19/10 בס"ד

①

דחיסת נתונים א' / דנה שפירא

90% מבחן, 10% 4-5 מטלות.

נושאים:
מבוא
מידול
קידוד סטט'
קידוד' שנון
קידוד' הפמן
קידוד אריתמטי'
קידוד מילוני'
LZ77
LZ78

נושא 1 - מבוא

א) שימושים של דחיסת נתונים

- העברת מידע רב יותר בתשתית מוגבלת.
- שיפור זמני התגובה של אפליקציות מסוימות.
- שיפור יכולות מחשב ע' שמירת מידע רב יותר בזכרון הפנ' ולא בזכרון משני.

ב) מערכת דחיסה

כל מערכת דחיסה מורכבת מ-3 שלבים:

מידול - למידה והנחות על מבנה הקובץ, כמו מהו הא"ב בקובץ.

איסוף סטטיסטי - על הקובץ וכמו כיצד כמות האותיות מתפלגת.

קידוד - בחירת אלגוריתם לקידוד הקובץ. לרוב מעל א"ב בינארי.

בכל דחיסה יש מקודד ומפענח. המקודד מקבל את הקובץ המקורי ומקודד אותו לרצף ביטים. המפענח מקבל את רצף הביטים ומפענח אותם חזרה לקובץ המקורי או קובץ דומה לו.

יש שני סוגים של דחיסה: lossless שאינה מאבדת מידע כך שהקובץ המשוחזר זהה למקורי, בעיקר חשובה בקבצי טקסט. ו-lossy שמאבדת קצת מהמידע אך שומרת את העיקר, משמשת בעיקר בדחיסת תמונות וקבצי סאונד.

ג) סימונים והגדרות

- את קבוצת הא"ב בקובץ נסמן $S=[s_1, s_2, \dots, s_n]$.
- את ההסתברות להופעת האות בקובץ נסמן $P=[p_1, p_2, \dots, p_n]$. נניח כי $p_1 \ge p_2 \ge \dots \ge p_n$, וגם $\sum_{i=1}^{n} p_i = 1$.
- את הקידוד של כל אות נסמן $C=[c_1, c_2, \dots, c_n]$ ואת אורך הקידוד של c_i כלשהו נסמן $|c_i|$.
- התוחלת של אורך קידוד עבור אות אחת נחשב $E(C,P)=\sum_{i=1}^{n} p_i \cdot |c_i|$. ערך זה הוא המדד שלנו ליעילות הדחיסה. ככל ש-$E(C,P)$ נמוך יותר כך הדחיסה טובה יותר.

ד) סוגי קידודים

- **Prefix - Free**: זהו קידוד שבו אין קידוד של מילה שהוא רישא של קידוד מילה אחרת. הדרך הנוח ביותר לשמור קידוד חסר רישות היא באמצעות עץ שבו כל פנייה שמאלה היא 0 ופנייה ימינה היא 1. בעץ כל העלים הם תווים. בכל פענוח של תו נטייל מהשורש עד לעלה. קוד חסר רישות אינו מוריד את איכות הדחי

19/10

②

בס"ד

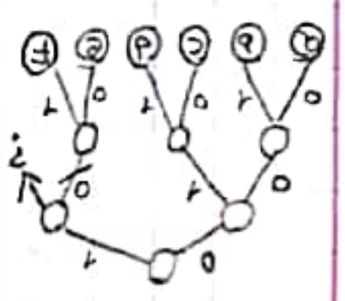
- Uniquely Decipherable (UD): זהו קידוד שבו כל רצף של ביטים שנוצר בעזרתו ניתן לפענח בדרך אחת ויחידה שתחזיר את הקובץ לצורתו המקורית.

משפט: כל קוד שהוא חסר רישות הוא גם UD, אך ההיפך לא נכון.

- Complete Code: זהו קידוד שעבור כל מחרוזת בינארית (לאו דווקא שנוצרה באמצעות הקידוד) ניתן לפענח באופן יחודי. אם נייצג קידוד שלם בעץ אזי העץ יהיה בהכרח מלא, שהוא עץ שבו לכל צומת פנימית יש שני בנים אך אינו בהכרח שלם. עבור קוד שהוא לא שלם, לא כל מחרוזת תוכל לפענות באופן יחודי או תוכל להיות מפוענחת בכלל. בנוסף ניתן יהיה לצמצם ולייעל את הקידוד.

דוגמה לקידוד לא שלם: $\overset{*a}{000}, \overset{b}{001}, \overset{c}{010}, \overset{d}{011}, \overset{e}{100}, \overset{f}{101}$. ניתן לראות שעבור מחרוזת בינארית שבה יש רצף 11 לא יהיה ניתן לפענח אותה, וכן שאת הקידוד $\overset{e}{100}, \overset{f}{101}$ היה אפשר לקצר ל- 10,11 ולקבל קידוד יעיל יותר.

דוגמה נגדית:
0,01,011,0111
נדע שמסתיימת מילה
כשרואים 0 ולכן יש
פענוח יחיד אך קידוד
לא חסר רישות.



- Instantaneous Code: זהו קידוד שבו במעבר על מחרוזת בינארית ניתן לדעת מתי מילה מסתיימת גם בלי שצריך לבחון את הביטים שאחריה. לדוגמה, עבור הקידוד $\overset{a}{0}, \overset{b}{01}, \overset{c}{011}, \overset{d}{0111}$, ניתן לדעת שמילה הסתיימה רק כאשר מגיעים לביט השייך למילה הבאה, ולכן אינו קידוד מ"ד. קידוד מ"ד אינו גורר UD, אמנם קידוד מ"ד גורר קידוד חסר רישות וכן קידוד חסר רישות גורר קידוד מ"ד.

במטלת 1 שקופית
16 קידוד 3 מקיים
את כל התנאים.

מטרתנו היא למצוא קידוד שמקיים את כל ארבעת התנאים לעיל שהוא גם בעל תוחלת מילה הכי קטנה שאפ

ה) מבחן UD - אלגוריתם Sardinas-Patterson

נלמד אלגוריתם פולינומי אשר בהינתן קידוד מחזיר האם הוא UD או לא. לפני תיאור האלגוריתם נסביר שני מושגים. נזכיר בנוסף ש- C היא קבוצת כל המילים המקודדות.

- dangling suffix - עבור שני קידודי מילים a,b: אם a רישא של b אזי שאר הביטים ב-b (שאינם ב-a) הם סיפא תלויה.
- עבור שני קבוצות מילים S,T נגדיר left $\overset{\text{מנה שמאלית}}{\text{quotient}}$ שלהם כך $S^{-1}T=\{d \mid ad\in T, a\in S\}$. כלומר קבוצת כל הסיפות התלויות ב-T שיש להם רישא ב-S.

```
Sardinas-Patterson {
    j = 1
    S_1 = C^-1 C - {ε}    // Si זהו קבוצה של מילים שמכיל את כל הסיפות התלויות שנוסף לה כל איטרציה. בהתחלה מכילה סיפות תלויות של c עם עצמה.
    while (true) {
        S_i+1 = C^-1 S_i U S_i^-1 C    // בכל איטרציה נוסיף ל-Si+1 סיפות תלויות שיש ל-Si באיטרציה הקודמת עם c וההיפך.
        i++
        if ε∈S_i or c∈S_i for c∈C  print not UD, Exit
        else if ∃j<i s.t S_i=S_j  print UD, Exit    // אין שינוי ב-Si. הוספנו את כל הסיפות התלויות האפשריות.
    }
}
```

מטרת האלגוריתם היא למצוא סיפא תלויה ב-Si שהיא גם קידוד של מילה ב-C. אם נמצאה סיפא תלויה כזו אזי הקידוד אינו UD. אך אם אין סיפא כזו נחזיר שהקידוד אכן UD.

דוגמאות ניתן
למצוא במצגת 1
שקופיות 21-22.

בס״ד

26/10

③

1) משפט הקידוד של שנון (Shannon)

כאמור לעיל, המטרה בדחיסת נתונים היא למצוא, עבור קובץ שבו התפלגות P מסוימת, קידוד שעבורו אורך מילת קוד ממוצעת היא מינימלית. אם הקידוד אכן מקיים תנאי זה נאמר שהוא "קידוד בעל יתירות מינימלית", כלומר שהמידע המיותר בו מינימלי. בניסוח פורמלי יותר נאמר ש-C הוא בעל יתירות מינימלית אם מתקיים $E(C,P) \le E(C',P)$ עבור כל קידוד C' אחר.

משפט הקידוד של שנון נותן לנו חסם תחתון ל-$E(C,P)$*, כדי להסביר חסם זה יש להגדיר כמות אינפורמ

כמות אינפורמציה: שנון הגדיר שכמות האינפורמציה החדשה שמכיל תו S_i עם הסתברות P_i מחושבת לפי הנוסחה: $I(S_i) = -\log_2 P_i$.

*חסם תחתון זה הוא אכן מינימלי עבור כל קוד שבו לכל תו בטקסט יש אורך קוד קבוע שאינו משתנה.

תכונות של אינפורמציה - ככל ש-P_i מתקרב ל-1 כך האינפורמציה $I(S_i)$ מתקרב ל-0.

ככל ש-P_i מתקרב ל-0 כך האינפורמציה $I(S_i)$ מתקרב ל-∞.

אינפורמציה של רצף תווים $S_i, S_{i+1}, \dots, S_{i+j}$ מחושב: $I(S_i S_{i+1} \dots S_{i+j}) = I(S_i) + I(S_{i+1}) + \dots + I(S_{i+j})$

אנתרופיה של התפלגות: מידת האינפורמציה הם אורך ביטים. שנון הוכיח שהקידוד הטוב ביותר הוא כזה שנותן לכל תו קוד באורך $I(S_i)$. בצורה פורמלית יותר, עבור התפלגות P של תווים, נגדיר $H(P)$ להיות האורך הממוצע של האינפורמציה עבור כל התווים, $H(P) = -\sum_{i=1}^{n} P_i \cdot \log_2 P_i$. לכל קידוד C מתקיים: $H(P) \le E(C,P)$ כלומר $H(P)$ הוא חסם תחתון ל-$E(C,P)$, לכן תמיד נרצה לתת לתו S_i קידוד באורך $I(S_i)$.

$H(P)$ נקרא "האנתרופיה של התפלגות P".

2) אי-שוויון קראפט-מקמילן (Kraft-McMillan)

לפי חסם שנון לעיל אם קיימת התפלגות של תווים P, כך שלכל תו S_i ההסתברות שלו היא $P_i = 2^{-K_i}$ עבור $K \in \mathbb{N}$, אזי $I(S_i) = K_i$ מספרי שלם. בהתפלגות כזו ניתן ליצור בקלות (נמצא בהמשך) קידוד שהוא בדיוק שווה לחסם שנון. הסבר זה הוא המוטיבציה לפונקציית קראפט על כל קידוד C המוגדרת כך: $K(C) = \sum_{i=1}^{n} 2^{-|C_i|}$. הערך של $K(C)$ יכול ללמד אותנו משפטים רבים חשובים בדחיסת נתונים:

(1) אי שוויון קראפט-מקמילן: קידוד C הוא מ״פ אם ורק אם $K(C) \le 1$.

(2) אם עבור קידוד C מתקיים $K(C) \le 1$ אזי קיים קידוד C' המקיים: $E(C',P) = E(C,P)$, $|C'| = |C|$ ו-C' הוא חסר רישות. כלומר קיים קידוד שקול שהוא חסר רישות. מסיבה זו תמיד נרצה שהקידוד שלנו יהיה חסר רישות, שכן הוא לא מוריד מאיכות הדחיסה ויש לו יתרונות רבים כמתואר לעיל.

מסקנה מ-(2) ו-(3): חסר רישות $\Leftrightarrow K(C) \le 1$.

(3) אם עבור קידוד C מתקיים $K(C) > 1$ אזי הוא בודאי לא חסר רישות, ולא מ״פ ולא UD.

(4) אם קידוד C הוא UD אזי מתקיים $K(C) \le 1$.

הוכחות למשפטים (1) ו-(4) במצגת 2 שקופיות 8-9.

(5) שוויון מקמילן: קידוד C הוא שלם (ניתן לקודד בעזרתו אבל שלא ניתן להוסיף בו עוד מילים) אם $K(C) = 1$.

26/10

(4)

המסקנה מכל האשטיק לעיל היא שעבור התפלגות p של תווים, כדי לקבל קידוד c שהוא חסר רישות, אי־פי, UD שאף בעל יעילות מינימלית, מספיק שיקיים שני תנאים:

(1) $K(c) \le 1$

(2) $E(c,p)$ מינימלי.

אם בנוסף נדרה קוד שלם נדרוש $K(c)=1$.

בס"ד

2/11

⑤

נושא 2 - מידול

א) הגדרה

כאמור לעיל, כל מערכת דחיסה מורכבת מ-3 מרכיבים: מידול, איסוף סטטיסטי ואלגוריתם קידוד. הפרק זה נעסוק במרכיב הראשון ונכיר מספר סוגי מודלים. ההבדלים בין המודלים הוא בכיצד הם מנתחים את הקובץ / הודעה שאנו רוצים לדחוס, וכתוצאה מכך גם גודל ה-header, הנקרא גם "Prelude", שצריך לתת למפענח (decoder) כדי שיוכל לפענח את הדחיסה. עבור כל מודל נחשב את האנטרופיה $H(P)$ ואת האורך הממוצע של קידוד $E(C,P)$, ונבחן את היחס ביניהם.

נסתכל על שני סוגים של משפחות מודלים

(1) Zero-order: מודלים שבהם ההסתברויות שניתנים למילות הקוד הן בלתי תלויות.* נקרא גם "מרקוב מסדר 0"

(2) First-order: מניחים שיש תלות בין מילות הקוד. נתייחס להסתברות של שתי מילות קוד רצופות ביחס למספר המופעים שלה בהודעה. נקרא גם "מרקוב מסדר 1".

* נעסוק שובה סוגי מודלים כאלו.

ב) מודל דחיסה סטטית

המודל זה אין כל למידה על ההודעה שרוצים לדחוס. לדוגמה, נוכל להניח שיש בהודעה את כל התווים בטבלת ה-ASCII, סה"כ 256 תווים. כל תו S_i מקבל הסתברות $P_i=\frac{1}{256}$. קידוד כל תו יעשה לפי הערך של טבלת ASCII שנותן לכל תו קידוד באורך 8 ביטים.

עבור מודל זה האנטרופיה היא: $H(P)=-\sum_{i=1}^{256}\frac{1}{256}\cdot\log_2\left(\frac{1}{256}\right)=8$ [ביטים]. נזכיר שהאנטרופיה היא חסם תחתון לאורך כל קידוד אפשרי. לכן הקידוד שתואם לפי טבלת ASCII, שהוא Fixed-Code, ולכן יתן לנו $E(C,P)=8$, והוא אכן הטוב ביותר עבור מודל זה.

ג) מודל דחיסה סטטית למחצה

זהו מודל שכן לומד את ההודעה שרוצה לדחוס. הוא נותן הסתברות אך ורק לתווים שנמצאים בהודעה M. אמנם כל תו מקבל הסתברות אחידה. נסמן את גודל הא"ב $|\Sigma|$. עבור כל תו S_i ההסתברות שלו תהיה $P_i=\frac{1}{|\Sigma|}$. האנטרופיה עבור מודל זה היא: $H(P)=-\sum_{i=1}^{|\Sigma|}\frac{1}{|\Sigma|}\cdot\log_2\left(\frac{1}{|\Sigma|}\right)$

במודל זה צריך גם header להעביר את כל Σ בקידוד ASCII וכן את מספר התווים $|\Sigma|$. סה"כ אורך ה-header יהיה 8 ביטים עבור $|\Sigma|$ ועוד $8\cdot|\Sigma|$ עבור הקידוד של כל תו.*

* כל תו i יקבל קידוד באורך $\lceil\log_2(|\Sigma|)\rceil$ אמנם במידה ו-$|\Sigma|$ לא חזקה של 2 אפשר לשפר זאת כמו שנעשה בהמשך.

מכאן נחשב אורך קידוד תו ממוצע ב-M: $E(C,P)=H(P)+\frac{|\Sigma|\cdot8+8}{|M|}$.

$|M|$ מספר התווים בהודעה.

דוגמה: עבור הודעה באורך 128 תווים שמכילה 25 תווים שונים נקבל: $(c,P)=\overbrace{-\sum_{i=1}^{25}\frac{1}{25}\cdot\log_2\left(\frac{1}{25}\right)}^{H(P)=4.64}+\frac{25\cdot8+8}{128}=6.27$

מודל זה טוב יותר מדחיסה סטטית. שיפור נוסף הוא שהתו ה-25 שאמור להיות 11000 יקבל רק 11 שכן אין

(6)

ב) מודל בחיסה 666ית למחדה עם הסתברות עצמית

להו מודל שלומד את הקובץ ולכל מילת קוד S_i ניתן את ההסתברות שבחירת מילה אקראית מההודעה/קובץ היא אכן S_i. $P_i = \frac{V_i}{|M|}$, כאשר V_i היא מספר הפעמים של S_i בהודעה.

האנטרופיה עבור מודל כזה היא: $H(P)=-\sum_{i=1}^{|\Sigma|} \frac{V_i}{|M|} \log_2\left(\frac{V_i}{|M|}\right)$. במודל כזה אפשר להגדיר קידודים קבועים יותר מאשר המודלים שלמדנו. החיסרון הוא שה- header של הקובץ הדחוס צריך להיות גדול שכן הוא צריך להכיל את התדירות של כל מילת קוד S_i. $E(C,P)=H(P)+\frac{\text{גודל ה-header}}{|M|}$.

דוגמא: מספר הודעה באורך 128 תווים שמכילה 25 תווים שונים, וכדי להגיע תדירות של כל תו צריך לפחות 4 ביטים נקבל: $E(C,P)=-\sum_{i=1}^{25} \frac{V_i}{128} \log_2\left(\frac{V_i}{|M|}\right)+\frac{4\cdot 25+8\cdot 25+4}{128} = 6.63$

V_i נלקח מטבלה במגבלת 2 שקופית 17.

ניתן לראות שקידת פחות טוב מהדוגמה במודל הקודם. אלא בהוספת טבלת מודל זה הרבה יותר טוב.

ה) מודל סדר-ראשון

עד כה שלושת המודלים הין מסוג zero-order שבן לא היה תלות בהסתברות בין שני תווים.

במודל first-order מחלקים את ההודעה לכל שני תווים סמוכים S_i ו-S_j, ואז כל מילת קוד S_iS_j תקבל הסתברות $P_{ij} = \frac{V_{ij}}{|M|}$, כאשר V_{ij} הוא מספר הפעמים של S_iS_j בהודעה.

בשיטה בו מודל ה- header מאוד גדול אך האנטרופיה הטבה יותר. לכן מודל כזה שימושי רק בהודעות/קבצים גדולים מאוד.