוריתם החדש, נתייחס להודעה כזו (z), כאל הודעה שאנחנו לא יודעים לפענח.
לכל אחד מהסימבולים שנפענח ניתן משקל, המתאר עד כמה אנו בטוחים שזה הסימבול הנכון.

קורס : קודים לתיקון שגיאות ושימושיהם במדעי המחשב
מרצה : ד״ר קלים יפרמנקו
סמסטר סתיו תשפ״א
06/12/2020

<u>תיאור כללי :</u>

נקבל כקלט $z_1, \dots, z_n$ ונפענח אותם, לאחר מכן באופן הסתברותי ובהתאם לכמה שגיאות היו בפענוח (המרחק מהמילת קוד שפענחנו אליה), נבצע מחיקות - חלק מהסימבולים יימחקו.

כפי שהראינו בתרגיל הבית, אם יש e שגיאות, s מחיקות ו- $2e + s < D$ אז אפשר לפענח מ- s מחיקות ו- e שגיאות, לכן **כל שגיאה שקולה לשתי מחיקות.**

<u>תיאור האלגוריתם של Forney :</u>

אלגוריתם אקראי העובד בשיטת Generalized Minimum distance (אלגוריתם יעיל לפענוח קודים משורשרים, המבוסס על שימוש במפענח שגיאות ומחיקה עבור הקוד החיצוני.)

מקבל כקלט $z_1, \dots, z_n$

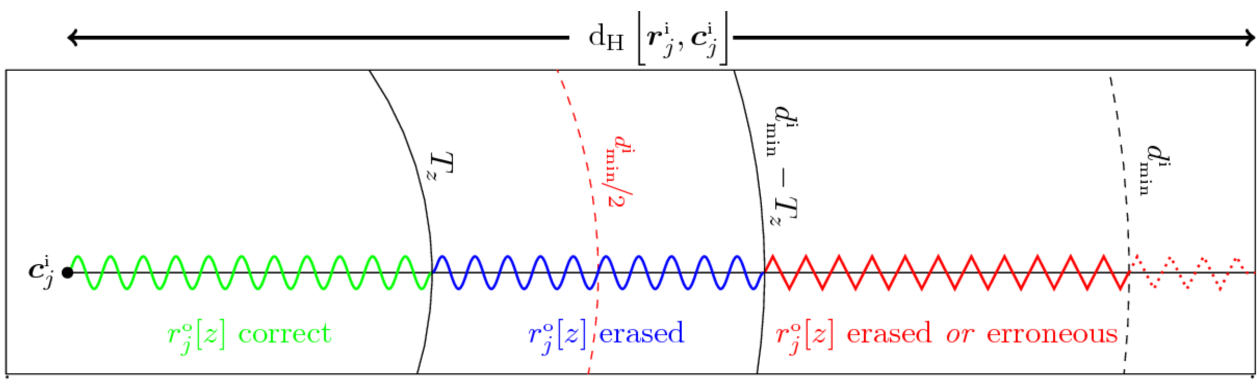
שלב ראשון : לכל i מצא a_i כך ש $\Delta(c_{in}(a_i), z_i)$ מינימאלי

שלב שני : נגדיר $w_i = \Delta(c_{in}(a_i), z_i)$

שלב שלישי : בסיכוי $\frac{2w_i}{d}$ נחליף סימבול a_i בסימן ״?״ (כלומר במחיקה)

שלב רביעי : נריץ אלגוריתם פיענוח על מילת $a_1, \dots, a_n$ כשאר קיימים $1 \le i \le n$ כך ש? $a_i =$ - נמחקו.

האלגוריתם מפענח כל z_i ובהסתברות שגדלה ככל שהיו שגיאות רבות יותר בפיענוח, מחליף את הפיענוח במחיקה. לאחר מכן מריץ אלגוריתם פיענוח על ה-a_i



<u>משפט:</u> אם מספר השגיאות קטן מ $\frac{dD}{2}$ ונסמן במשתנה אקראי e את מספר השגיאות ב- a_i ובמשתנה אקראי- s את מספר המחיקות ב- a_i אז מתקיים $E(2\varepsilon + s) < D$

<u>הוכחה:</u> נניח כי מספר השגיאות קטן מ$\frac{dD}{2}$.

לכל i נסמן $\varepsilon_i = \begin{cases} 0, & a_i = c_i \\ 1, & a_i \neq c_i \end{cases}$ כלומר משתנה אינדיקטור לשגיאה

ונסמן לכל i נסמן $s_i = \begin{cases} 0, & a_i = '?' \\ 1, & a_i \neq '?' \end{cases}$ כלומר משתנה אינדיקטור למחיקה

לפי הגדרה $\varepsilon = \sum_{i=1}^{i=n} \varepsilon_i$, $s = \sum_{i=1}^{i=n} s_i$ ולכן מתקיים:

$E(\varepsilon) = E(\sum_{i=1}^{i=n} \varepsilon_i)$, $E(s) = E(\sum_{i=1}^{i=n} s_i)$ ומליניאריות התוחלת נקבל

$E(\varepsilon) = \sum_{i=1}^{i=n} E(\varepsilon_i)$, $E(s) = \sum_{i=1}^{i=n} E(s_i)$

<u>למה:</u> אם בz_i היו ε_i שגיאות, אז $E(2\varepsilon_i + s_i) \leq \frac{2e_i}{d}$

<u>הוכחת הלמה:</u>

כפי שהגדרנו $w_i = \Delta(c_{in}(a_i), z_i)$

נחלק למקרים:

<u>מקרה ראשון</u>- $a_i = c_i$: כי אין שגיאה אז מתקיים $\varepsilon_i = 0$

מההגדרה $E(s_i) = \frac{2w_i}{d}$

מכיוון ש$a_i = c_i$ מתקיים $\Delta(c_{in}(c_i), z_i) = \Delta(c_{in}(a_i), z_i)$ ו$\Delta(c_{in}(c_i), z_i) = e_i$ מהגדרה

ולכן מתקיים $e_i = w_i \Leftarrow \frac{2w_i}{d} = \frac{2e_i}{d}$

לכן $E(s_i) = \frac{2e_i}{d}$

סה"כ קיבלנו כי $E(2\varepsilon_i + s_i) = E(s_i) = \frac{2e_i}{d}$ כנדרש.

<u>מקרה שני</u> - $a_i \neq c_i$(יש שגיאה או מחיקה):

כמו במקרה הקודם $E(s_i) = \frac{2w_i}{d}$ ותוחלת השגיאה היא המשלים כי אנו במקרה בו יש שגיאה או מחיקה ולכן: $E(\varepsilon_i) = 1 - \frac{2w_i}{d}$.

מכאן: $E(2\varepsilon_i + s_i) = 2 - \left(\frac{2w_i}{d}\right)2 + \frac{2w_i}{d} = 2 - \frac{2w_i}{d}$

<u>טענת עזר:</u> $d \le e_i + w_i$

<u>הוכחת טענת עזר</u>:

במקרה של שגיאה המרחק הינו לפחות d לכן מתקיים: $d \le d(c_{in}(a_i), c_{in}(c_i))$.

מאי שוויון המשולש נקבל: $\Delta(c_{in}(a_i), c_{in}(c_i)) \le \Delta(c_{in}(a_i), z_i) + \Delta(c_{in}(c_i), z_i)$

לפי הגדרה $\Delta(c_{in}(c_i), z_i) = e_i$ ו- $w_i = \Delta(c_{in}(a_i), z_i)$

ולכן $d \le d(c_{in}(a_i), c_{in}(c_i)) \le \Delta(c_{in}(a_i), z_i) + \Delta(c_{in}(c_i), z_i) = w_i + e_i$ $\Leftarrow$ $d \le w_i + e_i$ כנדרש.

מטענת העזר $w_i \ge d - e_i$ ומכאן קיבלנו $E(2\varepsilon_i + s_i) = 2 - \left(\frac{2w_i}{d}\right)2 + \frac{2w_i}{d} = 2 - \frac{2w_i}{d} \le 2 - \frac{2(d-e_i)}{d}$

$E(2\varepsilon_i + s_i) \le 2 - \frac{2(d-e_i)}{d} = 2 - \frac{2d-2e_i}{d} = \frac{2e_i}{d}$ כנדרש.

כעת נוכיח את המשפט בעזרת הלמה.

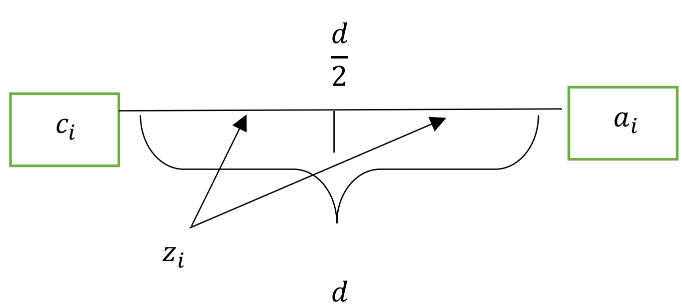
$E(2\varepsilon + s) = \sum_{i=1}^{i=n} E(2\varepsilon_i + s_i)$ מליניאריות התוחלת והגדרת ε ו- s

מהלמה $\sum_{i=1}^{i=n} E(2\varepsilon_i + s_i) \le \frac{2\sum_{i=1}^{i=n} e_i}{d}$

מההנחה כי מספר השגיאות קטן מ- $\frac{dD}{2}$ $\frac{2\sum_{i=1}^{i=n} e_i}{d} \le \frac{\frac{2dD}{2}}{d} = D$

לכן קיבלנו $E(2\varepsilon + s) = \sum_{i=1}^{i=n} E(2\varepsilon_i + s_i) \le \frac{2\sum_{i=1}^{i=n} e_i}{d} \le \frac{\frac{2dD}{2}}{d} = D$ $\Leftarrow$ $E(2e + s) \le D$ משל.

הסתכלנו על שני מקרים:



כשאר המרחק בין c_i ל z_i מייצג את מספר השגיאות, במקרה בו המרחק $> \frac{d}{2}$ אנו נקבל ש $z_i = w_i$

במקרה בו המרחק $\le \frac{d}{2}$ ככל ש w_i קטן יותר, כך נסיק כי יש יותר שגיאות. וככל ש w_i גדול יותר, כך הסיכוי למחוק את הסימבול הלא נכון גבוהה יותר.

קורס: קודים לתיקון שגיאות ושימושיהם במדעי המחשב
מרצה: ד״ר קלים יפרמנקו
סמסטר סתיו תשפ״א
06/12/2020

אלגוריתם *Generalized Minimum distance* דטרמיניסטי:

אנחנו יכולים לבחור מספר אקראי בין אפס ל1 - θ ואם $\theta < \frac{2w_i}{d}$ נבצע מחיקה.

אנו יודעים שקיים θ ככה שהמשתנים המקריים מקיימים את האי שוויון על התוחלת, אחרת אם לכל θ האי שוויון לא היה מתקיים, זו לא הייתה התוחלת. כלומר קיים θ כך ש $2e(\theta) + s(\theta) < D$.

היינו רוצים לעבור כל ערכי θ האפשריים אך קיימים אינסוף.

נמיין את $w_1, w_2, \dots, w_n$ ונניח ש $w'_1, w'_2, \dots, w'_n$ הינו מערך ממוין.

לכל $\theta \in [\frac{2w_i}{d}, \frac{2w_{i+1}}{d}]$ נקבל את אותם המחיקות, לכן לא צריכים למנות את כל ערכי θ, אלא לכל i מספיק למנות ערך θ בטווח הזה.

<u>תיאור האלגוריתם:</u>

מקבל כקלט $z_1, \dots, z_n$

נמיין את $w_1, w_2, \dots, w_n$ ונסמנם $w'_1, w'_2, \dots, w'_n$ ונסמן $Q = \{1,0\} \cup \{w'_1, \dots, w'_m\}$

לכל $\theta \epsilon Q$ בצע:

שלב ראשון: לכל i מצא a_i כך ש $\Delta(c_{in}(a_i), z_i)$ מינימאלי

שלב שני: נגדיר $w_i = \Delta\left(c_{in}(a_i), \frac{d}{2}\right)$

שלב שלישי: אם $\theta < \frac{2w_i}{d}$ נסמן $y_i = '?'$ אחרת $y_i = a_i$

שלב רביעי: הרץ את אלגוריתם הפענוח של c_{out} עבור כל y_i נסמן $c_\theta = c_{out} \circ c_{in}$

שלב חמישי: החזר את c_θ הקרוב ביותר לz_i