

פתרונות למבחנים

דחיסת נתונים א'

מדעי המחשב, אוניברסיטת אריאל

מס'	שנה	סמסטר	מועד	עמוד	Check ✓
			סיכום הקורס		
			אלגוריתמים		
1	2015	סמסטר א'	מועד א'	20	
2	2016	סמסטר א'	דוגמה	29	
3	2016	סמסטר א'	מועד א'	43	
4	2017	סמסטר א'	מועד א'	51	
5	2018	סמסטר א'	מועד א'	57	
6	2018	סמסטר א'	מועד ב'	66	
7	2019	סמסטר א'	מועד א'	73	
8	2021	סמסטר א'	מועד א'	78	
9	2021	סמסטר א'	מועד ב'	84	
10	2022	סמסטר א'	מועד א'	93	
11	שאלות שעוד לא הופיעו במבחנים			102	

אל תפנו אלי בפרטי, אני לא המרצה/מתרגל שלכם.

הפתרונות בספר לא נבדקו ע"י מרצים, יכול להיות שתמצאו טעויות, באחריותכם בלבד!

בהצלחה 😊

פתרונות למבחנים - דחיסת נתונים א'

נערך ונכתב על-ידי דור עזריה

2022

ספר זה לא נבדק על ידי מרצה, יתכן שימצאו טעויות.

לסיכומים נוספים שלי במדעי המחשב ומתמטיקה:

<https://dorazaria.github.io/>

מוזמנים לעקוב אחרי 😊:

 LinkedIn: <https://www.linkedin.com/in/dor-azaria/>

 GitHub: <https://github.com/DorAzaria>

סיכום הקורס

קוד חסר רישות Prefix Free Code

קוד פרפיקסי, בו אף מילת קוד אינה רישא של מילה אחרת, כל קוד פרפיקסי יפוענח UD. כלומר כל פרפיקסי הוא UD, אך לא כל UD הוא פרפיקסי.

קוד (Uniquely Decipherable) UD

ניתן לפענח בצורה אחת בלבד
כדי להפריך UD, נביא מחרוזת בינארית בעלת שני פירושים שונים,
לדוגמא: $a=0, b=1, c=10$ אז $c=10$ אבל גם $ba = 1+0=10$.. לכן זה לא UD

$$\text{Code efficiency} = \text{Entropy}(H) / \text{Average codeword length}(N)$$

קוד אופטימלי

הוא קוד שמיוצג ע"י עץ מלא שלכל צומת יש 2/0 בנים

קוד שלם

קוד בו כל מחרוזת אינסופית תפוענח בצורה ייחודית ותיצור עץ מלא, אך בהוספת מילת קוד למחרוזת - הקוד לא יהיה ייחודי.
איך להפריך קוד שלם? נראה בו מחרוזת שאינה ניתנת לפענוח.

קוד מייד

קוד פרפיקסי, רק ברגע שהמילה מסתיימת, המפענח ידע לפענח. לא כל UD הוא קוד מייד!

Dangling Suffix

אם מחרוזת a היא תחילת המחרוזת b למשל נראה דוגמה:
• לדוגמה: $a = 010$ ו- $b = 01011$, אז ה-dangling suffix הוא 11 (לאחר השמטת הרישא מ-b).

אלגוריתם Sardinas-Patterson - בודק אם קוד הוא UD

הרעיון בו הוא להוסיף את כל dangling suffix האפשריים מתוך מילות הקוד הנתונות. אם אחד מהסיומות שהוספנו היא מילת קוד שהייתה נתונה ברשימה - הקוד אינו UD. אם היא לא הייתה לפני כן ברשימה - הקוד UD. לדוגמא - קיבלנו (0, 10, 1, 01). 1 היא התחלה של 10, אז ה-suffix d מחזיר לנו את ה-0 שנשאר, והוא קיים ברשימה.. לכן הקוד אינו UD.

קודי יתירות מינימליים

זה הקידוד שייתן לי אורך מילה ממוצעת הכי קצרה בקוד

אינפורמציה

רמת ה"אי ודאות" - כמות הביטים המינימלית לייצוג. מחושב על סמך הסתברות. אם היא 1 אז האינפ' = 0. אם ההסתברות אפס אז אינפ' = אינסוף. ככל שההסתברות גדולה יותר כך האינפורמציה קטנה יותר.

משפט הקידוד של שאנון

מגדיר פונקציה H שתשמש ממוצע משוקלל של האינפורמציה. במילים אחרות, קצב הדחיסה המינימלי האפשרי של המקור, לכן מהווה חסם תחתון.

אנטרופיה

תוחלת על האינפורמציה, המשמשת מדד לגודל מרחב ההסתברות. כלומר, ככל שהיא גדולה יותר - כך קשה יותר לחזות את התוצאה. אנטרופיה זה בעצם ממוצע משוקלל של כמות המידע. בהינתן הסתברות בהתפלגות P , נגדיר:

$$H(P) = - \sum_{i=1}^n p_i \cdot \log_2 p_i$$

אי שוויון קראפט

תנאי לשיון מילים לצמתי עץ, כך שתשוויך מילה אחת בלבד לאורך כל המסלול. הולך כך: סיגמא של שתיים בחזקת מינוס האורך הראשון של המילה ה- i - אם סכום השברים קטן או שווה מאחד - התנאי מתקיים.

- אם $K(C) = 1$ - העץ מלא, אם קטן ממש מאחד - זה קוד פרפיקסי ובהכרח גם UD.
- אם $K(C) \leq 1$ אז קוד הוא ייחודי.
- אם $K(C) > 1$ - בהכרח הקוד לא פרפיקסי ולא UD

בשאלות בסגנון "האם ניתן לבנות קוד פרפיקסי ממילות קוד נתונות?" נציב אותן באי השוויון. אם יצא ערך גדול מ-1 זה אומר שלא ניתן. אם C הוא UD אז:

$$K(C) = \sum_{i=1}^n 2^{-|c_i|} \leq 1$$

דחיסה סטטית

נקבעת לפני הקידוד ע"פ ASCII ולא משתנה. לכן הסתברות ואז אנטרופיה קבועה.

דחיסה סטטית למחצה

נספור כמה תווים (אותיות) שונים יש ונניח שההסתברות אחידה, לדוג':
 "I am hungry" יש 9 תווים שונים לכן $p_i=1/9$ וכעת נוח להציב ולחשב את האנטרופיה.
 נצטרך להכפיל כל מילת קוד ב-8 ביטים של קידוד שיהיו הסבר למפענח. לכן יש להוסיף בחיבור לחישוב הישיר של האנטרופיה את:
 8 הביטים כפול התווים השונים (9 בדוגמא) + 8 חלקי כמות האותיות הכוללת בהודעה (גם תשע כי כולם שונים זה מזה)

מודל סטטי למחצה עם הסתברויות עצמאיות

הפעם ההסתברות לא אחידה, ותחושב ע"י כמות הפעמים שהתו הופיע חלקי גודל ההודעה.

קוד אונרי

קוד קבוע מראש, עם 0 כהפרדה בין כל מילת קוד. ייראה כך: 0,10,110,1110 וכו' בקוד אינסופי, כאשר כמות האינפורמציה (ביטים מינימלית לייצוג תו) תהיה שווה לאורך מילת הקוד - הקוד האונרי יהיה ללא יתירות (אורך מילה ממוצעת מינימלי) בקוד סופי, פשוט נקצץ את הביט האחרון, ה-0 החוצץ בסוף, בכדי שיהיה קוד ללא יתירות.

$$x_i = 1^{i-1}0$$

קוד בינארי פשוט

קוד בעל אורך קבוע.

קוד בינארי מינימלי

אפשר אורכים שונים במילות הקוד וכך לייעל את הקידוד.

קוד Elias

שתי שיטות קידוד כך שהביטים יהיו בסדר גודל לוגריתמי של ההודעה:
1. גמא - בתחילה, כותבים בקוד אונרי (על פי הנוסחה) של כמות הביטים בבינארי אליו נשרשר מימין את הבינארי של X ללא ה-1 המוביל (השמאלי ביותר) כי הוא מיותר. לדוגמא: כמה ביטים דרושים לייצוג בינארי של 5? 3 סה"כ. אז נחשב 3 כקוד אונרי ונקבל שני אחדות ותמיד אפס בסוף, כלומר: 110 עכשיו, הבינארי של 5 - זה 101. נעיד את ה-1 המוביל ונישאר עם 01. לכן קיבלנו: 11001

2. דלתא - בצד שמאל נכתוב את מספר הביטים כייצוג בינארי אך הפעם כגמא ומצד ימין שוב את הבינארי בלי ה-1 המוביל.

בשאלה אם קוד מסויים הוא UD

אפשר להציב באי שיוויון קרפאט, אם קטן שווה מאחד התשובה היא כן, כי זה תנאי הכרחי. אם הוא גדול - הוא בהכרח לא פרפיקסי/UD

בשאלה אם קוד הוא אופטימלי

נחשב את אורך המילה הממוצעת שלו, נחשב גם את האורך מילה ממוצעת של האפמן (ידוע שבהכרח אופטימלי). אם אורך מילה ממוצעת של הקוד קטנה שווה מהאפמן - זה אופטימלי.

חישוב אורך מילה ממוצעת

נסכום את השכיחויות, נגיד $20=1+6+13$. עכשיו ההסתברויות p_i זה שכיחות חלקי 20. נעשה סיגמא של כל p_i מוכפל בשכיחות שלו ($6/20*6+13/20*13..$), התוצאה זה אורך מילת הקוד הממוצעת.

חישוב אורך מילה ממוצעת של האפמן

נכתוב את השכיחויות כעלים בעץ, כדאי מקטן לגדול, וליד כל אחד נרשום איזה תו הוא. נחבר כל פעם בין צמד העלים הכי קטנים מספרית עד שלכולם יהיה אבא. נקבע שרירותית שצלע ימין זה אפס ושמאל זה אחד או להפך. נספור מה הגובה כל פעם כדי להגיע לתו (נגיד עברנו על 0110 להגיע לעלה a - אז 4). כל גובה נכפיל בהסתברות (שכיחות ספציפית של תו חלקי סכום כל השכיחויות) ונחבר ביניהם, הסכום הוא אורך מילה ממוצעת של האפמן.

Reversible Variable Length Codes = fix free = affix code = קוד גם פרפיקסי וגם סופיקסי

- קוד אפיקסי לא ניתן לסנכרון.

הוכח/הפריך קוד UD

אי שוויון קראפט ואלג' סרדינס כדי להוכיח ולהפריך עם שתי מחרוזות שמביאות אותו דבר - אפשר להפריך

קוד מיידי הוא לא בהכרח קוד שלם

LZ77

בהתחלה השלשה תמיד תהיה (תו שלא קיים במחרוזת בכדי שנדע לגשת למיקום ההתחלתי, 0,0) לאחר מכן השלשות יהיו (בניסיון להגיע למופעי עבר של התו/מחרוזת, אם יש כאלה): (התו הבא בתור, אורך של תווים חוזרים, מס' תווים אחורה). עדיף להפנות למופע האחרון שראינו. מה שמופיע לראשונה, נגיד b, נראה (התו בי, 0,0)

LZSS

במקום שלשה כמו ב77, נעשה זוג סדור (כמה תווים להעתיק, כמה תווים ללכת אחורה). בגדול הרעיון בSS זה לא להעתיק תווים בודדים (כי זה לא משתלם)

אלא לתפוס את הדגים השמנים

LZ78

נמספר את האינדקסים ונאתחל במחרוזת הריקה.
נחפש כל פעם את המחרוזת הארוכה בכל המילון (גם קדימה) שהיא רישא של התו הנוכחי עליו אנחנו עוברים.
אם אין, אז נפנה לאפסילון (0).
נניח ואנחנו בתו בי אז זה ייראה כך (התו בי, 0)

אם יש מופע כזה, נכתוב (התו הבא אליו הגענו, אינדקס של תחילת המחרוזת החופפת)

מתי מילון גדל?

כאשר אנו מוסיפים למילון שורה חדשה, בכל פעם שנעבור במילון גודל שהוא חזקה של 2 אז אנחנו מוסיפים סיבית נוספת לקידוד וזה גורם למילון להיות מוכפל בגודלו פי 2.
כאשר אנחנו בונים את המילון, נסמן ב output כל פעם שאנחנו מגדילים את המילון.

קוד אריתמטי

כשבהאפמן יש תו עם הסתברות גבוהה יש איתו בעיות,
לכן בקוד אריתמטי ניקח זרם של תווי קלט ונחליף אותם במספר בין (0,1) . ככל שההודעה מורכבת וארוכה - כך יותר ביטים ייצגו את ה output.
נכיר את הנוסחא קודם: $UL = LL + d(p)(d)$.
כלומר התו הבא מלמטה למעלה, שווה לתו התחתון ביותר + (התו העליון ביותר פחות התחתון ביותר) כפול ההסתברות המקורית של האות עליה אנחנו נמצאים.

אלגוריתמים

UD Test Algorithm - אלגוריתם

1:	נגדיר רשימה חדשה L ונכניס לתוכה את כל מילות הקוד הקיימות ב- C .
2:	עוברים על כל זוגות מילות קוד ב- L ובודקים האם אחת היא הרישא של האחרת.
3:	כל עוד a היא רישא של b , אנחנו מוסיפים את ה-dangling לרשימה L עד אשר מתקיים אחד מהתנאים הבאים:
4:	1. אם ה-dangling שהוספנו ל- L היא כבר מילת קוד מקורית ב- C - אז הקוד לא UD ונעצור.
5:	2. אם ה-dangling שהוספנו ל- L לא מילת קוד מקורית אז נמשיך הלאה ונעצור את האלגוריתם אם עברנו על כל צירופי הזוגות האפשריים ונחזיר כי הקוד הוא כן UD .

Sardinas-Patterson Algorithm

```

1:   $i \leftarrow 1$ 
2:   $S_1 \leftarrow C^{-1}C - \{\epsilon\}$ 
3:  while true:
4:       $S_{i+1} \leftarrow C^{-1}S_i \cup S_i^{-1}C$ 
5:       $i = i + 1$ 
6:      if  $\epsilon \in S_i$  or  $c \in S_i$ :
7:          for  $c$  in  $C$ :
8:              print "not UD"
9:              exit
10:     else if  $\exists j < i$  such that  $S_i = S_j$ :
11:         print "UD"
12:         exit

```

Elias Code C_γ Algorithm

```

1: Elias_Code_ $C_\gamma$ (x) {
2:     w = Binary(x);
3:     out1 = Unary(|w|);
4:     out2 = w - (first 1); // remove first '1'
5:     return out1*out2; // merge two strings
6: }
```

Elias Code C_δ Algorithm

```

1: Elias_Code_ $C_\delta$ (x) {
2:     w = Binary(x);
3:     out1 = Elias_Code_ $C_\gamma$ (|w|);
4:     out2 = w - (first 1); // remove first '1'
5:     return out1*out2; // merge two strings
6: }
```

Golomb Encode Algorithm (הצפנה)

```

1: Golomb_encode(x,b) {
2:     q ← (x-1) div b; // החלק השלם
3:     r ← x-q*b; // השארית
4:     Unary_encode(q+1); // מקודדים באונרי את השארית
5:     Minimal_binary_encode(r,b); // מקודדים את השאר
6: }
```

Golomb Decode Algorithm (פענוח)

```

1: Golomb_decode(b) {
2:     q ← Unary_decode() - 1;
3:     r ← Minimal_binary_decode(b);
4:     return r + q * b;
5: }

```

Fibonacci Binary Encode Algorithm (הצפנה)

```

1: Fibonacci_Binary_Encode(n) {
2:     find the largest Fibonacci number ≤ n.
3:     Repeat recursively with  $n - F_k$  until  $n - F_k = 0$ .
4: }

```

Fib1 Algorithm - אלגוריתם

1:	נמיר כל מספר דצימלי לפיבונאצ'י בייצוג בינארי על ידי "הצבת אחדות" בסיבית המתאימה.
2:	נוסיף 1 מוביל לכל מילה.
3:	נהפוך את כל המחרוזות.

Fib2 Algorithm - אלגוריתם

1:	בונים לפי Fib1.
2:	מורידים את ה-1 שהוספנו מימין בסוף כל מילת קוד.
3:	נוסיף "10" לכל מילת קוד בהתחלה (משמאל).
4:	נדחוף את כל הקודים שהיו קיימים רמה אחת מתחת (כלומר 2 יקבל את מה שהיה ל-1 ו-3 יקבל את מה שהיה ל-2 וכו...) ואז נוסיף את 1 כמילת קוד חדשה.

Shannon-Fano Algorithm

- 1 עבור רשימה נתונה של סמלים, יש להתאים עבור כל סמל את ההסתברות שלו בקטע המקור.
- 2 נמין את רשימת הסמלים לפי ההסתברות שלהם כאשר הגבוהה ביותר נמצאת בשמאל והנמוכה ביותר בימין.
- 3 נפצל את הרשימה לשתי קבוצות כך שסכום ההסתברויות בכל קבוצה שווה כמה שאפשר לקבוצה השנייה.
- 4 החלק השמאלי של הרשימה מקבל את הספרה הבינארית "0" והימני מקבל "1".
- 5 חזר באופן רקורסיבי על חלקים 3 ו-4 עבור כל אחד משני החלקים - חלק בשנית כל חלק והוסף ביטים לקוד, עד שכל סמל הופך לעלה בעץ ויש לו קוד מתאים.

Huffman Algorithm

```

1: Huffman ( $\Sigma$ ) {
2:      $n \leftarrow |\Sigma|$ 
3:      $Q \leftarrow \Sigma$ 
4:     for  $i \leftarrow 1$  to  $n-1$  do:
5:         do ALLOCATE-NODE ( $z$ )
6:          $\text{left}[z] \leftarrow x \leftarrow \text{EXTRACT-MIN}(Q)$ 
7:          $\text{right}[z] \leftarrow y \leftarrow \text{EXTRACT-MIN}(Q)$ 
8:          $w[z] \leftarrow w[x] + w[y]$ 
9:         INSERT ( $Q, z$ )
10:    end-for
11: return EXTRACT-MIN ( $Q$ )
12: }
```

שלבי האלגוריתם של הפמן

1. צור תור עדיפות Q שיכיל צמתים המייצגים את כל הסמלים, ממוינים על פי הסתברות הופעתם.
2. כל עוד בתור Q יש יותר מצומת אחת, בצע:
 - a. צור צומת חדש z .
 - b. הוצא מתור העדיפויות את שני האיברים העליונים, x, y - בעלי ההסתברות הנמוכה ביותר.
 - c. הפוך את x, y לבנים הימני והשמאלי של z .
 - d. קבע את ההסתברות של z להיות סכום ההסתברויות של x ו- y .

e. הכנס את z לתור העדיפויות.

3. הצומת הבודד שנותר בתור העדיפויות הוא שורש עץ הקוד המבוקש.

4. הקוד עבור כל סמל יקבע לפי המסלול מהשורש לעלה שמייצג את הסמל. אם הצלע ה- n ית מובילה לבן שמאלי, ערך הביט ה- n י יהיה "0". אם היא מובילה לבן ימני, ערכו יהיה "1".

Huffman Assigning Codewords Algorithm

```

1: HuffmanAssigningCodewords ( $\Sigma, \{\ell_i\}$ ) {
2:   maxlength  $\leftarrow \max(\{\ell_i\})$ 
3:   // find the number of codewords of each length
4:   for  $\ell \leftarrow 1$  to maxlength do:
5:     num[ $\ell$ ]  $\leftarrow 0$ 
6:   for  $i \leftarrow 1$  to  $n$  do:
7:     num[ $\ell_i$ ]++
8:   // store the first codeword
9:   firstcode[maxlength]  $\leftarrow 0$ 
10:  for  $\ell \leftarrow \text{maxlength}-1$  down to 1 do:
11:    firstcode[ $\ell$ ]  $\leftarrow (\text{firstcode}[\ell+1] + \text{num}[\ell+1])/2$ 
12:  for  $\ell \leftarrow 1$  to maxlength do:
13:    nextcode[ $\ell$ ]  $\leftarrow \text{firstcode}[\ell]$ 
14:  for  $i \leftarrow 1$  to  $n$  do:
15:    codeword[ $i$ ]  $\leftarrow \text{nextcode}[\ell_i]$ 
16:    symbol[ $\ell_i$ , nextcode[ $\ell_i$ ]-firstcode[ $\ell_i$ ]]  $\leftarrow i$ 
17:    nextcode[ $\ell_i$ ]++
18: }
```

Canonical Huffman Decoding Algorithm

```

1: CanonicalHuffmanDecoding() {
2:      $v \leftarrow \text{nextInputBit}()$ 
3:      $l \leftarrow 1$ 
4:     while  $v < \text{firstcode}[l]$  do:
5:          $v \leftarrow 2v + \text{nextInputBit}()$ 
6:          $l++$ 
7:     end-while
8:     // the integer  $v$  is a valid codeword of  $l$  bits
9:     return  $\text{symbol}[l, v - \text{firstcode}[l]]$ 
10: // return the index of the decoded symbol
11: }
```

Dynamic Huffman Algorithm

```

1: Update()
2:      $q \leftarrow \text{leaf}(x_t)$ 
3:     if ( $q$  is the 0-node)
4:         replace  $q$  by a parent 0-node with two 0-node children;
5:          $q \leftarrow \text{left child}$ ;
6:     if ( $q$  is a sibling of a 0-node)
7:         interchange  $q$  with the highest numbered leaf of the same
            weight.
8:         increment  $q$ 's weight by 1;
9:          $q \leftarrow \text{parent of } q$ 
10:    while ( $q \neq \text{root}$ )
11:        interchange  $q$  with the highest numbered node of the same
            weight.
12:        increment  $q$ 's weight by 1;
13:         $q \leftarrow \text{parent of } q$ 
14:    increment  $q$ 's weight by 1;
```

Skeleton Huffman Trees Decoding Algorithm

```

1: tree_pointer ← root
2: i ← 1
3: start ← 1
4: while i < length_of_string do:
5:     if string[i]=0 then:
6:         tree_pointer ← left(tree_pointer)
7:     else:
8:         tree_pointer ← right(tree_pointer)
9:     if value(tree_pointer) > 0 then:
10:        codeword ← string[start... (start+value(tree_pointer)-1)]
11:        output ← table[ I(codeword) - diff[value(tree_pointer)] ]
12:        tree_pointer ← root
13:        start ← start + value(tree_pointer)
14:        i ← start
15:    else:
16:        i++

```

Encoding Arithmetic Code Algorithm

```

1: low ← 0.0
2: high ← 1.0
3: while input symbols remain do:
4:     range ← high - low
5:     get symbol
6:     high ← low + high_bound(symbol)*range
7:     low ← low + low_bound(symbol)*range
8: end-while
9: output any value in [low, high)

```

Decoding Arithmetic Code Algorithm

```

1: encoded ← get(encoded number)
2: do {
3:     Find symbol whose range contains encoded
4:     Output the symbol
5:     range ← high(symbol) - low(symbol)
6:     encoded ← (encoded - low(symbol))/range
7: } until (EOF)

```

LZ77 Encoding Algorithm - קידוד

```

1: p ← 1 // the next char to be coded
2: while there is text remaining to be coded do:
3:     search for the longest match for S[p..] in S[p-W..p-1]
        suppose that the match occurs at position m with length l
4:     output the triple(p-m, l, S[p+l])
5:     p ← p+l+1

```

LZ77 Decoding Algorithm - פענוח

```

1: p ← 1 // the next char to be decoded
2: for each triple (f, l, c) in the input do:
3:     S[p..p+l-1] ← S[p-f ... p-f+l-1]
4:     S[p+l] ← c
5:     p ← p+l+1

```

LZW Encoding Algorithm - קידוד

```

1: Dictionary ← single characters
2: w ← first char of input
3: repeat do:
4:     k ← next char
5:     if (EOF) then:
6:         output code(w)
7:     else if (w·k) ∈ Dictionary then:
8:         w ← w·k
9:     else: // if w·k is not in the dictionary then add it.
10:        output code(w)
11:        Dictionary ← w·k
12:        w ← k
13:    end-if-else
14: end-repeat

```

LZW Decoding Algorithm

```

1: Initialize table with single character strings
2: OLD = first input code
3: output translation of OLD
4: while not end of input stream do:
5:     NEW = next input code
6:     if NEW is not in the string table then:
7:         S = translation of OLD
8:         S = S · C
9:     else:
10:        S = translation of NEW
11:    output S
12:    C = first character of S
13:    Translation(OLD) · C to the string table
14:    OLD = NEW
15: end-while

```

Binary Run-Length Encoding Algorithm

```

1:  MaxCount  $\leftarrow 2^k - 1$  ; count  $\leftarrow 0$  ; PrevBit  $\leftarrow$  "0";
2:  while input remains do:
3:      CurrentBit  $\leftarrow$  next input bit
4:      if PrevBit == CurrentBit and count < MaxCount then:
5:          count++
6:      else then:
7:          output count using k bits
8:          if count == MaxCount and PrevBit  $\neq$  CurrentBit then:
9:              output 0 using k bits
10:         count  $\leftarrow 1$ 
11:         PrevBit  $\leftarrow$  CurrentBit
12: output count using k bits

```

Binary Run-Length Decoding Algorithm

```

1:  MaxCount  $\leftarrow 2^k - 1$ 
2:  parity  $\leftarrow 0$ 
3:  while input remains do:
4:      read k bits from the input stream to get the integer x.
5:      if parity == 0 then:
6:          output x times "0" bits
7:      else then:
8:          output x times "1" bits
9:      if x < MaxCount then:
10:         parity := 1 - parity
11: end-while

```

BWT Reconstructing T Algorithm

```

1: BWT(L, index) {
2:     F ← sort(L)
3:     I ← index
4:     for (i=0; i<n; i++) {
5:         T[i] ← F[I];
6:         I ← S[I]
7:     }
8: }
```

Exclusion Principle (EP) Algorithm

```

1: estimate_prob(k) {
2:     sum ← count['$'];
3:     for(i = 1; i <= |Σ|; i++) {
4:         if(k ==  $k_{max}$ ) {
5:             count[i] ← count[ $a_i$  | prev k symbols]
6:         }
7:         else if(k >= 0) {
8:             count[i] ← (count[ $a_i$  | prev k+1 symbols]==0) *
                        (count[ $a_i$  | prev k symbols])
9:         }
10:        else {
11:            count[i] ← (count[ $a_i$ ]==0)
12:        }
13:        sum += count[i];
14:    }
15:    for(i=1 i <= |Σ|; i++) {
16:        p(encode  $a_i$ ) ← count[i]/sum;
17:    }
18:    p(encode '$') ← count['$']/sum;
19: }
```

אלגוריתם Sequiter

החוקים:

1. לא נשאיר 2 תווים סמוכים שיחזרו על עצמם יותר מפעם אחת.
2. חוק צריך להופיע לפחות פעמיים.

האלגוריתם

כל עוד אנחנו לא ב-EOF, אז נבדוק:

1. אם יש 2 תווים צמודים שחוזרים על עצמם 0 ניצור חוק דקדוק חדש שמתקן את זה.
2. אם יש חוק שמשתמשים בו רק פעם אחת - נסיר אותו ונציג במקומו את הצורה המוכלת שלו.

אלגוריתם Re-Pair

1. נמצא את הזוג הכי שכיח ab בטקסט S .
2. ברגע שמצאנו אותו, פשוט ניצור חוק דקדוק $A \rightarrow ab$.
3. נחליף את כל המופעים של ab בטקסט S על ידי A .
4. נפעל כך שוב ושוב עד שכל זוג יופיע רק פעם אחת.

אלגוריתם	מה יושב ב-Prelude?
מודל מרקוב מסדר ראשון	$prelude = 8 \cdot \Sigma + 8 + \Sigma ^2$ ההסתברות של כל תו מופיע אחרי כל תו, אנחנו נבנה טבלה $\Sigma \times \Sigma$, אנחנו לא יודעים מי לא מופיע אחרי מי לכן אנחנו חייבים להגדיר גודל טבלה כזאת גדולה.

מבחן 2015 - סמסטר א' - מועד א'

שאלה 1

הוכח או הפרך:

א. נתונים אורכי מילות הקוד $\{1, 2, 3, 3, 4\}$.

לא קיים קוד חסר רישות ($prefix - free$) בעל אורכים הנ"ל.

ב. הקוד הבא הינו קוד אופטימלי:

קוד	שכיחות	תו
000	1	a
001	3	b
100	2	c
101	4	d
11	5	e
01	6	f

פתרון סעיף א':

אם $K(C) > 1$ אז הקוד C לא פרפיקסי.

הוכחה. נניח בשלילה שקיים קוד פרפיקסי C עם האורכים האלה, אז:

$$K(C) = \sum_{i=1}^n 2^{-|c_i|} = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-3} + 2^{-4} = \frac{17}{16} > 1$$

ומכאן C לא קוד פרפיקסי, בסתירה למשפט קראפט.

הסבר סעיף א':

נחזור על ההגדרות שלמדנו בהרצאה עבור $K(C)$:

- אם $K(C) \leq 1$ אז ניתן לבנות קוד חסר רישות עם אותם אורכים (UD).
- אם $K(C) > 1$ אז C הוא לא קוד חסר רישות ובפרט לא ייחודי.

פתרון סעיף ב':

הפרכה. נתון לנו קוד C . אנו יודעים שקוד האפמן שנסמנו C' הוא קוד אופטימלי.

אנחנו נראה ש: $E(C', P) < E(C, P)$ כלומר שהקוד C הוא לא האופטימלי.

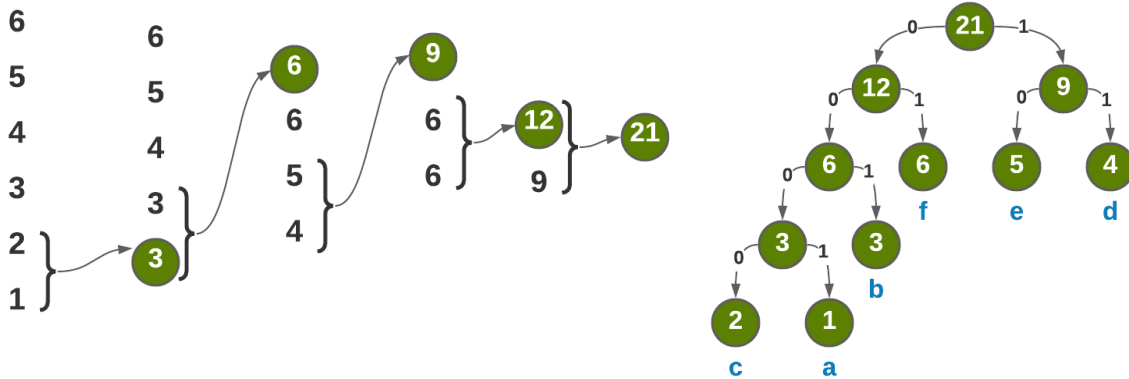
1. נסכום את כל השכיחויות כדי לדעת מה מספר התווים בסה"כ $1 + 3 + 2 + 4 + 5 + 6 = 21$.

2. נחשב את אורך מילת הקוד הממוצעת עבור $E(C, P)$:

$$\begin{aligned} E(C, P) &= \sum_{i=1}^n |c_i| \cdot p_i = 3 \cdot \left(\frac{1}{21} + \frac{3}{21} + \frac{2}{21} + \frac{4}{21} \right) + 2 \cdot \left(\frac{5}{21} + \frac{6}{21} \right) = \\ &= 3 \cdot \left(\frac{10}{21} \right) + 2 \cdot \left(\frac{11}{21} \right) = \frac{52}{21} = 2.476 \end{aligned}$$

כלומר $E(C, P) = 2.476$.

3. נחשב את הופמן C' עם השכיחויות הנתונות (שמאל) ואז נצייר עץ (ימין):



4. נציב את קוד ההופמן שקיבלנו בטבלה ונחשב את אורך מילת הקוד (של הופמן) הממוצעת עבור

$E(C', P)$:

s_i	c'_i	p_i	$ c'_i $
a	0001	1/21	4
b	001	3/21	3
c	0000	2/21	4
d	11	4/21	2
e	10	5/21	2
f	01	6/21	2

$$\begin{aligned} E(C', P) &= \sum_{i=1}^n p_i \cdot |c'_i| = \\ &= \frac{1}{21} \cdot 4 + \frac{3}{21} \cdot 3 + \frac{2}{21} \cdot 4 + \\ &+ \frac{4}{21} \cdot 2 + \frac{5}{21} \cdot 2 + \frac{6}{21} \cdot 2 = \\ &= \frac{17}{7} = 2.428 \end{aligned}$$

כלומר $E(C', P) = 2.428$

5. קיבלנו $E(C', P) < E(C, P)$ כלומר $2.428 < 2.476$ ולכן הקוד C לא קוד אופטימלי.

הסבר סעיף ב': הפתרון מסביר כבר מספיק לפי החלוקה לשלבים.

שאלה 2

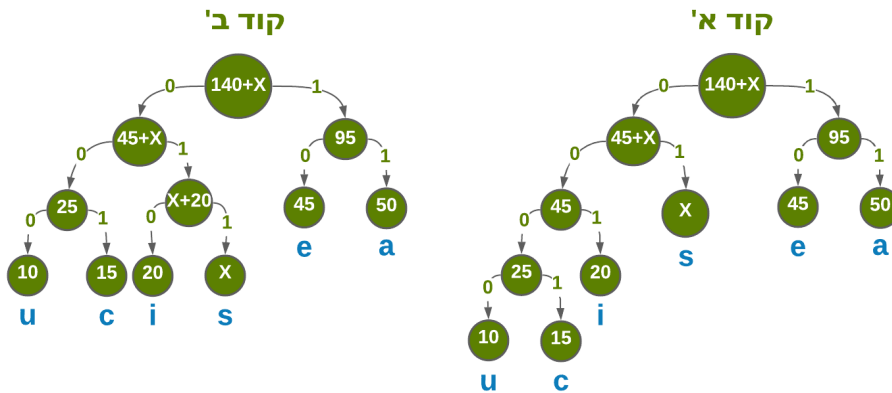
תו	שכיחות	קוד א'	קוד ב'
a	50	11	11
c	15	0001	001
e	45	10	10
i	20	001	010
s	X	01	011
u	10	0000	000

על טבלת התווים והשכיחויות הבאה הופעל אלגוריתם הפמן בשתי צורות כך שהתקבלו שני קודים: **א.** שרטט את שני עצי הפמן המתאימים לקודים שהתקבלו.

ב. חשב את X (שכיחות הסימון S). תן תשובה מספרית מדויקת!

ג. פענח את המילה הבאה (משמאל לימין) על-פי קוד ב':
1100100110011011

פתרון - סעיף א':



הסבר - סעיף א':

פשוט בונים את העצים לפי המספרים בטבלה כי נתון לנו מראש שהעצים האלה הם עצי האפמן.

פתרון - סעיף ב':

כיוון ששני העצים הם עצי האפמן אז צריכה להיות להם את אותה התוחלת ולכן צד שמאל (א') צד ימין (ב'):

$$2(50 + 45 + X) + 3(20) + 4(15 + 10) = 2(50 + 45) + 3(15 + 20 + 10 + X)$$

$$2(95 + X) + 160 = 190 + 3(45 + X)$$

$$2X + 350 = 325 + 3X$$

$$25 = X$$

פתרון - סעיף ג':

נעזר בטבלה ונפתור: **1100100110011011**

פשוט נפריד למילים לפי כל c_i בקוד ב':

11 001 001 10 011 011
a c c e s s

ואחרי הפענוח קיבלנו: **access**.

שאלה 3

נתון הא"ב $\Sigma = \{a, b\}$ בהסתברויות $P = \{0.8, 0.2\}$ בהתאמה ונתונה המחרוזת הבאה: $aaaab$.
א. חשב מבחינה תיאורטית את מספר הסיביות המינימלי הנחוץ לקידוד המחרוזת (האנטרופיה).
ב. חשב מבחינה תיאורטית את מספר הסיביות המינימלי הנחוץ לקידוד האינטרוול בקידוד אריתמטי.
ג. חשב ערך סופי היכול לשמש לקידוד ההודעה ע"פ קידוד אריתמטי.

פתרון - סעיף א':

$$-(4 \cdot 0.8 \cdot \log_2(0.8) + 1 \cdot 0.2 \cdot \log_2(0.2)) = 1.4946$$

אנו יודעים כי האנטרופיה מחשב את ממוצע אורך המילה ולכן נכפיל ב-5 ונקבל:

$$1.4946 \cdot 5 = 7.47$$

כלומר 8 ביטים.

הסבר - סעיף א':

פשוט לחשב את האנטרופיה: $H(P) = - \sum_{i=1}^n p_i \cdot \log_2 p_i$ ואז להכפיל באורך המחרוזת הנתונה.

פתרון - סעיף ב':

למדנו בהרצאה שגודל האינטרוול האחרון יהיה זהה למכפלת ההסתברויות ולכן:

$$r = (0.8)^4 \cdot (0.2)^1 = 0.08192$$

ומכאן:

$$\lceil \log_2\left(\frac{1}{r}\right) \rceil = \lceil \log_2\left(\frac{1}{0.08192}\right) \rceil = 3.60$$

כלומר 4 ביטים.

הסבר - סעיף ב':

ראינו בהרצאה שצריך $\lceil \log_2\left(\frac{1}{r}\right) \rceil$ ביטים כדי לייצג את r כאשר מוגדר: $r = \prod_{i=1}^k p(m_i)$.

פתרון - סעיף ג':

חשב ערך סופי היכול לשמש לקידוד ההודעה ע"פ קידוד אריתמטי.
נחשב ערך סופי לקידוד ההודעה בשיטה הבאה כפי שהוצגה בהרצאה:

	low	high	range
	0	1	1
a	0	0.8	0.8
a	0	$0.8 \cdot 0.8 = 0.64$	0.64
a	0	$0.64 \cdot 0.8 = 0.512$	0.512
a	0	$0.512 \cdot 0.8 = 0.4096$	0.4096
b	$0.4096 \cdot 0.8 = 0.32768$	$1 \cdot 0.4096 = 0.4096$	$0.4096 \cdot 0.2 = 0.08192$

לכן נבחר את המספר 0.4 שישמש כערך לקידוד ההודעה.

הסבר - סעיף ג':

בחרנו את 0.4 כיוון שהוא נמצא בין *high* לבין *low*.

שאלה 4

א. נתונה המחרוזת הבאה של סיביות (משמאל לימין):

011101111010

האם מחרוזת זו יכולה לייצג קידוד של מספרים בעזרת C_γ – Elias Code.

אם כן, פענח את המחרוזת, אחרת, הסבר/י מדוע זה לא יתכן.

ב. הוכח או הפרך באמצעות דוגמה נגדית:

כל מספר מקודד בקוד C_γ במספר שווה או גדול יותר של סיביות מאשר בקוד C_δ .

פתרון סעיף א':

בעזרת C_γ נפצל לחלק בינארי וחלק אונרי.

נשים לב שקוד אונרי תמיד נגמר ב-0, לכן נשבור כל פעם שנראה 0 וזה מספר התווים שנצטרך לקרוא.

0 111011110 10

- המספר **0** ב- C_γ כלומר **קיבלנו 1**.
- המספר האונרי **1110** זה 4 בעשרוני ולכן נקרא $3 = 4 - 1$ התווים הבאים:
יש לנו **111** עם 1 מוביל ולכן $15 = 8 + 4 + 2 + 1 = 2^3 + 2^2 + 2^1 + 2^0$ כלומר **קיבלנו 15**.
- המספר האונרי **110** זה 3 בעשרוני ולכן נקרא $2 = 3 - 1$ התווים הבאים:
יש לנו **10** עם 1 מוביל ולכן: $6 = 4 + 2 = 2^2 + 2^1 + 0$ כלומר **קיבלנו 6**.

לכן הפענוח הוא 1, 15, 6.

פתרון סעיף ב':

נפריך.

למשל עבור המספר 2:

ב- C_γ נייצג בתור 100.

ב- C_δ נייצג בתור 1000.

כלומר, ל-2 בייצוג של C_δ יש יותר סיביות מאשר בייצוג של C_γ .

הסבר סעיף ב':

אמרנו בהרצאה שעבור מספרים גדולים נעדיף את C_δ , אז ניקח דוגמה של מספר קטן.

יש לנו מספרים ש- C_δ גדול יותר מ- C_γ .

שאלה 5

נניח כי המילון בן משתמשים באלגוריתם LZW מאותחל ע"פ הטבלה הבאה:

מספר כניסה	מחרוזת
0	"A"
1	"B"
2	"C"
...	...

א. פרוש את ההודעה הבאה (משמאל לימין):

0, 1, 3, 5, 6, 2, 7, 9, 2

הראה את המילון בכל שלב ושלב של האלגוריתם.

ב. בהנחה כי מספר הכניסות במילון הראשוני הוא 4 ומוכפל בכל פעם שהוא מתמלא, מה אורכה של ההודעה הדחוסה מבחינת מספר סיביות?

פתרון א'

הפענוח הולך בכיוון ההפוך אבל כדי לא לבלבל במקום לקרוא להם w, k נקרא להם OLD, NEW . נתון לנו 0, 1, 3, 5, 6, 2, 7, 9, 2. אנחנו צריכים גם את S מחרוזת ו- C זה התו הראשון של S . וגם נתון לנו המילון ההתחלתי:

Code	Symbol
0	A
1	B
2	C
...	...

נפעל לפי האלגוריתם (לפי הקוד לפענוח שלמדנו בהרצאה):

OLD	NEW	S	C	Output	כניסה למילון $OLD \cdot C$
0	-	-	-	A	
0	1	B	B	B	AB
1	3	AB	A	AB	BA
3	5	ABA	A	ABA	ABA
5	6	ABAA	A	ABAA	ABAA
6	2	C	C	C	ABAAC
2	7	ABAAC	A	ABAAC	CA
7	9	ABAACA	A	ABAACA	ABAACA
9	2	C	C	C	ABAACAC
2	EOF				

והמילון:

Code	Symbol
0	A
1	B
2	C
3	AB
4	BA
5	ABA
6	ABAA
7	ABAAC
8	CA

9	ABAACA
10	ABAACAC

תרגום ההודעה, לא נדרש, הרווחים הם בשביל הקריאות ולא חלק מההודעה עצמה:
A B AB ABA ABBA C ABAAC ABAACA C

פתרון ב'

בהנחה כי מספר הכניסות במילון הראשוני הוא 4 ומוכפל בכל פעם שהוא מתמלא, מה אורכה של ההודעה הדחוסה מבחינת מספר סיביות?
 נחשב לפי הכניסות בכל שלבי הרחבה (פקטור של 2), כלומר 2, 4, 8, 16, 32 וכו'...
 בכניסות יהיו לנו את ה-OLD שיגדיר לנו את השלב שעשינו.

- **4 כניסות:** בהתחלה המילון הראשוני היה בעל 4 מקומות, כלומר $2^2 = 4$ לכן נדרש כאן 2 סיביות. קודדנו את *AB*, *B*, כלומר 0, 1 לפי ה-OLD.
- **8 כניסות:** ברגע שהכנסנו את *AB* המילון היה מלא והיינו צריכים הגדיל אותו ל- $2 \cdot 4 = 8$, כלומר $2^3 = 8$ ולכן 3 סיביות עבור כל אחד מ- 2, 3, 5, 6.
- **16 כניסות:** לאחר שקודדנו את *C* הוספנו למילון את *ABAAC*, המילון מלא (אפשר לראות לפי החלוקה של הצבעים במילון) ולכן הכפלנו את המילון $2 \cdot 8 = 16$ כלומר $2^4 = 16$ לכן נדרש 4 סיביות עבור כל אחד מ- 2, 7, 9.

אורכה של ההודעה הדחוסה מבחינת מספר סיביות היא: $2 \cdot 2 + 3 \cdot 4 + 4 \cdot 3 = 28 \text{ bits}$

מבחן 2016 - מבחן דוגמה

שאלה 1

נתון א"ב $\Sigma = \{a, b, c, d, e\}$ בהסתברויות $P = \{0.4, 0.2, 0.2, 0.1, 0.1\}$ בהתאמה.

א. תן/י שלשה קודי הפמן קנונים היכולים להתקבל מהסתברויות אלו.

ב. חשבו/י את אורך הקוד הממוצע של כל קוד.

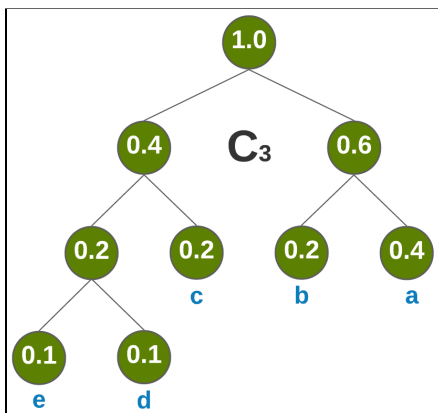
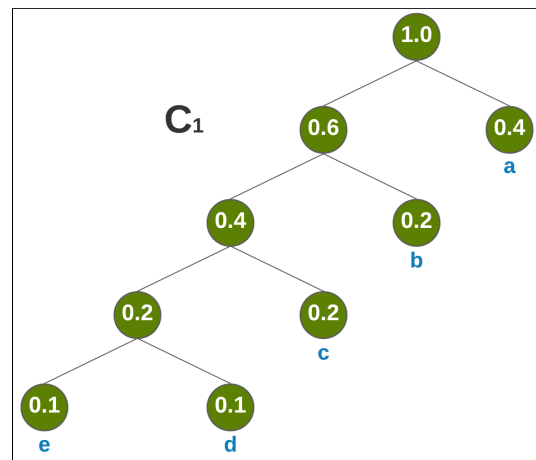
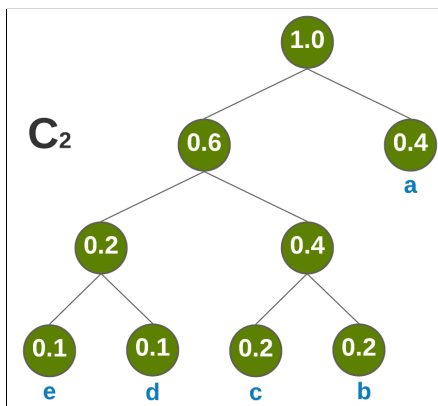
ג. חשבו/י את האנטרופיה.

ד. איזה קוד תעדיף/י לבחור מבין האופציות שקיבלת?

לנוחות אני אסדר:

$$\Pr(a) = 0.4, \quad \Pr(b) = 0.2, \quad \Pr(c) = 0.2, \quad \Pr(d) = 0.1, \quad \Pr(e) = 0.1$$

פתרון סעיף א'



פתרון סעיף ב'

נחשב לפי הנוסחה: $E(C, P) = \sum_{i=1}^n p_i \cdot |c_i|$

$$E(C_1, P) = 0.4 \cdot 1 + 0.2 \cdot (2 + 3) + 0.1 \cdot (4 + 4) = 2.2$$

$$E(C_2, P) = 0.4 \cdot 1 + 0.2 \cdot (3 + 3) + 0.1 \cdot (3 + 3) = 2.2$$

$$E(C_3, P) = 0.4 \cdot 2 + 0.2 \cdot (2 + 2) + 0.1 \cdot (3 + 3) = 2.2$$

אין צורך לחשב את שלושתם כי הפמן הוא גם ככה מינימלי ולכן שלושתם תמיד יקבלו אותו תוצאה.
זה יכול לחסוך זמן במבחן וגם מקנה נקודות על הבנה.

פתרון סעיף ג'

נחשב לפי הנוסחה: $H(P) = - \sum_{i=1}^n p_i \cdot \log_2(p_i)$

$$H(P) = - \left(0.4 \cdot \log_2(0.4) + 2 \cdot \left[0.2 \cdot \log_2(0.2) \right] + 2 \cdot \left[0.1 \cdot \log_2(0.1) \right] \right)$$

$$H(P) = 2.121928$$

אפשר לוודא ש: $2.12 < 2.22$ כדי לבדוק תקינות (כי האנטרופיה היא האופטימלית ביותר).
לא ייתכן שהאורך הממוצע יהיה קטן יותר מהחסם התחתון שהוא 2.12 כי אי אפשר לנצח את האנטרופיה.

פתרון סעיף ד'

צריך לחשוב על ה-prelude.

נרצה מינימום בלוקים (מספר מילות-קוד באורכים שונים - מינימלי).

לכן, נבחר או את (2) או את (3) משום שיש להם 2 בלוקים (2 רמות בעץ).

הערה: אמרנו שבהפמן קנוני נשלח ב-prelude את המילות-קוד הראשונות בכל בלוק ומה הגודל של אותן מילות-קוד באותו בלוק.

שאלה 2

נתון א"ב $\Sigma = \{a, b, c, d, e, f\}$ בהסתברויות $P = \{0.2, 0.3, 0.1, 0.2, 0.1, 0.1\}$ בהתאמה.
א. מהי ההסתברות למופע של המחרוזת baccf?
ב. השתמש/י בקוד אריתמטי על מנת לקודד את המחרוזת baccf.
ג. האם קיים קשר בין סעיף א' לסעיף ב'? הסבר/י!
ד. מהם היתרונות והחסרונות של קוד אריתמטי בהשוואה לקוד הפמן?

לנוחות אני אסדר:

$$\Pr(a) = 0.2, \Pr(b) = 0.3, \Pr(c) = 0.1, \Pr(d) = 0.2, \Pr(e) = 0.1, \Pr(f) = 0.1$$

פתרון סעיף א'

ההסתברות למופע זה פשוט מכפלת ההסתברויות (נכפיל עם כל תו מהמחרוזת הנתונה):

$$\begin{aligned} \Pr(b) \cdot \Pr(a) \cdot \Pr(c) \cdot \Pr(c) \cdot \Pr(f) &= \\ &= 0.3 \cdot 0.2 \cdot 0.1 \cdot 0.1 \cdot 0.1 = \\ &= \frac{3}{50000} = 0.00006 \end{aligned}$$

פתרון סעיף ב'

נעזר באלגוריתם לקידוד אריתמטי שלמדנו בהרצאה:

Encoding Arithmetic Code Algorithm

```

1: low ← 0.0
2: high ← 1.0
3: while input symbols remain do:
4:     range ← high - low
5:     get symbol
6:     high ← low + high_bound(symbol)*range
7:     low ← low + low_bound(symbol)*range
8: end-while
9: output any value in [low, high)
```

ונכין טבלת חישוב בהתאם כפי שנלמד בהרצאה:

תו	low	high	range
	0	1	1
b	$0 + (0.2) \cdot 1 = 0.2$	$0 + (0.5) \cdot 1 = 0.5$	$0.5 - 0.2 = 0.3$
a	$0.2 + (0) \cdot 0.3 = 0.2$	$0.2 + (0.2) \cdot 0.3 = 0.26$	$0.26 - 0.2 = 0.06$
c	$0.2 + (0.5) \cdot 0.06 = 0.23$	$0.2 + (0.6) \cdot 0.06 = 0.236$	$0.236 - 0.23 = 0.006$
c	$0.23 + (0.5) \cdot 0.006 = 0.233$	$0.23 + (0.6) \cdot 0.006 = 0.2336$	$0.2336 - 0.23 = 0.0006$
f	$0.233 + (0.9) \cdot 0.0006 = 0.23354$	$0.233 + (1) \cdot 0.0006 = 0.2336$	$0.2336 - 0.23354 = 0.00006$

פתרון סעיף ג'

כן, קיים קשר. למדנו בהרצאה שגודל האינטרוול האחרון הוא בעצם מכפלת כל ההסתברויות שלנו. אנחנו צריכים להעביר בסופו של דבר את המספר בתת-הקטע האחרון, ככל שתת-הקטע יותר קטן, אנחנו נצטרך יותר סיביות כדי לייצג את המספר הזה ככל שההסתברות יותר נמוכה.

פתרון סעיף ד'

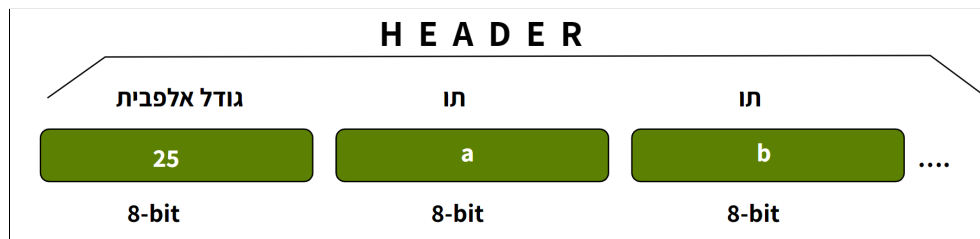
קוד אריתמטי	קוד האפמן
מחליפים את ההודעה כולה במספר יחיד של floating-point בטווח מסוים (הרבה סיביות אחרי הנקודה).	כל תו בא"ב מקבל מילת קוד
לא צריכים התפלגויות (חוץ מאריתמטי סטטי).	הוא צריך התפלגויות (חוץ מדינאמי שלומד בתהליך)
קוד אדפטיבי קל	כבד לשינוי (לעדכן התפלגות)
לא צריך לשלוח את המודל בשביל הפענוח כי המפענח מתחזק את כל המודל בעצמו	טבלה או עץ שיגיד את המיפוי בין מילות הקוד לא"ב
משתמשים בשברי סיביות בשביל מילת קוד אחת.	יש לנו הגבלה - משתמשים במספר שלם של ביטים כדי לקודד כל מילת קוד
בקוד אריתמטי נשפר יותר טוב מהאפמן.	הוכחנו שהאפמן אופטימלי אבל ההבדל בינו לבין האנטרופיה נמדדת בסיביות אחת לכל אות וזה יכול להצטבר כלומר בטקסט בגודל n יצבר לנו כמות של n סיביות וזה פחות טוב.

שאלה 3

נתון א"ב $\Sigma = \{a, b, c\}$ ונתונה המחרוזת הבאה: *ababbac*.
 נניח כי אנו משתמשים במודל סטטי למחצה בו מניחים התפלגות אחידה של התווים.
א. חשבו/י את מספר הביטים הדרוש לתיאור המודל (prelude) במודל סטטי למחצה (Semi Static) ובמודל סטטי למחצה עם הסתברויות עצמיות (semi-static self probabilities).
 הנח/י שמספר הסיביות (bits) הדרושות לקידוד הסתברות אחת הוא 4.
ב. השתמש/י בקוד $Elias C_\gamma$ לקידוד המחרוזת.
ג. חשבו/י את האורך הממוצע של מילת קוד.

פתרון סעיף א' - מודל סטטי למחצה

לפי הנלמד בכיתה, צריך להעביר ב-Header את:



1. גודל הא"ב שלנו, במקרה הזה הגודל הוא 3 ונציג אותו ב-8 סיביות.

2. כל תו מיוצג בקוד ASCII בן 8 סיביות כלומר $|\Sigma| \cdot 8$.

לכן, צריך להעביר:

$$8 + 3 \cdot 8 = 32$$

פתרון סעיף א' - מודל סטטי למחצה עם הסתברויות עצמיות

לפי הנלמד בכיתה, צריך להעביר ב-Header את:

1. גודל הא"ב שלנו, במקרה הזה הגודל הוא 3 ונציג אותו ב-8 סיביות.

2. כל תו מיוצג בקוד ASCII בן 8 סיביות כלומר $|\Sigma| \cdot 8$.

3. ההסתברות של כל תו כלומר $|\Sigma| \cdot p_i$ (נתון לנו שהסתברות היא 4).

לכן, צריך להעביר:

$$8 + 3 \cdot 8 + 3 \cdot 4 = 44$$

פתרון ב'

צריך לבנות 3 מילות-קוד ב- c_y .

לכל אות נבדוק:

1. ייצוג בינארי של אורך מספר הסיביות של כל מספר, למשל המספר 3 מיוצג ב-'11' והאורך שלו הוא 2.
2. מה הייצוג שלו בבינארי ונשמיט את הרישא שלו (בייצוג הבינארי).
3. נשרשר את שלב (1) עם שלב (2) וזה יהיה מילת הקוד עבור האות.

נפעל בטבלת חישוב - עבור כל עמודה נעבור משמאל לימין:

c_y	השמטה אחרי בינארי	אורך בינארי	ייצוג בינארי		s_i
$0 \pm 0 = 0$	$\pm$	0	1	1	a
$10 \pm 0 = 100$	± 0	$(10)_2 = (2)_2 = 10$	10	2	b
$10 \pm 1 = 101$	± 1	$(11)_2 = (2)_2 = 10$	11	3	c

נקודד את המחרוזת $ababbac$:

0 100 0 100 100 0 101

פתרון ג'

האורך הממוצע של מילת הקוד:

- האורך של a הוא 1 ומספר המופעים שלו במחרוזת הוא 3.
- האורך של b הוא 3 ומספר המופעים שלו במחרוזת הוא 3.
- האורך של c הוא 3 ומספר המופעים שלו במחרוזת הוא 1.
- האורך של המחרוזת הוא 7.
- לפי מודל סטטי למחצה עם הסתברויות עצמאיות ראינו בסעיף א' שצריך להעביר 44 סיביות.

לכן בסה"כ נקבל:

$$\frac{1 \cdot 3 + 3 \cdot 3 + 3 \cdot 1 + 44}{7} = 8.42$$

שאלה 4

נתונה המחזורת הבאה שאורכה $2n$:

abababa ... ab

- א. צייר/י את ה trie המתקבל מ-LZW
- ב. חשבו/י את יחס הדחיסה האסימפטוטי - כלומר את היחס בין גודל הקובץ הדחוס והקובץ המקורי כאשר n שואף ל- אינסוף.
- ג. השווה/י את התוצאה שקיבלת עם הדחיסה המתקבלת ע"י הפמן ועם האנטרופיה.
- ד. הצג/י את הקובץ הדחוס שנוצר באמצעות אלגוריתם LZSS.

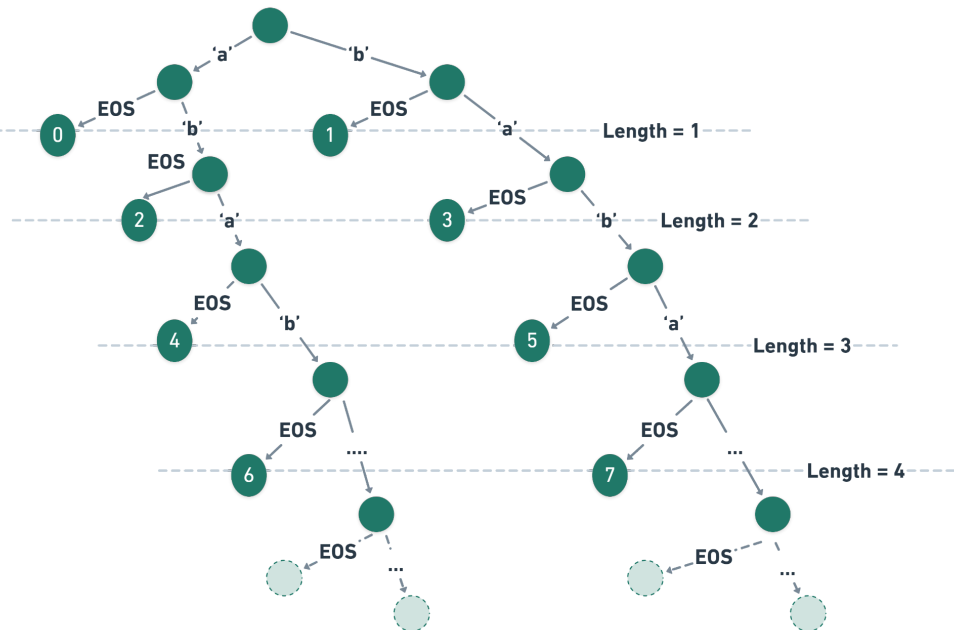
פתרון א'

נתון לנו: $\Sigma = \{a, b\}$, $|\Sigma| = 2$, $S = abababa \dots ab$, $|S| = 2n$
 באלגוריתם LZW אנחנו מאתחלים את המילון עם הא"ב ואז ממלאים לפי האלגוריתם שנלמד.

נשים לב שלאחר מספר הכנסות למילון קיבלנו חוקיות מסוימת:

לכל אורך יש 2 מחזורות, כלומר לאורך 1 יש את המחזורות a, b ולאורך 2 יש את ab, ba ..
 החוקיות הזאת ממשיכה ככה הלאה לכל אורך ...

Code	Phrase
0	a
1	b
2	ab
3	ba
4	aba
5	bab
6	abab
7	baba
.....	
.....	
....	
.....	



פתרון ב'

(מסוג השאלות שככל הנראה לא יופיעו בבחינה).

נסמן מספר האינדקסים במילון ב- k .

ראינו בחוקיות בסעיף א' שלכל אורך יש 2 מילים.

המטרה היא לכתוב את k כפונקציה של n .

$$\sum_{i=1}^k 2i = 2n \quad /: 2$$

$$\sum_{i=1}^k i = n$$

כעת, לפי סכום של סדרה חשבונית נקבל: $\frac{(1+k) \cdot k}{2} = n$

לכן כאשר $n \rightarrow \infty$ אז $\sqrt{n} \rightarrow k$ כלומר היחס בין גודל הדחיסה לגודל המקורי הוא: $\frac{\sqrt{n}}{2n} = \frac{1}{\sqrt{n}}$

פתרון ג'

במחרוזת יש n פעמים a ו- n פעמים b ולכן הפמן יתן אורך 1 לכל אות.

גם האנטרופיה תהיה 1 $H(P) = 2 \cdot 0.5 \cdot \log_2(0.5) = 1$ כי יש 2^k אותיות לקודד.

פתרון ד'

- הערה: כל פעם שמתבקש ממנו לדחוס משהו, צריך לעשות זאת הכי טוב שיש אלא אם נאמר אחרת במפורש.

אם מדברים על LZSS אז הדרישה היא לבדוק אם אפשר להשתמש בהצבעות עצמיות.

בלי הצבעה עצמית: $ab(2, 2)(4, 4)(8, 8) \dots (n, n)$

עם הצבעה עצמית: $LZSS(T) = ab(2, 2n - 2)$

כלומר, נלך כל פעם 2 תווים אחורה ואז נעתיק משם $2n - 2$ תווים.

שאלה 5

נתונה המחרוזת הבאה $bcabbaaca$

- א. השתמש/י בקוד $Shannon - Fano$ לחישוב אורך הממוצע של מילות הקוד כאשר נתון לנו הא"ב $\Sigma = \{a, b, c\}$.
- ב. השתמש/י בקוד $Shannon - Fano$ לחישוב אורך הממוצע של מילות הקוד כאשר נתון הא"ב של כל זוגות האותיות האפשריים.
- ג. חשב/י את האנטרופיה ($Entropy$) עבור כל א"ב.
- ד. איזה קוד תעדיף/י לבחור מבין האופציות שקיבלת?

פתרון א':

נציג תחילה את השכיחויות: $a: \frac{4}{9}$, $b: \frac{3}{9}$, $c: \frac{2}{9}$, נמיינ את ההסתברויות בהתאם ונצייר את העץ:

מילת קוד	תו
0	a
10	b
11	c

נחשב את אורך מילת הקוד הממוצעת:

$$E(C, P) = \sum_{i=1}^{|\Sigma|} p_i \cdot |c_i| = \frac{4}{9} \cdot 1 + \frac{3}{9} \cdot 2 + \frac{2}{9} \cdot 2 = \frac{14}{9} \approx 1.55$$

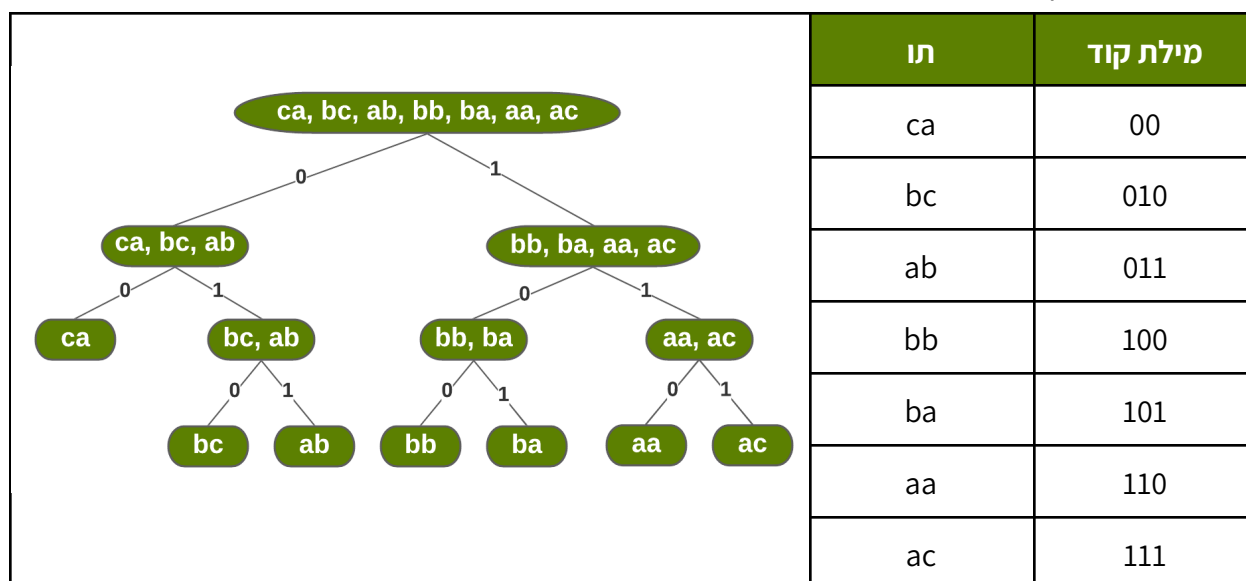
פתרון ב':

תחילה צריך לסדר את כל הזוגות והשכיחויות שלהן:

שכיחות	זוג
1	bc
2	ca
1	ab

bb	1
ba	1
aa	1
ac	1

כעת נצייר את העץ לפי סדרת ההסתברות הממוינת:



נחשב את אורך מילת הקוד הממוצעת: $E(C, P) = 6 \cdot \frac{1}{8} \cdot 3 + 2 \cdot \frac{1}{8} \cdot 2 = \frac{22}{8} = 2.75$

פתרון ג':

$$H(P) = -\frac{4}{9} \cdot \log_2\left(\frac{4}{9}\right) - \frac{3}{9} \cdot \log_2\left(\frac{3}{9}\right) - \frac{2}{9} \cdot \log_2\left(\frac{2}{9}\right) \approx 1.530$$

למדנו מההרצאה ש: $H(p(s_i, s_j)) = -\sum_{(i,j)} p(s_i, s_j) \cdot \log_2 p(s_i, s_j) = 2H(P(s))$

לכן: $H(p(s_i, s_j)) = 2H(P(s)) = 2 \cdot 1.530 \approx 3.06$

פתרון ד':

אנחנו נעדיף את סעיף א' (תווים בודדים) כי אורך המילה הממוצעת שלו הרבה יותר קטן מאשר זוגות של תווים.

שאלה 5

נתון האלגוריתם הבא לקידוד:

1. הפעל את אלגוריתם הפמן ה- $ary-8$ על איברי הטקסט
2. דחס את האיברים באמצעות התווים $0, 1, 2, 3, 4, 5, 6, 7$
3. החלף כל אחד מהתווים לעיל במחרוזת בינארית ייחודית באורך 3.

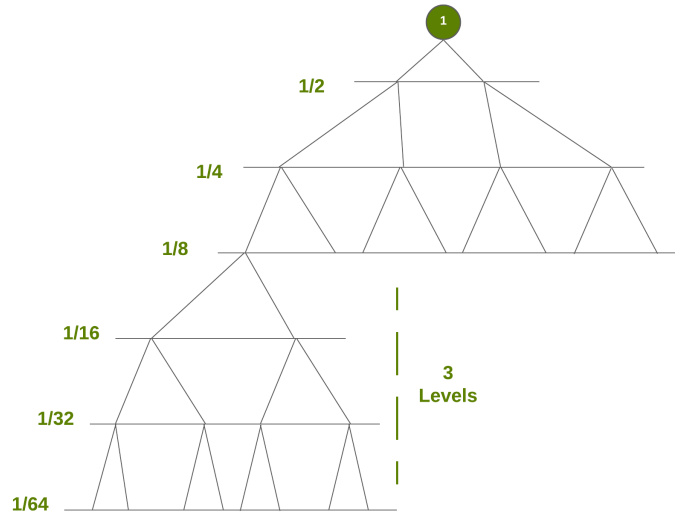
א. מתי האלגוריתם לעיל אופטימלי?

ב. תן דוגמה להסתברויות על א"ב בן 15 תווים שעבורה האלגוריתם לעיל נותן קוד אופטימלי.

ג. תן דוגמה להסתברויות על א"ב בן 15 תווים שעבורה האלגוריתם לעיל אינו נותן קוד אופטימלי.

פתרון א'

אם נצייר עץ של 8-ארי אז אפשר לראות שיש 7 מילים ברמה הראשונה ו-8 ברמה השניה, אנחנו צריכים ליצור מצב שהפרש בין עץ בינארי לעץ 8-ארי הוא פי 3 מההפרשים אפשר לראות שיש 3 רמות הפרשים בעץ בינארי רגיל ורק אז זה יהיה אופטימלי. נציג כאן עץ בייצוג בינארי להשוואה עם העלים האפשריים עבור 8-ארי:



פתרון ב'

נתן דוגמה כאשר הא"ב מכיל 7 תווים בעלי ההסתברות $\frac{1}{8}$ ו-8 תווים בעלי ההסתברות $\frac{1}{64}$, למשל:

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
$\frac{1}{64}$	$\frac{1}{64}$	$\frac{1}{64}$	$\frac{1}{64}$	$\frac{1}{64}$	$\frac{1}{64}$	$\frac{1}{64}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$

ואז ברמות הגבוהות יהיה ההסתברות גבוהה יותר מאשר ברמות הנמוכות.

פתרון ג' האלגוריתם לא יתן קוד אופטימלי אם נפעל בהסתברות אחידה, גם אם נפעל על חלוקה יחסית שווה

של ההסתברויות בין המילים וגם אם נפעל בחלוקה קיצונית של ההסתברויות למשל 14 באורך 4 ו-1 באורך 3.

שאלה 3

נתון הא"ב $\Sigma = \{a, b, c, d, e, f\}$ בהסתברויות $P = \{0.15, 0.04, 0.26, 0.05, 0.1, 0.4\}$ בהתאמה.

- א. חשב את האנטרופיה.
- ב. מהו סדר קוד גולומב (Golomb Code) אופטימלי עבור ההסתברויות לעיל?
- ג. חשב את אורך הקוד הממוצע של הקוד לעיל.
- ד. מהי היתירות של קוד זה? (*redundancy*).

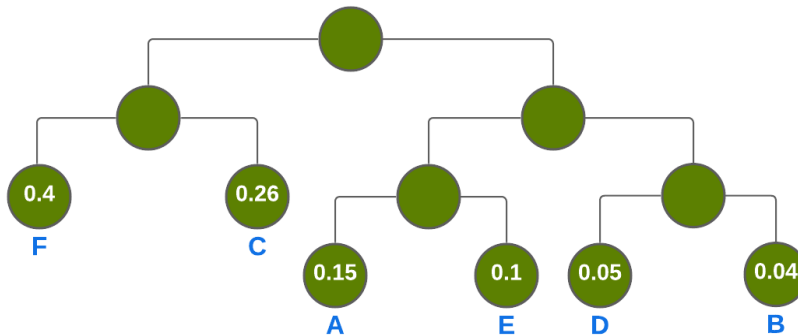
פתרון א'

$$\begin{aligned}
 H(P) &= - \sum_{i=1}^6 p_i \cdot \log_2(p_i) = \\
 &= - (0.15 \cdot \log_2(0.15) + 0.04 \cdot \log_2(0.04) + 0.26 \cdot \log_2(0.26) + \\
 &\quad + 0.05 \cdot \log_2(0.05) + 0.1 \cdot \log_2(0.1) + 0.4 \cdot \log_2(0.4)) = 2.178
 \end{aligned}$$

פתרון ב'

קידוד גולומב הוא אלגוריתם לדחיסת נתונים, זה קוד עם בלוקים באורך קבוע - Fixed size. במקום לדבר על בלוקים בגודל שעולה אקספוננציאלי כמו שהיה לנו ב- C_γ ו- C_δ אנחנו נדבר על מקרה בו כל הבלוקים הם באותו הגודל. למעשה יהיה לנו פרמטר לקוד הזה שנקרא לו b והוא יהיה הגודל של כל בלוק. גודל הא"ב הוא b וגם $0 < r \leq b$. במקרה שלנו $6 = |\Sigma| = b$. ה- b זה כל מילות הקוד שיש להם אותו פתיח. אנחנו רוצים לבנות מילות קוד שהם בערך באותו האורך:

- אורך מילות הקוד הארוכות: $\lceil \log_2(b) \rceil = \lceil \log_2(6) \rceil = 3$
 - מספר מילות הקוד הארוכות הם: $2^{\lceil \log_2(6) \rceil} - 2^{\lfloor \log_2(6) \rfloor} = 2^3 - 2^2 = 8 - 4 = 4$
 - אורך מילות הקוד הקצרות: $\lfloor \log_2(b) \rfloor = \lfloor \log_2(6) \rfloor = 2$
 - מספר מילות הקוד הקצרות הם: $2^{\lfloor \log_2(6) \rfloor} - n = 2^2 - 6 = 4 - 6 = -2$
- נצייר את העץ וכך נציג את סדר גולומב אופטימלי:



פתרון ג'

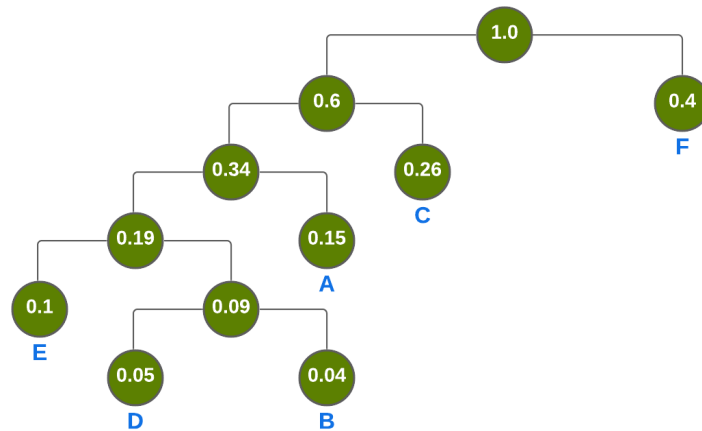
לפי סעיף ב' קיבלנו: $C = \{A: 100, B: 111, C: 01, D: 110, E: 101, F: 00\}$
 כאשר נתונות לנו ההסתברויות הבאות: $P = \{0.15, 0.04, 0.26, 0.05, 0.1, 0.4\}$ בהתאמה.
 לכן, נחשב:

$$E(C, P) = \sum_{i=1}^n p_i \cdot |c_i| = 3 \cdot (0.15 + 0.04 + 0.05 + 0.1) + 2 \cdot (0.26 + 0.4) =$$

$$= 1.02 + 1.32 = 2.34$$

פתרון ד'

שואלים אותנו מהי היתירות של קוד זה? אפשר לדעת את היתירות של הקוד לפי קוד הפמן או קוד אריתמטי.
 הקוד שלנו מסעיף ב' הוא C , והקוד הפמן C' יהיה:



ולכן, $E(C', P) = 2.22$ ולפי יתירות מינימלית:

$$E(C', P) \leq E(C, P)$$

$$2.22 \leq 2.34$$

שאלה 3נתונה המחזורית הבאה: *ababbac*

נניח כי אנו משתמשים במודל מסדר ראשון - כלומר הא"ב הוא זוגות של אותיות וההסתברות של כל תו היא ביחס להסתברות הופעתו לאחר התו הקודם לו.

א. חשב את מספר הביטים הדרוש לתיאור המודל (*prelude*) במודל הסטטי למחצה המתואר לעיל. הנח שמספר הסיביות הדרושות לקידוד הסתברות אחת הוא 5.

ב. השתמש בקוד $Elias C_\gamma$ לקידוד המחזורית.

ג. חשב את האורך הממוצע של מילת קוד.

ד. מהי היתירות של קוד זה? (*redundancy*).

פתרון א'

יש לי פה *ababbac* ונתון לנו מודל מסדר ראשון כאשר הא"ב C הוא זוגות של אותיות כלומר:

$$\Sigma = \{aa, ab, ab, ba, bb, bc, ca, cb, cc\}$$

ההסתברויות שלנו הם:

ab	ba	bb	ac
$\frac{2}{6}$	$\frac{2}{6}$	$\frac{1}{6}$	$\frac{1}{6}$

- לכל אחד מהתווים בא"ב שלנו - 3 כאלה, כל תו בא"ב מייצג לנו קוד ASCII בן 8 סיביות.
- 8 סיביות עבור גודל הא"ב.
- ההסתברויות הם $5 \cdot 3^2 = 45$ סיביות
- סה"כ נקבל $77 = 45 + 24 + 8$ סיביות.

הערה: אני לא בטוח לגבי הפתרון הזה במיוחד כי מדובר בא"ב של זוגות ולא היה לזה פה התייחסות מלבד ההסתברויות.

פתרון ב'

נגדיר $x = 92$.

$$w = \text{Binary}(92) = 1011100$$

$$\text{out1} = \text{Unary}(|w|) = \text{Unary}(7) = 1111110$$

$$\text{out2} = w - (\text{first} - 1) = 011100$$

$$\text{return out1} \cdot \text{out2} = 1111110 011100$$

מבחן 2016 - סמסטר א' - מועד א'

שאלה 1

א. נתונות מילות הקוד הבאות:

$$a = 101, b = 1101, c = 1011, d = 1100$$

האם קוד זה ניתן לפענוח חד ערכי (UD)?

ב. נתון א"ב בגודל $x + 3$. יש לבנות קוד $C = \{c_1, c_2, c_3, c_4, \dots, c_{x+3}\}$ הניתן לפענוח בצורה

UD, כך ששלושת מילות הקוד הראשונות c_1, c_2, c_3 מכילות 3 סיביות ושאר מילות הקוד $c_4, \dots, c_{x+3}$ מכילות 8 סיביות.

חשב מהו גודל הא"ב המקסימלי כך שמצב זה אפשרי.

פתרון א'

נזכר באלגוריתם:

עוברים על כל זוגות מילות קוד ובודקים האם אחת היא הרישא של האחרת.

כל עוד a היא רישא של b , אנחנו מוסיפים את ה-dangling עד אשר מתקיים אחד מהתנאים הבאים:

1. או שה-dangling suffix הוא מילה מקורית - זה אומר שהוא לא UD.

2. או שלא - זה אומר שהוא כן UD.

איטרציה 1

נתון רשימה $L = \{101, 1101, 1011, 1100\}$. עוברים על כל זוגות מילות קוד ובודקים האם אחת היא

הרישא של האחרת. ה-dangling בין $a = 101$ ל- $c = 1011$ הוא 1 לכן נוסיף אותו לרשימה.

כעת הרשימה תראה כך: $L = \{101, 1101, 1011, 1100, 1\}$.

איטרציה 2

עוברים על כל זוגות מילות קוד ב- L ובודקים האם אחת היא הרישא של האחרת.

כעת נשים לב שעבור '1' יחד עם '1101' נקבל dangling של '101' שהיא מילה מקורית a ולכן לא UD.

פתרון ב'

לפי אי שוויון קראפט $K(C) \leq 1$ אז C הוא קוד UD, לכן המקרה המקסימלי הוא כאשר $K(C) = 1$.

$$3 \cdot 2^{-3} + x \cdot 2^{-8} = 1$$

$$x \cdot 2^{-8} = 1 - 3 \cdot 2^{-3} = 5 \cdot 2^{-3} \quad \setminus : 2^{-8}$$

$$x = 5 \cdot 2^5 = 160$$

מאחר ונתון לנו שהא"ב בגודל $x + 3$ אז גודל הא"ב המקסימלי יהיה $160 + 3 = 163$.

שאלה 2

כזכור, קוד נקרא **פרפיקסי**, אם אף מילת קוד אינה רישא של מילת קוד אחרת.
 באופן סימטרי הגדרנו קוד **סופיקסי**, בו אף מילת קוד אינה סיפא של מילת קוד אחרת.
 קוד נקרא קוד **אפיקסי** אם הקוד הוא גם פרפיקסי וגם סופיקסי.
 להלן מילות קוד שהן חלק מקוד אפיקסי:

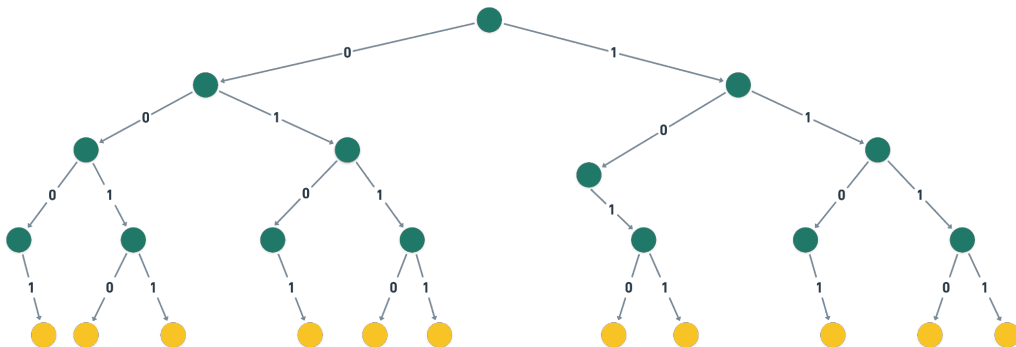
0010, 1010, 0110, 1110, 0001, 0101, 1101, 0011, 1011, 0111, 1111

- א.** השלם את הקוד באמצעות מילת קוד אחת באורך 3, אחת באורך 4 ו-4 מילות קוד באורך 5.
ב. הסבר מדוע קוד אפיקסי אינו ניתן לסנכרון.

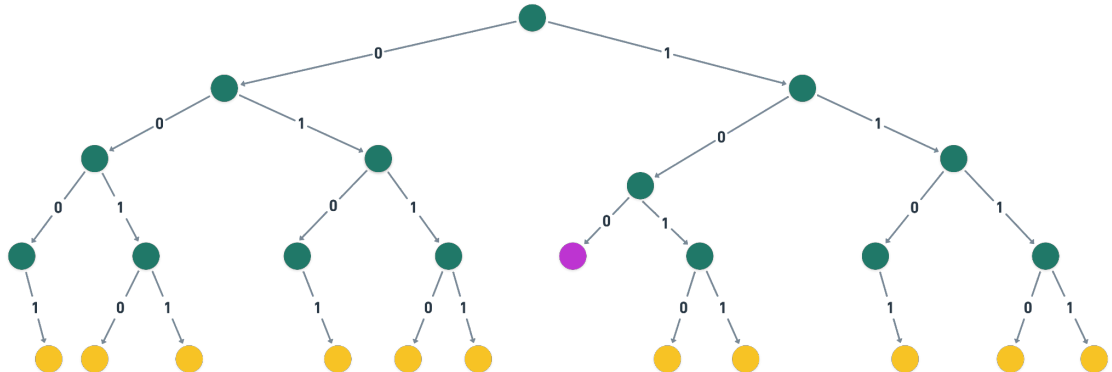
סעיף א'

הדרך הכי טובה ונוחה כדי לדעת איזה מילים להוסיף היא בעזרת ציור עץ של המילות הקוד הקיימות.
 ואחרי ציור העץ אפשר לראות יותר בקלות איפה ליצור ענפים חדשים שיהיו המילות הנוספות כנדרש.

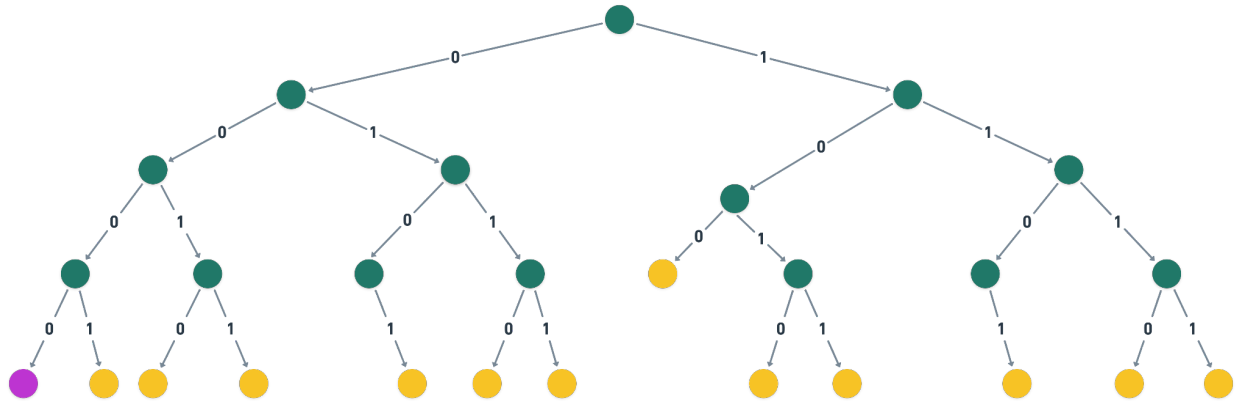
נצייר:



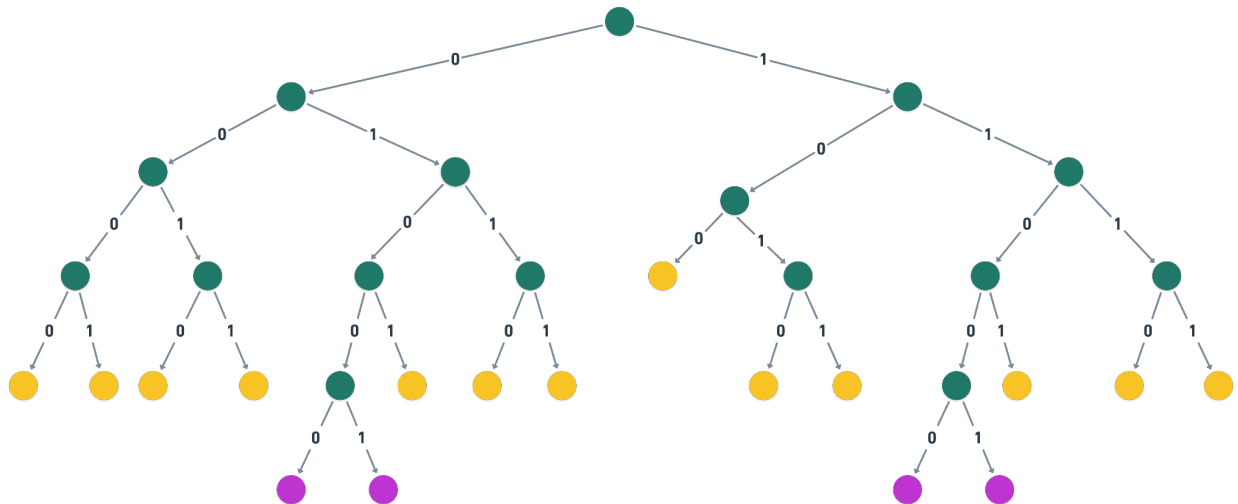
עבור הכנסה של מילה באורך 3 צריך לחפש מסלול באורך 2 שניתן להאריך אותו ב-1, לכן נציב (בסגול):



עבור הכנסה של מילה באורך 4 צריך לחפש מסלול באורך 3 שניתן להאריך אותו ב-1, לכן נציב (בסגול):
נשים לב שיש לנו 3 מסלולים פנויים: אם נבחר את 1100 או את 0100 אז נשים לב שהצומת הקודמת שהוספנו
(100) תהיה הסיפא ולכן נפסול אותם. האופציה החוקית היא לבחור את המסלול הפנוי 0000:



עבור הכנסה של 4 מילים באורך 5 נציב (בסגול):



סעיף ב'

קוד אפיקסי - לא ניתן לסנכרון.

בקוד אפיקסי אף מילת קוד אינה סיפא של מילה אחרת ולכן במקרה של איבוד ביטים הקריאה לא תסתנכרן.

הסבר:

כדי שקוד יסתנכרן לאחר איבוד של ביטים צריך לקרוא אותו כאלו לא איבדנו ביטים, עד שנגיע למילת קוד
שהיא סיפא של מילת הקוד המקורית שהיתה מקודדת (לפני שאיבדנו ביטים).

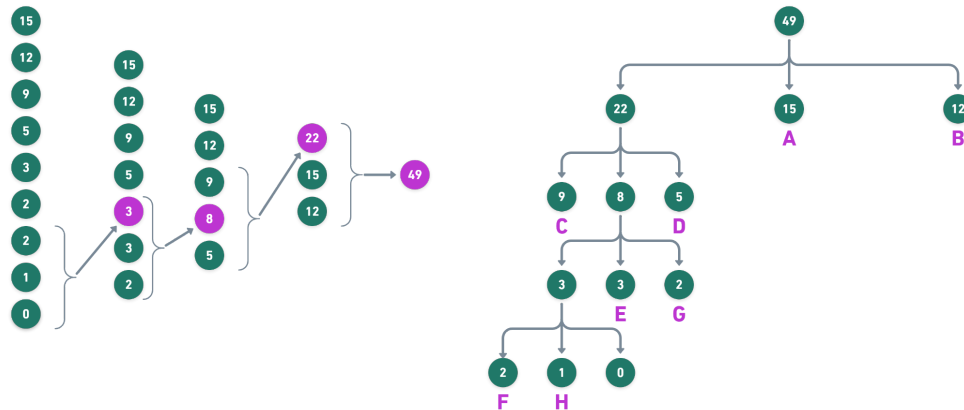
מרגע זה אנחנו מסונכרנים וכל מילות הקוד שנקרא יהיו מתאימות להודעה המקורית (ללא איבוד).

שאלה 3

נתונות השכיחויות 15,12,9,5,3,2,2,1 עבור האותיות A,B,C,D,E,F,G,H בטקסט מסוים. ברצוננו לבנות עץ Huffman טרינרי (על בסיס 3) ולהשוותו לעץ Huffman בינארי. **א.** בנה/י עץ Huffman טרינרי אופטימלי עבור תדירויות אלה. **ב.** בנה/י מילות קוד טרינריות עבור האותיות, ואח"כ צור/י את מילות הקוד הבינאריות על ידי החלפת שלושת התווים הטרינריים ע"י 0, 10 ו-11 בהתאמה. דאג/י לבצע את ההחלפה בצורה הטובה ביותר מבחינת הדחיסה. **ג.** בנה/י עץ Huffman בינרי וחשב/י את גדלי הטקסטים הדחוסים בשני השיטות. איזו שיטה עדיפה?

סעיף א'

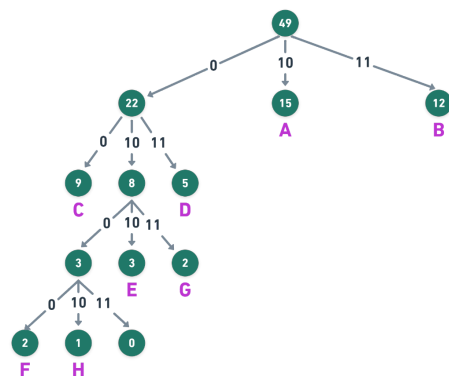
דבר ראשון נשים לב כי צריך לבנות עץ אופטימלי. מכיוון שמספר השכיחויות זוגי צריך להוסיף צומת פיקטיבית (בשכיחות של 0) ואז לפעול לפי $ary - 3$.



סעיף ב'

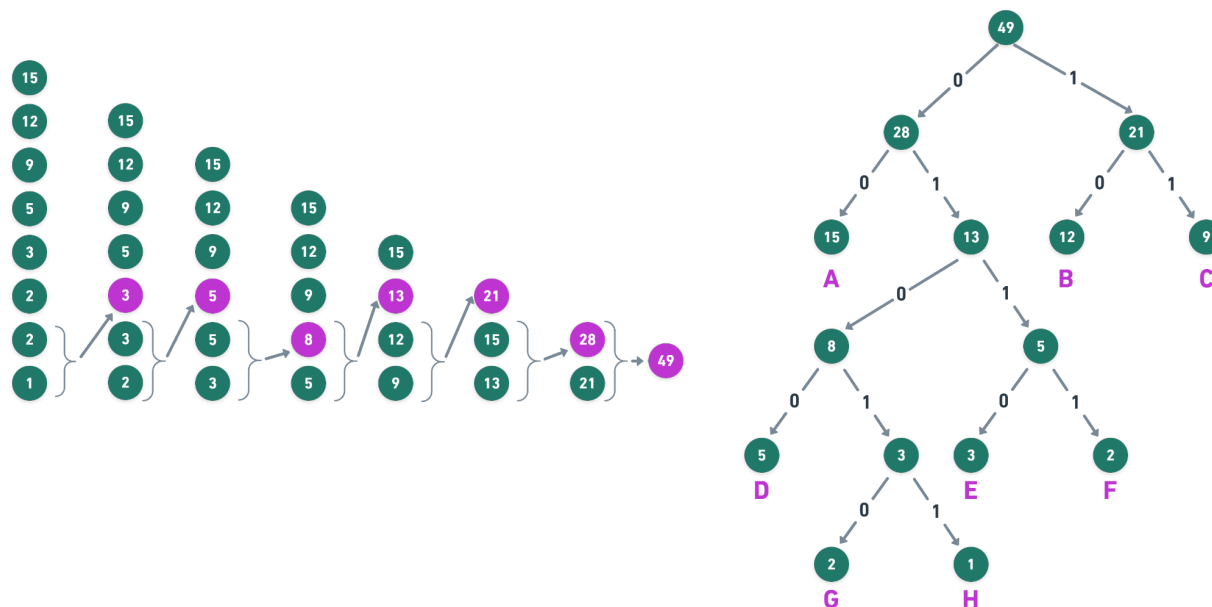
בהתאמה שמים לשכיחות הגבוהה ביותר $\alpha = 0$, לשניה $\beta = 10$ ולשלישית $\delta = 11$. נטייל בעץ ונשלוק את מילות הקוד:

A	10
B	11
C	00
D	011
E	01010
F	01000
G	01011
H	010010



סעיף ג'

ניצור עץ הפמן בינארי ואז נחשב לפי אורך ממוצע $E(C, P) = \sum_{i=1}^n |c_i| \cdot p_i$ ונעשה השוואה.



כעת נבצע חישוב:

- עבור עץ הפמן טרינרי נקבל: $2 \cdot 36 + 3 \cdot 5 + 4 \cdot 3 + 5 \cdot 2 + 6 \cdot 2 + 7 \cdot 1 = 128$.
 - עבור עץ הפמן בינארי נקבל: $5 \cdot 3 + 4 \cdot 10 + 2 \cdot 36 = 127$.
- לכן, עץ הפמן בינארי עדיף כי חסכנו איתו בסיבית.

שאלה 4

נתון הא"ב $\Sigma = \{a, b, c, d\}$ וברצוננו לקודד את ההודעה *bad* באמצעות דחיסה אריתמטית אדפטיבית. השלים את הטבלה הבאה הנבנית במהלך אלגוריתם הדחיסה: תזכורת: עבור א"ב עם 4 אותיות ההסתברות של האות $x \in \Sigma$ היא:

$$\Pr(x) = \frac{N(x)+1}{N(a)+N(b)+N(c)+N(d)+4}$$

כאשר $N(x)$ הוא מספר המופעים של x בקטע המחרוזת שהתקבל.

פתרון

נתונה הטבלה הבאה, כל מה שמודגש בשחור היה נתון לפני הפתרון כבר בטבלה:

נמלא את הטבלה:

התו	הסתברות a	הסתברות b	הסתברות c	הסתברות d	low	high	range
	1/4	1/4	1/4	1/4	0	1	1
b	1/5	2/5	1/5	1/5	1/4	1/2	1/4
a	1/3	1/3	1/6	1/6	1/4	3/10	1/20
d	2/7	2/7	1/7	2/7	7/24	3/10	1/120

הסבר:

נזכר בנוסחאות:

$$low_bound(s_i) = \sum_{j=1}^{i-1} \Pr(s_j)$$

$$high_bound(s_i) = \sum_{j=1}^i \Pr(s_j)$$

נפעל לפי סדר השורות (לפי האלגוריתם הנלמד):

$$range = high - low$$

$$high = low + high_bound(s_i) \cdot range$$

$$low = low + low_bound(s_i) \cdot range$$

שאלה 5

נתונה המחרוזת הבאה: "_wed_we_wee_web_wet".

א. הצג את המילון המתקבל מהפעלת דחיסת LZW ואת הקובץ הדחוס.

הנח כי המילון ההתחלתי מכיל את כל 256 אותיות ה-ASCII.

כמה סיביות יכיל קובץ זה בהנחה שהמילון יכול להכיל 512 כניסות?

ב. הצג את הקובץ הדחוס שנוצר באמצעות אלגוריתם LZSS.

מה גודלו (במספר סיביות) של הקובץ **כולו**, בהנחה שרכיב הצבעה מכיל 12 ביטים ורכיב תו בודד מכיל 8 סיביות?

הערה: אם יש מספר אפשרויות בחר את הייצוג המשתמש במספר הקטן ביותר של סיביות.

פתרון סעיף א'

לפי הנתון אנחנו נניח כבר שהמילון מלא מתא 0 עד התא 255 ונתחיל למלא החל מהתא 256.

נפעל לפי אלגוריתם דחיסת LZW שנלמדה בהרצאה ונעבור על המחרוזת הנתונה משמאל לימין:

אינדקס	התו w מהמחרוזת	התו הבא k	פלט	הכנסה למילון
256	_	w	_	_w
257	w	e	w	we
258	e	d	e	ed
259	d	_	d	d_
	_	w		
260	_w	e	256	_we
261	e	_	e	e_
	_	w		
	_w	e		
262	_we	e	260	_wee
	e	_		
263	e_	w	261	e_w
	w	e		
264	we	b	257	web
265	b	_	b	b_
	_	w		
	_w	e		
266	_we	t	260	_wet
	t	EOF	t	

הצגת המילון כנדרש

Code	Phrase
0	ASCII
...	
255	
256	_w
257	we
258	ed
259	d_
260	_we
261	e_
262	_wee
263	e_w
264	web
265	b_
266	_wet

הצגת הקובץ הדחוס כנדרש - קידוד סופי:

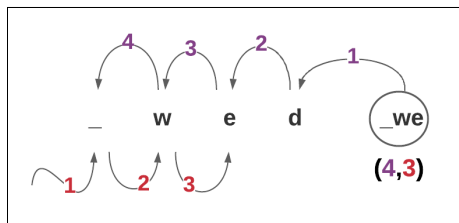
$_w, e, d, 256, e, 260, 261, 257, b, 260, t$

לגבי מספר הסיביות, למדנו שברגע שהמילון מתמלא צריך להכפיל אותו פי 2 בגודל שלו. תחילה קיבלנו מילון שכבר מלא עם 256 תאים תפוסים (בגלל ASCII) ולכן אנחנו כבר בהתחלה מגדילים את המילון $2 \cdot 256 = 512$ תאים. המילון לא אדפטיבי ולכן קידוד כניסה במילון יהיה באורך קבוע של $\log_2(512) = 9$ סיביות. בהודעה שלנו יש 12 הצבעות (הכנסות) למילון ולכן $12 \cdot 9 = 108$ סיביות.

פתרון סעיף ב'

נפעל לפי אלגוריתם דחיסת LZSS שנלמדה בהרצאה ונעבור על המחרוזת הנתונה משמאל לימין:

נקבל: $_wed(4, 3)(3, 3)(4, 4)b(4, 3)t$



בהרצאות למדנו שבשביל להבדיל בין Char לבין זוג סדור נעזר בביט מוביל של 0 או 1.

- $(0, char)$ - במקרה כזה, גודל התו תמיד יהיה 8 סיביות (ASCII) יחד עם ביט נוסף שיסמן זאת.

כלומר סה"כ יהיה לנו במקרה זה 9 bits .

- $(1, (offset, length))$ - קידוד כזה יהיה תלוי במספר הסיביות שהגדרנו עבור len, off וראינו

שפרמטרים אלה תלויים בגודל החלון שלנו.

יש לנו בקידוד 6 תווים ו-4 זוגות סדורים.

עבור התווים זה יעלה לנו ב- $54 = 6 \cdot 9$ סיביות.

עבור הזוגות הסדורים יש לנו 4 זוגות ובכל זוג יש את $length$ לכן נסכום את כולם $3 + 3 + 4 + 4 = 13$.

ונקבל בסה"כ:

$$106 = 54 + 4 \cdot 13 \text{ סיביות.}$$

מבחן 2017 - סמסטר א' - מועד א'

שאלה 1

(אין קשר בין סעיפי השאלה)
הוכח או הפרך:

- א.** הקוד {11, 111, 0100, 0110, 10111, 11011} ניתן לפענוח בצורה יחידה (UD).
ב. קוד מיידי הוא קוד שלם.

פתרון א'

הפרכה - נפעל לפי UD-Test שנלמד בהרצאה.

בהינתן הרשימה $L = \{11, 111, 0100, 0110, 10111, 11011\}$

- נעבור על כל הזוגות האפשריים ואם נמצא Dangling נוסף אותו לרשימה L .

נשים לב כי 11 וגם 111 יש Dangling של 1 לכן נוסף ל- L .

נשים לב כי 11 וגם 11011 יש Dangling של 011 לכן נוסף ל- L .

כעת $L = \{11, 111, 0100, 0110, 10111, 11011, 1, 011\}$

- נעבור שוב על כל הזוגות האפשריים ואם נמצא Dangling נוסף אותו לרשימה L .

נשים לב כי 1 וגם 111 יש Dangling של 11 לכן נוסף ל- L אך נשים לב כי 11 היא מילה מקורית ולכן

הקוד לא ניתן לפענוח בצורה יחידה UD.

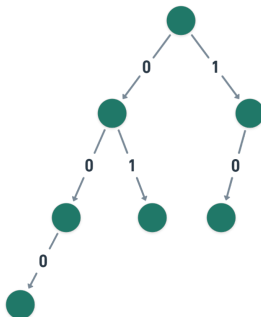
פתרון ב'

הפרכה - דוגמה נגדית:

קוד מיידי הוא לא בהכרח קוד שלם. אנו יודעים שקוד שלם ניתן לצייר עץ מלא.

עבור הקוד המיידי הבא: $\{a: 000, b: 01, c: 10\}$ נצייר עץ:

נשים לב כי העץ הוא לא עץ מלא ולכן הוא גם לא קוד שלם.



- משפט: קוד שלם יוצר עץ מלא.
- משפט: קוד פרפיקסי $\Leftrightarrow$ קוד מיידי.

שאלה 2

נתונות ההסתברויות $\{0.5, 0.3, 0.2\}$ עבור האותיות $\{A, B, C\}$ בהתאמה, ונתונה ההודעה הבאה:
 $ABABBAACAAAC$

- א.** בנה קוד *Tunstall* עבור מילות קוד באורך 3.
- ב.** בנה קוד *Tunstall* עבור מילות קוד באורך 4.
- ג.** דוחסים את ההודעה באמצעות הקוד שקיבלת בסעיף א' ו-ב' בשיטה החמדנית משמאל לימין. איזה קוד דוחס יותר טוב? נמק.

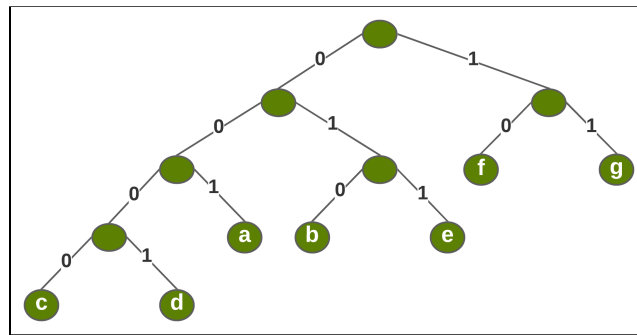
נושא שלא נלמד הסמסטר.

שאלה 3

א. האם קיים קוד הפמן קנוני שבו מספר מילות הקוד באורך 3 הוא שלוש, ומילת הקוד הראשונה בבלוק זה היא 001? אם כן, צייר עץ המתאים לקוד כזה, אחרת, הסבר מדוע קוד כזה אינו אפשרי.

ב. האם קיים קוד הפמן קנוני שבו קיימות 5 מילות קוד באורך 6 ומילת הקוד הראשונה בבלוק זה היא 000100? אם כן, צייר עץ המתאים לקוד כזה, אחרת, הסבר מדוע קוד כזה אינו אפשרי.

פתרון א':



פתרון ב':

לא קיים קוד הפמן קנוני שכזה שמילת הקוד הראשונה עבור הבלוק היא 000100. לפי האלגוריתם של הפמן קנוני וגם הראינו בכיתה שעבור $firstcode[i] = \frac{firstcode[i+1] + num[i+1]}{2}$ כלומר עבור מילת הקוד הראשונה בבלוק ה- i , אז $firstcode[i+1] + num[i+1]$ תמיד יהיה זוגי. לכן, עבור הבלוק השישי $i = 6$ אם יש 5 מילות קוד באורך 6 אז $num[6] = 5$ וגם נתון כי מילת הקוד הראשונה בבלוק צריכה להיות 000100 אז $firstcode[6] = 4$. לכן עבור $i = 5$ נקבל:

$$firstcode[5] = \frac{firstcode[6] + num[6]}{2} = \frac{4+5}{2} = \frac{9}{2}$$

כלומר $firstcode[5] = \frac{9}{2}$ אבל זה לא זוגי, ולכן זה לא ייתכן.

שאלה 4

נתון הא"ב הסדור $\Sigma = \{a, z, o\}$ וברצוננו לפענח את ההודעה 0.64 באמצעות פענוח של דחיסה אריתמטית אדפטיבית. הנח/י התפלגות אחידה של תווי הא"ב.

- א.** בנוסף לא"ב הסדור וההודעה הדחוסה, על המקודד לשלוח מידע נוסף. מהו אותו מידע?
ב. פענח את ההודעה. הנח כי המידע החסר הוא 4.

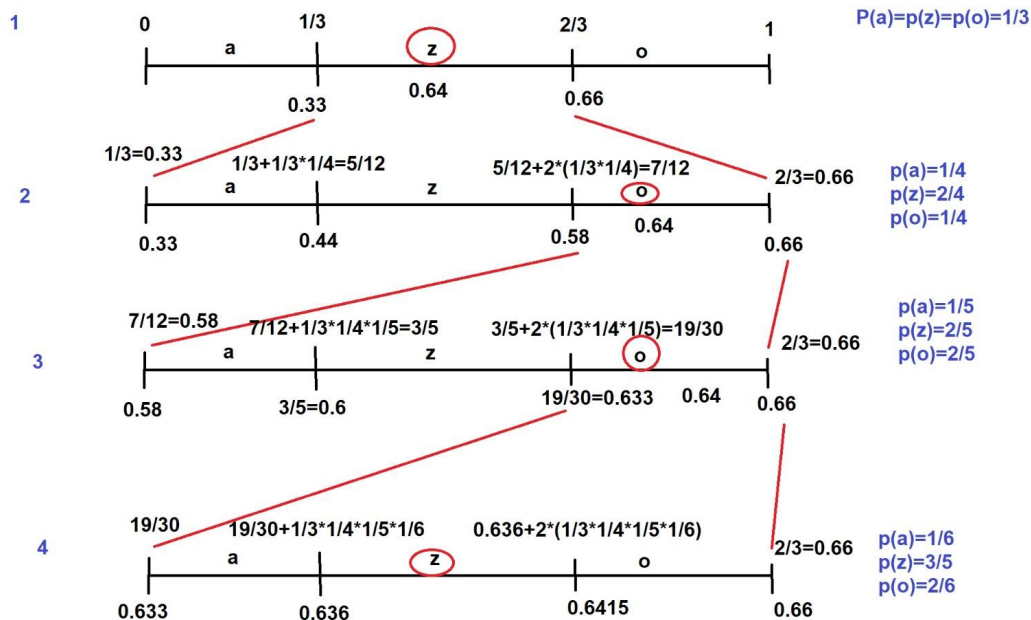
תזכורת: עבור א"ב עם 3 אותיות ההסתברות של האות $x \in \Sigma$ היא:

$$\Pr(x) = \frac{N(x)+1}{N(a)+N(z)+N(o)+3}$$

כאשר $N(x)$ הוא מספר המופעים של x בקטע המחרוזת שהתקבל.

פתרון א':

המידע הנוסף הוא כמה איטרציות צריך לבצע, כלומר מה אורך ההודעה בהנחה שזה הא"ב הסופי. היה ניתן גם להגדיר תו נוסף שמסמל EOF שמוסכם מראש.

פתרון ב':

לכן ההודעה שפענחנו היא zooz.

שלבי האלגוריתם:

אינדקס	התו w	התו הבא k	Output	הכנסה למילון	הרחבת המילון
3	b	c	b	bc	הרחבה 3
4	c	c	c	cc	
5	c	a	c	ca	
6	a	c	a	ac	
7	c	b	c	cb	הרחבה 4
	b	c			
8	bc	c	bc	bcc	
	c	c			
9	cc	c	cc	ccc	
	c	c			
	cc	c			
10	ccc	c	ccc	cccc	
	c	c			
	cc	c			
	ccc	c			
11	cccc	c	cccc	ccccc	

	c	a			
12	ca	c	ca	cac	
	c	c			
	cc	c			
	ccc	EOF	ccc		

המילון:

Index	Phrase	Code
0	a	00
1	b	01
2	c	10
3	bc	11
4	cc	100
5	ca	101
6	ac	110
7	cb	111
8	bcc	1000
9	ccc	1001
10	cccc	1010
11	ccccc	1011
12	cac	1100

כדי לכתוב את הרצף הבינארי אנחנו צריכים לעבוד לפי ה-Output של המילון.
 נעבור על המחרוזת משמאל לימין, לכל Output נתאים לפי הסדר את הסיביות שלו לפי מיקומו במילון.
 אם המילון הורחב אז מספר הסיביות גדלות ב-1 בכל הרחבה.
 למשל נשים לב שהפלט הראשון הוא b ואכן הצבנו 01.
 לאחר מכן, המילון הורחב ויצא הפלט c אז נציב 010.
 וכך הלאה ולבסוף נקבל:

b	c	c	a	c	bc	cc	ccc	cccc	ca	ccc
0	1	0	1	0	0	1	0	0	0	1
0	1	0	0	0	0	1	1	0	1	0
0	0	1	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	1

מבחן 2018 - סמסטר א' - מועד א'

שאלה 1

הוכח או הפרך:

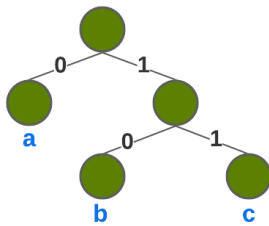
א. קוד שלם הוא קוד אופטימלי.

ב. הקוד $\{0, 01, 11, 1011\}$ ניתן לפענוח בצורה יחידה UD .

פתרון א'

הפרכה.

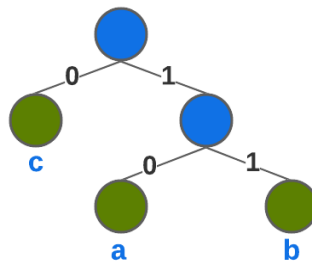
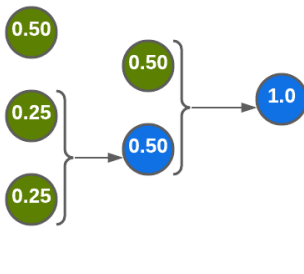
עבור $\Sigma = \{a, b, c\}$ נצייר עץ מלא שיגדיר לנו קוד שלם:



נגדיר את ההסתברויות בהתאמה: $P = \{a: 0.25, b: 0.25, c: 0.5\}$.
לכן, אורך מילת הקוד הממוצעת היא:

$$E(C, P) = \frac{1}{4} \cdot 1 + \frac{1}{4} \cdot 2 + \frac{1}{2} \cdot 2 = \frac{7}{4}$$

כעת, נציג קוד הפמן ונקבל:



לכן, אורך מילת הקוד הממוצעת היא:

$$E(C, P) = \frac{1}{4} \cdot 2 + \frac{1}{4} \cdot 2 + \frac{1}{2} \cdot 1 = \frac{3}{2}$$

כלומר קיבלנו קוד אופטימלי יותר.

הסבר: השיטה פה היא להציג קוד כאשר למילים הקצרות ביותר יקבלו הסתברות נמוכה. לאחר מכן, נשתמש בהפמן כדי להראות קוד אופטימלי, כלומר קוד יעיל יותר.

פתרון ב'

הפרכה. נראה שהקוד לא UD לפי UD Test:

- ניצור רשימה עם המילים המקוריות: $L = \{0, 01, 11, 1011\}$.
- נעבור על כל הזוגות ונכניס את ה-dangling במידה וקיים לתוך L .
 - עבור $0, 01$ נכניס 1 ונקבל: $L = \{0, 1, 01, 11, 1011\}$.
- נעבור שוב על כל הזוגות ונכניס את ה-dangling במידה וקיים לתוך L .
 - עבור $1, 11$ נכניס 1 , אך הוא כבר שייך לשם והוא לא מילה מקורית אז נמשיך.
 - עבור $1, 1011$ נכניס 011 ונקבל: $L = \{0, 1, 01, 11, 011, 1011\}$.
- נעבור שוב על כל הזוגות ונכניס את ה-dangling במידה וקיים לתוך L .
 - עבור $0, 011$ נכניס 11 , אך 11 היא מילה מקורית בקוד ולכן הקוד הוא לא UD .

שאלה 2

שאלה זו עוסקת בפתיח prelude של קובץ דחוס.

א. מהם המרכיבים של הפתיח לקובץ המקודד על סמך הקוד הבא, במקרה בו המודל מתאר את העץ הבינרי המלא המתאים לקוד, כנלמד בכתה.
 $\{0010, 011, 0011, 0100, 0101, 000, 1000, 1111, 1001, 1010, 1011, 110, 1110\}$
 עבור הא"ב $\{a, b, c, d, e, f, g, h, i, j, k, l, m\}$ בהתאמה.
 שים/י לב, עליך לתאר את **כל** המרכיבים בפתיח, להסביר את הייצוג של כל רכיב, ולחשב את מספר הסיביות הדרושות לייצוג כל רכיב בדרך היעילה ביותר ומספר הסיביות הכולל.
ב. בהנחה שהקוד בסעיף א' מתאים לקוד הפמן עבור הסתברויות כלשהן, המר/י את הקוד של סעיף א' לקוד הפמן קנוני. כעת, ייצג/י את הפתיח לפי השיטה שנלמדה בכתה לייצוג קוד הפמן קנוני. שוב, עליך לתאר את **כל** המרכיבים בפתיח, להסביר את הייצוג של כל רכיב, ולחשב את מספר הסיביות הדרושות לייצוג כל רכיב ומספר הסיביות הכולל.

פתרון א'

ב-Prelude של עץ בינארי יופיעו:

1. גודל הא"ב עומד על 8 סיביות.
2. אותיות הא"ב, כל אות מיוצגת ב-8 סיביות לפי ASCII ולכן $8 \cdot 13$.
3. הקוד, למדנו בהרצאה כי מספר הביטים הנחוצים לקידוד עץ בינארי מלא עם n עלים הוא $2n - 1$ ולכן במקרה שלנו: $25 = 13 \cdot 2 - 1$.
 לכן, מספר הסיביות הכולל הוא $137 = 8 + 8 \cdot 13 + 25$.

פתרון ב'

ב-Prelude של הפמן קנוני יופיעו:

1. גודל הא"ב עומד על 8 סיביות.
 2. אותיות הא"ב, כל אות מיוצגת ב-8 סיביות לפי ASCII ולכן $8 \cdot 13$.
 3. לכל בלוק - מילת הקוד הראשונה בכל בלוק, את התו הראשון בבלוק ואת האורך בייצוג C_γ .
 נחשב את מספר הסיביות עבור הרכיב השלישי, אורכי מילות הקוד הם 3, 4. כלומר יש לנו 2 בלוקים.
 - עבור אורך 4, משתתפים $\{a, c, d, e, g, h, i, j, k, m\}$ וכן מילות הקוד יהיו:
 $\{0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001\}$
 לכן, עבור אורך 4, מילת הקוד הראשונה היא 0000 ולכן 4 סיביות, התו הראשון a דורש 8 סיביות וייצוג האורך לפי C_γ היא 110 00 כלומר אורך של 5 סיביות.
 - עבור אורך 3, משתתפים $\{b, f, l\}$ וכן מילות הקוד יהיו: $\{000, 001, 010\}$.
 לכן, עבור אורך 3, מילת הקוד הראשונה היא 000 ולכן 3 סיביות, התו הראשון b דורש 8 סיביות וייצוג האורך לפי C_γ היא 01 1 כלומר אורך של 3 סיביות.
- סה"כ סיביות:

$$8 + 13 \cdot 8 + (4+8+5) + (3+8+3) = 135 \text{ bits}$$

שאלה 3

נתונה ההודעה הבאה מעל הא"ב $\{a, b, c, d, e, f, g, h\}$ שבו האותיות מופיעות עם שכיחות 12, 11, 8, 7, 4, 4, 3, 1 בהתאמה.

בנה/י קוד הופמן 4-ארי. לא"ב זה, ז"א מילות הקוד בנויות מהתווים $\{\alpha, \beta, \gamma, \delta\}$ ולא מהסיביות $\{0, 1\}$ כרגיל. כדי לקבל את הקידוד הבינרי הסופי, התווים ב- A מתורגמים לבינרית בצורה אופטימלית, ז"א התרגום הוא לאחד מהקודים $\{0, 10, 110, 111\}$ או $\{00, 01, 10, 11\}$. מהו הקוד שנוצר? מה מספר סיביות ההודעה הדחוסה בשיטה זו? הסבר!

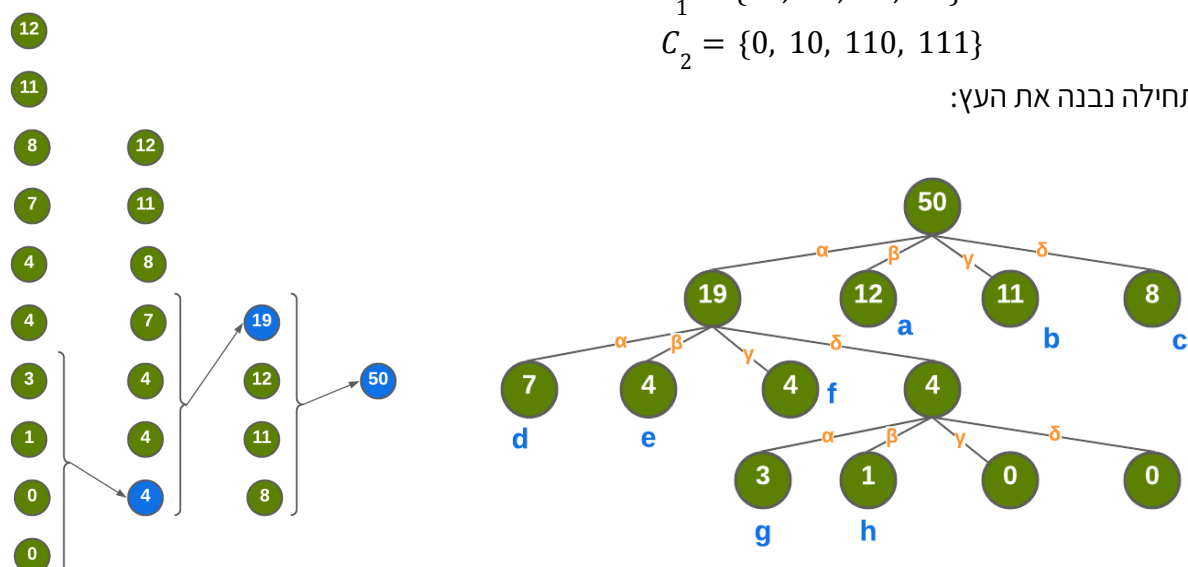
פתרון

אנחנו צריכים לבדוק אילו מהקודים הבאים מקיימים את התרגום בצורה האופטימלית לבינארית:

$$C_1 = \{00, 01, 10, 11\}$$

$$C_2 = \{0, 10, 110, 111\}$$

תחילה נבנה את העץ:



נשים לב שכדאי לתת אות קצרה ל- δ ואחריה יותר ארוכה ל- γ וככה הלאה עד ל- α .
נתרגם לפי 2 הקודים ונבדוק אופטימליות בין שניהם על ידי חישוב אורך מילת הקוד הממוצעת:

$$C_2 = \{\alpha = 0, \beta = 10, \gamma = 110, \delta = 111\}$$

s_i	c_i	תרגום של C_1	אורכים של C_1	תרגום של C_2	אורכים של C_2
a	β	01	2	10	2
b	γ	10	2	110	3
c	δ	11	2	111	3
d	$\alpha\alpha$	0000	4	00	2
e	$\alpha\beta$	0001	4	010	3
f	$\alpha\gamma$	0010	4	0110	4
g	$\alpha\delta\alpha$	001100	6	01110	5
h	$\alpha\delta\beta$	001101	6	011110	6

כעת נבדוק מי יותר אופטימלי:

$$E(C_1) = \frac{8 \cdot 2 + 11 \cdot 2 + 12 \cdot 2 + 7 \cdot 4 + 4 \cdot 4 + 4 \cdot 4 + 3 \cdot 6 + 1 \cdot 6}{50} = 2.92$$

$$E(C_2) = \frac{8 \cdot 3 + 11 \cdot 3 + 12 \cdot 2 + 7 \cdot 2 + 4 \cdot 3 + 4 \cdot 4 + 3 \cdot 6 + 1 \cdot 5}{50} = 2.92$$

כלומר לא משנה איזה קוד נבחר שניהם מספקים אותה אופטימליות.
נבחר ב- C_1 .

לכן מספר הסיביות של ההודעה הדחוסה היא:

$$8 \cdot 2 + 11 \cdot 2 + 12 \cdot 2 + 7 \cdot 4 + 4 \cdot 4 + 4 \cdot 4 + 3 \cdot 6 + 1 \cdot 6 = 146 \text{ bits}$$

שאלה 4

נתונה המחזורות הבאה:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
a	b	b	a	a	b	b	a	a	b	a	b	b	a	a	a	a	b	a	a	b	b	a

עליך לקודד את הקובץ משמאל לימין באמצעות אלגוריתם LZSS בצורה חמדנית, בהנחה שרכיב הצבעה מכיל 2 סיביות עבור ה-length ו-3 סיביות עבור ה-offset, ורכיב תו בודד הוא בעל 5 סיביות.

א. איזה אורכים של מחזורות כדאי לקודד באמצעות זוג סדור (off, len) , ומה טווח ההיסט $(offset)$? הסבר!

ב. כיצד תקודד/י את ההודעה בצורה הטובה ביותר? מה מספר הסיביות הכולל של ההודעה הדחוסה?

פתרון א'

אלגוריתם LZSS מורכב מ: $(0, char) + (1, (offset, length))$.

- כאשר עבור $(0, char)$ הסיביות לפי הנתון הן $1 + 5 = 6$.
- כאשר עבור $(1, (offset, length))$ הסיביות לפי הנתון הן $1 + 2 + 3 = 6$.

כלומר, הם זהים מבחינת מספר הסיביות ולכן אין עדיפות מבחינת איכות הדחיסה.

- עבור ה- $offset$, טווח ההיסט הוא מ- $2^0 \leq off \leq 2^3$ כלומר $1 \leq off \leq 8$ תחת ההנחה שההיסט נשמר בייצוג הבינארי שלו. אין צורך לרשום שיש להעתיק 0 תווים ולכן "הרווחנו עוד תו". כלומר, משום שאין טעם לחזור אחורה 0 תווים, אז נגדיר עבור 3 סיביות:

מספר התווים ללכת אחורה	קוד
1	000
2	001
3	010
4	011
5	100
6	101
7	110
8	111

- עבור ה- $Length$, באותו האופן, אנחנו לא להעתיק מחרוזת בגודל 1 ולכן נוכל להעתיק עד 5 תווים.

מספר התווים להעתיקה	קוד
2	00
3	01
4	10
5	11

פתרון ב'

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
a	b	b	a	a	b	b	a	a	b	a	b	b	a	a	a	a	b	a	a	b	b	a

שיטת הקידוד בצורה הטובה ביותר היא:

$$\varepsilon(T) = abba(4, 5)(3, 2)(6, 4)(8, 4)(3, 2)(4, 2)$$

מספר הסיביות הכולל הוא:

- עבור תווים בודדים נקבל: $4 \cdot (5 + 1) = 24$
- עבור זוגות נקבל: $6 \cdot (3 + 2 + 1) = 36$

ולכן סה"כ נקבל: $24 + 36 = 60 \text{ bits}$.

שאלה 5

נתונה המחרוזת הבאה:

11010111011011

א. פענח את ההודעה לפי קוד פיבונאצ'י $Fib1$. (כלומר, עליך לרשום את החלוקה למילות קוד והאינדקס המתאים לכל מילת קוד).

ב. האם ניתן לפענח את אותה המחרוזת לפי קוד $Fib2$?
במידה וכן, פענח את המחרוזת. אחרת, רשום את הפענוח שקיבלת בסעיף א' כקוד $Fib2$.

פתרון א'

תחילה נחשב את פיבונאצ'י בייצוג בינארי:

	8	5	3	2	1
1 =					1
2 =				1	0
3 =			1	0	0
4 =			1	0	1
5 =		1	0	0	0
6 =		1	0	0	1
7 =		1	0	1	0
8 =		1	1	0	0

נסדר את פיבונאצ'י לצורה של קוד, כלומר נוסף 1 מוביל ואחרי זה נהפוך כל קוד:

	פיבונאצ'י בינארי	$Fib1$
1 =	1	11
2 =	10	011
3 =	100	0011
4 =	101	1011
5 =	1000	00011
6 =	1001	10011
7 =	1010	01011

לכן: **11010111011011**כלומר פענחנו את: **1742**.

פתרון ב'

נשתמש בשיטה השניה של *Fib2*.

1. מורידים את ה-1 שהוספנו מימין בסוף כל מילת קוד שקיבלנו מ-*Fib1*.
2. נוסיף 10 לכל מילת קוד בהתחלה (משמאל).
3. נדחוף את כל הקודים שהיו קיימים רמה אחת מתחת (כלומר 2 יקבל את מה שהיה ל-1 ו-3 יקבל את מה שהיה ל-2 וכו'...) ואז נוסיף את 1 כמילת קוד חדשה.

	<i>Fib1</i>	<i>Fib2</i>
1 =	11	1
2 =	011	101
3 =	0011	1001
4 =	1011	10001
5 =	00011	10101
6 =	10011	100001
7 =	01011	101001

לכן: **11010111011011**

כלומר פענחנו את: **151221**.

מבחן 2018 - סמסטר א' - מועד ב'

שאלה 1

נתונים אורכי מילות הקוד $\{4, 4, 4, 3, 3, 1\}$. האם ניתן ליצור קוד חסר רישות בעל אורכים כנ"ל?

פתרון א'

נפעל לפי אי שוויון קראפט כך שאם $K(C) \leq 1$ אז ניתן ליצור קוד חסר רישות בעל אורכים כנ"ל.

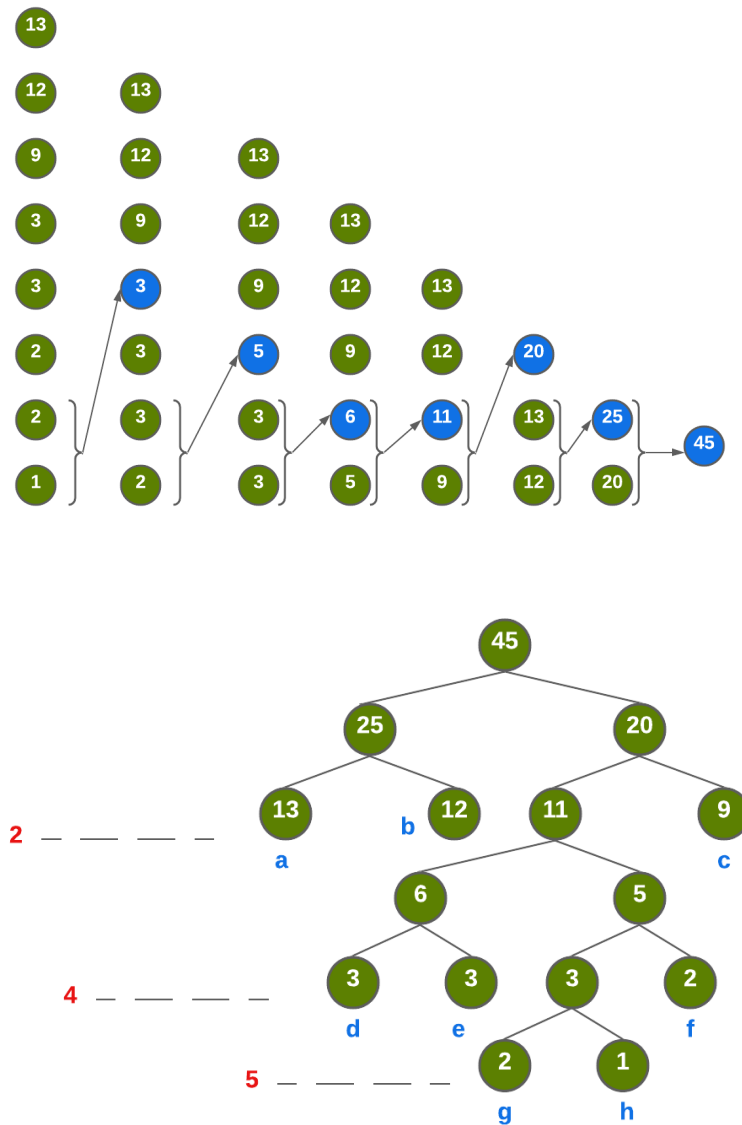
$$K(C) = \sum_{i=1}^n 2^{-|c_i|} = 3 \cdot 2^{-4} + 2 \cdot 2^{-3} + 1 \cdot 2^{-1} = \frac{3}{16} + \frac{2}{8} + \frac{1}{2} = \frac{15}{16} < 1$$

ולכן, ניתן ליצור קוד חסר רישות בעל אורכים כנ"ל.

שאלה 2

נתונה מחרזת מעל א"ב $\Sigma = \{a, b, c, d, e, f, g, h\}$ בשכיחויות $\{13, 12, 9, 3, 3, 2, 2, 1\}$ בהתאם.
א. מצאו את עץ ההפמן בעל העומק הגדול ביותר למחרזת זו.
ב. מהו אורך המחרזת?
ג. הוכח או הפרך: אם במחרזת באורך n כל תו מופיע בדיוק m פעמים, בקוד אופטימלי, אורך מילת הקוד של כל תו יהיה זהה בכל התווים.

פתרון א'



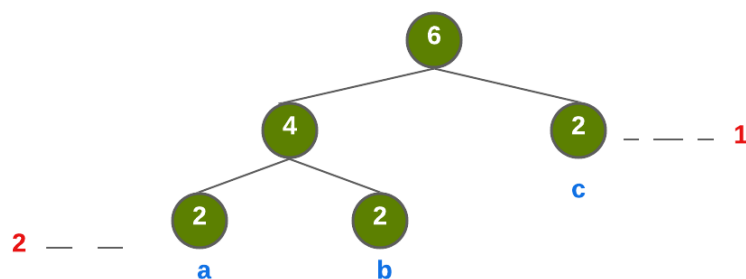
פתרון ב'

אורך המחרוזת היא:

$$2 \cdot (13 + 12 + 9) + 4 \cdot (2 + 3 + 3) + 5 \cdot (1 + 2) = 115$$

פתרון ג' - הפרכה.נבחר עבור מחרוזת $aabbcc$ כאשר $n = 6, m = 2$.

לפי עץ הפמן נקבל:

כלומר, קיבלנו $\{a: 00, b: 01, c: 1\}$ ולא כל אורך מילת הקוד של כל זהה לאחר.

שאלה 3

נגדיר את סדרת טריבונאצ'י:

$$T(0) = T(1) = 1$$

$$T(2) = 2$$

$$T(n) = T(n-1) + T(n-2) + T(n-3)$$

נתונה התחלה של סדרה אינסופית: (111, 0111, 1111, 00111, 10111, 01111, 000111,.....)
מה הקשר בין שתי הסדרות?

פתרון

מתחילים את סדרת טריבונאצ'י מהאיבר השני, כלומר 1, 2, 4, 7, ומוצאים את הביטוי הבינארי שלו:

	7	4	2	1
1 =				1
2 =			1	0
3 =			1	1
4 =		1	0	0
5 =		1	0	1
6 =		1	1	0
7 =	1	0	0	0

נסדר את טריבונאצ'י לצורה של קוד, כלומר נוסיף 1 מוביל ואחרי זה נהפוך כל קוד:

	טריבונאצ'י בינארי	Fib11
1 =	1	111
2 =	10	0111
3 =	11	1111
4 =	100	00111
5 =	101	10111
6 =	110	01111
7 =	1000	000111

כלומר על כל מחרוזת נוסף 11 וזה הקשר בין הסדרות הנ"ל.

שאלה 4

נתון הא"ב $\Sigma = \{a, b, c, d\}$ וברצוננו לקודד את ההודעה bad באמצעות דחיסה אריתמטית אדפטיבית. השלם את הטבלה הבאה הנבנית במהלך אלגוריתם הדחיסה: תזכורת: עבור א"ב עם 4 אותיות ההסתברות של האות $x \in \Sigma$ היא:

$$\Pr(x) = \frac{N(x)+1}{N(a)+N(b)+N(c)+N(d)+4}$$

כאשר $N(x)$ הוא מספר המופעים של x בקטע המחרוזת שהתקבל.
א. חשב את המחרוזת bad בקוד אריתמטי אדפטיבי $\Sigma = \{a, b, c, d\}$.
ב. חשב את התוצאה בדרך אחרת.

פתרון א'

נתונה הטבלה הבאה, כל מה שמודגש בשחור היה נתון לפני הפתרון כבר בטבלה:

נמלא את הטבלה:

התו	הסתברות a	הסתברות b	הסתברות c	הסתברות d	low	high	range
	1/4	1/4	1/4	1/4	0	1	1
b	1/6	2/6	1/6	1/6	1/4	1/2	1/4
a	1/3	1/3	1/6	1/6	1/4	3/10	1/20
d	2/7	2/7	1/7	2/7	7/24	3/10	1/120

הסבר:

נזכר בנוסחאות:

$$low_bound(s_i) = \sum_{j=1}^{i-1} \Pr(s_j)$$

$$high_bound(s_i) = \sum_{j=1}^i \Pr(s_j)$$

נפעל לפי סדר השורות (לפי האלגוריתם הנלמד):

$$range = high - low$$

$$high = low + high_bound(s_i) \cdot range$$

$$low = low + low_bound(s_i) \cdot range$$

פתרון ב'

ניתן לחשב את התוצאה ע"י מכפלת ההסתברויות כפי למדנו בהרצאה: $\frac{1}{4} \cdot \frac{1}{5} \cdot \frac{1}{6} = \frac{1}{120}$

שאלה 5

א. רשום את *minimal binary code* עבור $|\Sigma| = 11$.
 ב. האם ניתן לפענח את המחרוזת הבאה 110001110 לפי C_8 ?

פתרון א'

לפי תחום ייצוג סיביות: $2^3 < 11 < 2^4$.

נרצה לדעת איך לחלק את המילים לפי האורכים האלה כאשר חלק מהם באורך 3 וחלק באורך 4.

כדי לדעת את החלוקה נחשב: $2^4 - 11 = 5$, כלומר יש לנו 5 מילים באורך 3 ו-6 מילים באורך 4.

נצייר בצד עץ עם 11 עלים, 5 מהם בעומק 3 ו-6 מהם בעומק 4 ואז נציג את הקודים:

קוד	מילה
000	a
001	b
010	c
011	d
100	e
1010	f
1011	g
1100	h
1101	i
1110	j
1111	k

פתרון ב' - האם ניתן לפענח את המחרוזת הבאה 1100011110 לפי C_8 ?
ה- C_8 תמיד בנוי בהתחלה מכתוב אונרי אז נעבור על המחרוזת משמאל לימין עד נפגש ב-0 הראשון שנראה.

1100011110

המספר הזה בדצימלי הוא **3**.
אנחנו יודעים גם לפי האלגוריתם שאנחנו מחסירים את ה-1 המוביל עבור ייצוג בינארי ולכן לפי המסומן:

1100011110

מדובר בעצם ב-100 שבייצוג בינארי זה **4**.

1100011110

לאחר מכן לוקחים את 3 כי אורך המילה הוא 4 ולכן קיבלנו 111 עם 1 מוביל כלומר קיבלנו 1111 ובדצימלי זה **15**.

1100011110

נשאר עם 0 במחרוזת, 0 באונרי זה 1, כלומר נוסיף 1 מוביל לכלום, שזה בעצם נותן לנו 1 אז נוסיף אותו למחרוזת ונקבל לסיום: **151**. כלומר כן, ניתן לפענח את המחרוזת הזאת.

מבחן 2019 - סמסטר א' - מועד א'

שאלה 1

- א. האם $\{01, 000, 110, 0111, 1011, 10101, 111101\}$ הוא UD ?
- ב. איך התשובה שלנו ב-א' מסתדרת עם העובדה ש: $\sum_{i=1}^7 2^{-l_i} < 1$?
- כאשר l_i - מספר הסיביות של מילת הקוד ה- i ית המופיעה בסעיף א'.

פתרון א'

הקוד לא UD . נבדוק לפי UD Test:

- נגדיר רשימה חדשה L המכילה את כל מילות הקוד המקוריות:
 $L = \{01, 000, 110, 0111, 1011, 10101, 111101\}$
- נעבור על כל הזוגות ונכניס את ה-dangling במידה וקיים לתוך L .
 - עבור $01, 0111$ יש 11 , היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך... $L = \{01, 11, 000, 110, 0111, 1011, 10101, 111101\}$
- נעבור שוב על כל הזוגות ונכניס את ה-dangling במידה וקיים לתוך L .
 - עבור $11, 111101$ יש 1101 , היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך...
 - עבור $11, 110$ יש 0 , היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך... $L = \{0, 01, 11, 000, 110, 1101, 0111, 1011, 10101, 111101\}$
- נעבור שוב על כל הזוגות ונכניס את ה-dangling במידה וקיים לתוך L .
 - עבור $0, 000$ יש 00 , היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך...
 - עבור $11, 1101$ יש 01 אך היא מילת קוד מקורית לכן הקוד לא UD .

פתרון ב' -

אם מתקיים $K(C) \leq 1$ עבור קוד C כלשהו (לא בהכרח UD) אזי קיים C' אחר, כך ש:

$$1. E(C, P) = E(C', P)$$

$$2. |C'| = |C|$$

$$3. C' \text{ הוא קוד חסר רישות (prefix-free).}$$

במילים אחרות, קיים **קוד רגעי** (חסר רישות) אופטימלי (בפרט או ייחודי).

כלומר, זה לא מחייב ש- C הוא פרפיקסי ולכן C לא חייב להיות UD למרות ש: $K(C) \leq 1$.
 למה? - כי קיים C' אחר שמקיים את התכונות הנ"ל שהוא כן UD .

שאלה 2

- א.** מהו קוד אפיקסי?
- ב.** תנו דוגמה לקוד אפיקסי העומד בתנאים האלו: מילת קוד אחת באורך 2, 4 מילות קוד באורך 3, 4 מילות קוד באורך 4 כך שהקוד יהיה אפיקסי.

פתרון א'

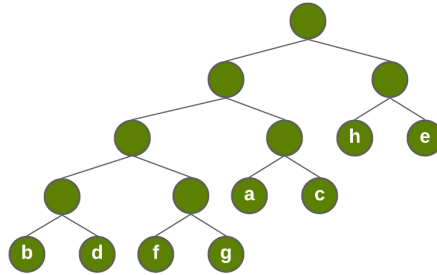
כזכור, קוד נקרא קוד חסר רישות או קוד פרפיקסי, אם אף מילת קוד אינה רישא של מילת קוד אחרת. באופן סימטרי נגדיר קוד חסר סיפות או קוד סופיקסי, בו אף מילת קוד אינה סיפא של מילת קוד אחרת. למעשה, הקוד המתקבל מהפיכת מילות הקוד של קוד חסר רישות, הוא קוד חסר סיפות. קוד נקרא קוד **אפיקסי**, אם הקוד הוא גם קוד פרפיקסי וגם קוד סופיקסי.

פתרון ב'

$$C = \{10, 111, 000, 001, 011, 0100, 1100, 0101, 1101\}$$

שאלה 3

א. באלגוריתם לבניית עץ קנוני שלמדנו, מה יכול $firstcode[]$ בעץ:



ב. הוכח / הפוך ע"י דוגמה נגדית: לכל i , ה- $firstcode[i]$ הוא תמיד זוגי.

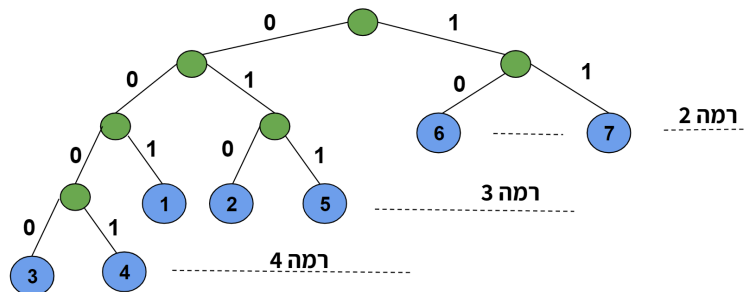
פתרון א'

תחילה לפי האלגוריתם נמלא את המערך שסופר את מילות הקוד באותו האורך: $num[] = \{0, 2, 2, 4\}$.
 לפי שלבי האלגוריתם: $firstcode[i] = \frac{firstcode[i+1] + num[i+1]}{2}$ לכל $1 \leq i \leq n - 1$.
 לכן הוא יכול: $firstcode[] = \{2, 2, 2, 0\}$.

פתרון ב'

הפרכה.

המערך $firstcode[]$ מייצג לנו גם את מספר הצמתים הפנימיים בכל רמה, כפי שהוכחנו במטלה 3 שאלה 1.
 נראה עץ לדוגמה:



בעץ זה יש $firstcode[] = \{2, 2, 1, 0\}$ כלומר עבור $firstcode[3] = 1$ הוא מספר אי זוגי.

שאלה 4

לא נלמד הסמסטר.

פתרון א'

שאלה 5הציגו על ידי שימוש ב- *Sequiter*:

א. a^4

ב. a^8

ג. מה ההכללה עבור a^{2^k} , כלומר, מה החוקיות שחוזרת על עצמה?**תזכורת האלגוריתם:**החוקים:

1. לא נשאר 2 תווים סמוכים שיחזרו על עצמם יותר מפעם אחת.

2. חוק צריך להופיע לפחות פעמיים.

האלגוריתם:

כל עוד אנחנו לא ב-EOF, אז נבדוק:

1. אם יש 2 תווים צמודים שחוזרים על עצמם 0 ניצור חוק דקדוק חדש שמתקן את זה.

2. אם יש חוק שמשתמשים בו רק פעם אחת - נסיר אותו ונציג במקומו את הצורה המוכלת שלו.

פתרון א'

$$a^4 \rightarrow aaaa \rightarrow AA$$

פתרון ב'

$$a^8 \rightarrow aaaaaaaaaa \rightarrow AAAA \rightarrow BB$$

פתרון ג'

החוקיות שחוזרת על עצמה לכל $k \geq 1$ היא שניתן להציג כל a^{2^k} עם דקדוק חדש וסופי בעל 2 תווים בלבד. למשל:

$$a^2 \rightarrow aa$$

$$a^4 \rightarrow AA$$

$$a^8 \rightarrow BB$$

$$a^{16} \rightarrow AAAAAAAAAA \rightarrow BBBB \rightarrow CC$$

כלומר באופן כללי, לכל $L_k \rightarrow L_{k-1} L_{k-1}$ כאשר תנאי ההתחלה הוא $L_1 \rightarrow aa$.

מבחן 2021 - סמסטר א' - מועד א'

שאלה 1

הוכח או הפרך באמצעות דוגמה נגדית:

נתון הקוד $C = \{A = 10, B = 011, C = 101, D = 010, E = 0111\}$

1. הטקסט 011110 ניתן לפענוח בצורה יחידה באמצעות C .
 2. הקוד C הוא UD - Uniquely Decipherable.
 3. (סעיף זה לא קשור לקוד C).
- קיים $n \geq 1$ כך ש- n מילות הקוד הראשונות של קוד פיבונאצ'י מהוות קוד שלם.

פתרון א'

הוכחה.

הקוד 011110 הוא קוד שניתן לפענוח בצורה יחידה.

המילים היחידות שמקיימות הם: $A=10$, $E=0111$.

כלומר **011110**.

פתרון ב'

נבדוק לפי $UD Test$:

$C = \{A = 10, B = 011, C = 101, D = 010, E = 0111\}$

- נגדיר רשימה חדשה L המכילה את כל מילות הקוד המקוריות:

$L = \{10, 011, 101, 010, 0111\}$

- נעבור על כל הזוגות ונכניס את ה-dangling במידה וקיים לתוך L .

○ עבור 10, 101 יש 1, היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך...

○ עבור 011, 0111 יש 1, היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך...

$L = \{1, 10, 011, 101, 010, 0111\}$

- נעבור שוב על כל הזוגות ונכניס את ה-dangling במידה וקיים לתוך L .

○ עבור 1, 10 יש 0, היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך...

○ עבור 1, 101 יש 01, היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך...

$L = \{0, 1, 01, 10, 011, 101, 010, 0111\}$

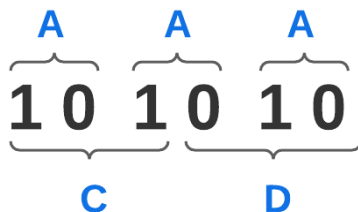
- נעבור שוב על כל הזוגות ונכניס את ה-dangling במידה וקיים לתוך L .

○ עבור 0, 011 יש 11, היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך...

○ עבור 0, 0111 יש 111, היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך...

○ עבור 0, 010 יש 10, היא לא מילת קוד מקורית לכן נכניס לרשימה ונמשיך...

לכן, הקוד הזה הוא לא UD .
הפרכה - צריך להראות דוגמה נגדית, כלומר נראה קידוד שאפשר לפרש אותו בשני אופנים.



פתרון ג'

דרוש תיקון: זה הפרכה, בשביל לבדוק ש: $K(C) = 1$ קוד שלם צריך לתחילה לוודא ש C הוא UD ומה שכתבתי זה לא קוד UD .

הוכחה:

נעזר בטענה אם $K(C) = \sum_{i=1}^n 2^{|c_i|} = 1$ אז C הוא קוד שלם.

צריך להראות שקיים n שבה שעבורו הקוד הוא שלם.

נבחר $n = 4$ ונראה $K(C) = 1$.

מילות הקוד הם: $C_4 = \{1, 10, 100, 101\}$ ולפי אי שוויון קראפט נקבל:

$$K(C_4) = \sum_{i=1}^n 2^{|c_i|} = 1 \cdot \frac{1}{2} + 1 \cdot \frac{1}{4} + 2 \cdot \frac{1}{8} = 1$$

כלומר $K(C_4) = 1$ ולכן C_4 הוא קוד כך ש-4 מילות הקוד הראשונות של קוד פיבונאצ'י מהוות קוד שלם.

שאלה 2

נתון הא"ב $\Sigma = \{a, b, c\}$ עם ההסתברויות: $P = \{p_a = 0.4, p_b = 0.1, p_c = 0.3\}$ בהתאמה, וכן נתון תו מיוחד, \$, המשמש כ- EOF שהסתברותו היא $p_\$ = 0.2$. האינטרוול מחולק ע"פ הסדר הלקסיקוגרפי $\Sigma = \{\$, a, b, c\}$, כאשר \$ תופס את תת האינטרוול הפותח ב-0 וכו'...

א. נתון כי ההודעה T_1 , קודדה באמצעות דחיסה אריתמטית **סטטית** ל-0.559326.

פענח/י את ההודעה. הראה/י את כל השלבים והחישובים.

ב. בדחיסה אריתמטית התייחסנו לגודל האינטרוול האחרון. מה גודלו ע"פ הדחיסה של סעיף א'?

ג. נתונה ההודעה $T_2 = \$abc$ שקודדה באמצעות דחיסה אריתמטית **אדפטיבית**.

המאותחלת, כרגיל, בהתפלגות אחידה. מה גודלו של האינטרוול האחרון ע"פ דחיסה זו?

שים לב, אין צורך לחשב את ההודעה הדחוסה עצמה.

ד. השווה/י את החסם התחתון של **גודל הקבצים** הדחוסים המתקבלים מהפעלת דחיסה אריתמטית **סטטית** ודחיסה אריתמטית **אדפטיבית** על ההודעה T_2 . איזו שיטה יעילה יותר עבור הודעה זו?

הסבר!. הבהרה - "החסם התחתון של גודל הקבצים הדחוסים" כלומר, אין צורך לקחת בחשבון את הסיביות שנוספות בסוף התהליך על מנת להפריד בין ה-low וה-high.

פתרון א'

תחילה נסדר את הנתונים:

תו	low	high	range
\$	0	0.2	0.2
a	0.2	0.6	0.4
b	0.6	0.7	0.1
c	0.7	1	0.3

נתון לנו ההודעה: $E = 0.559326$.

נפעל לפי השלבים של האלגוריתם: $E = \frac{\text{encoded} - \text{low}}{\text{range}}$

- בהתחלה נשים לב כי $E = 0.559326$ נמצא בקטע של **a**.
 - נעדכן: $E = \frac{\text{encoded} - \text{low}}{\text{range}} = \frac{0.559326 - 0.2}{0.4} = 0.898315$
- כעת $E = 0.898315$ נמצא בקטע של **c**.
 - נעדכן: $E = \frac{\text{encoded} - \text{low}}{\text{range}} = \frac{0.898315 - 0.7}{0.3} = 0.66105$
- כעת $E = 0.66105$ נמצא בקטע של **b**.
 - נעדכן: $E = \frac{\text{encoded} - \text{low}}{\text{range}} = \frac{0.66105 - 0.6}{0.1} = 0.6105$
- כעת $E = 0.6105$ נמצא בקטע של **b**.
 - נעדכן: $E = \frac{\text{encoded} - \text{low}}{\text{range}} = \frac{0.6105 - 0.6}{0.1} = 0.105$

• כעת $E = 0.105$ נמצא בקטע של \$ לכן, נעצור.

קיבלנו בסה"כ: $acbb\$$.

פתרון ב'

$$r = \prod_{i=1}^5 p(m_i) = p(a) \cdot p(c) \cdot p(b) \cdot p(b) \cdot p(\$) = \frac{3}{12500} = 0.00024$$

פתרון ג'

נתונה ההודעה $T_2 = abbc\$$ שקודדה באמצעות דחיסה אריתמטית **אדפטיבית**.

גודלו של האינטרוול האחרון ע"פ דחיסה זו היא:

$$\frac{1}{4} \cdot \frac{1}{5} \cdot \frac{2}{6} \cdot \frac{1}{7} \cdot \frac{1}{8} = \frac{1}{3360} \approx 0.00029761904$$

פתרון ד' - הפתרון מכיל הסבר מורחב ולא מחייב לפרט ברמה כזאת.

על מנת לחשב את גודל הקבצים הדחוסים המתקבלים מהפעלת דחיסה אריתמטית סטטית ודחיסה אריתמטית אדפטיבית, את תת הקטע האחרון נסמן ב- $[l, r + l]$ כך ש- r מייצג לנו את גודל תת הקטע הזה. אנו צריכים להעביר בסופו של דבר את המספר בתת הקטע האחרון, ככל שתת הקטע יותר קטן אנחנו נצטרך יותר סיביות כדי לייצג את המספר הזה. - למעשה, מספר הסיביות תלוי בגודל תת הקטע שלנו שזה בעצם גודל האינטרוול האחרון.

ראינו שגודל תת הקטע הזה הוא מכפלת ההסתברויות וכבר חישבנו אותו עבור דחיסה אריתמטית אדפטיבית. על מנת למצוא את גודל תת האינטרוול האחרון על פי דחיסה אריתמטית סטטית נחשב את כפל ההסתברות של ההודעה כך שההסתברויות הן: $p(a) = 0.4, p(b) = 0.1, p(c) = 0.3, p(\$) = 0.2$. ההודעה הנתונה היא: $T_2 = abbc\$$.

$$r = \prod_{i=1}^5 p(m_i) = p(b) \cdot p(a) \cdot p(b) \cdot p(c) \cdot p(\$) = \frac{3}{12500}$$

ולפי דחיסה אריתמטית אדפטיבית קיבלנו: $\frac{1}{3360}$.

כעת, על מנת להשוות את החסם התחתון של גודל הקבצים המתקבלים מהדחיסות השונות - נצטרך לחשב את כמות האינפורמציה של תת הקטע האחרון, כלומר: $\lceil \log_2 \left(\frac{1}{r} \right) \rceil$ ביטים כדי לייצג את r שמייצג את גודל האינטרוול האחרון.

אנחנו נראה כמה ביטים יקחו לנו לייצג מספר בתוך תת הקטע האחרון של הדחיסות השונות וכך נוכל להשוות ביניהם:

- עבור דחיסה אריתמטית סטטית נקבל: $\lceil \log_2 \left(\frac{1}{\frac{3}{12500}} \right) \rceil = 13$

- עבור דחיסה אריתמטית אדפטיבית נקבל: $\lceil \log_2 \left(\frac{1}{\frac{1}{3360}} \right) \rceil = 12$

ולכן, עבור ההודעה T_2 הדחיסה האדפטיבית יעילה יותר כי היא סיבית אחת פחות מהאריתמטית סטטית. כלומר, היא חוסכת בסיבית אחת בגודל הקובץ הדחוס.

שאלה 3

ברצוננו להשתמש באלגוריתם LZ78 על מנת לדחוס את המחרוזת הבאה:

$$M = A A B A B B B A B A A B A B B B A B B A B B A$$

נניח כי הא"ב הוא $\Sigma = \{A, B\}$ כך ש- A מקודד ע"י הסיבית 0 ו- B מקודד ע"י הסיבית 1. הטבלה, כרגיל, מאותחלת רק ב- ε באינדקס 0 ומתמלאת בהתאם למחרוזת. השלם/י את הטבלה הבאה משמאל לימין באופן דומה למה שעשינו בהרצאה: העמודה *Phrase* היא תת המחרוזת המתאימה לכניסה במילון שהאינדקס שלה הוא *index*. העמודה השלישית היא הצגת המחרוזת כזוג סדר $(index, symbol)$ והעמודה האחרונה היא הקידוד הבינרי המתאים (ולא רק גודלו). מספר הכניסות בטבלה אינו מעיד בהכרח על גודל הטבלה בתשובה.

פתרון

	i	Phrase	קידוד במבנה (index, symbol)	מספר סיביות	הרצף הבינרי
0	0	ε	-	-	-
0	1	A	(0, A)	0+1	0
1	2	AB	(1, B)	1+1	1 1
2	3	ABB	(2, B)	2+1	10 1
	4	B	(0, B)	2+1	00 1
3	5	ABA	(2, A)	3+1	010 0
	6	ABAB	(5, B)	3+1	101 1
	7	BB	(4, B)	3+1	100 1
4	8	ABBA	(3, A)	4+1	0011 0
	9	BBA	(7, A)	4+1	0111 0
	10				
	11				
	12				

הסבר עבור הרצף הבינארי:

- עבור הרצף הבינארי אנחנו יכולים לייצג אותו לפי גודל המילון בכל שלב הכנסה. בכל שלב נסתכל על $(index, symbol)$ ונשאל כמה סיביות ניתן לייצג את *index* בהתאם לאורך ה- $(i - 1)$ בייצוג הבינארי שלו ואז נצמיד אליו את ה-*symbol* בייצוג הבינארי שלו.
- למשל, עבור $i = 2$ אז האורך של $i - 1 = 1$ בבינארי הוא 1 ולכן נייצג את *index* במקרה הזה בסיבית 1, ואז עבור *symbol* יש לנו את B שמיוצג בסיבית 1 וביחד זה 11.
 - למשל, עבור $i = 6$ אז האורך של $i - 1 = 5$ בבינארי הוא 3 ולכן נייצג את *index* במקרה הזה ב-3 סיביות כלומר 101, ואז עבור *symbol* יש לנו את B שמיוצג בסיבית 1 וביחד זה 1011.

מבחן 2021 - סמסטר א' - מועד ב'

שאלה 1

הוכח או הפרך באמצעות דוגמה נגדית:

- הקוד $C = \{A = 00, B = 010, C = 0010, D = 010, E = 1001\}$ הוא Uniquely (UD) Decipherable.
- נניח שבקלט לאלגוריתם Huffman קיים תו ששכיחותו היא יותר מ-2/5. אזי חייבת להיות מילת קוד באורך 1.

פתרון א'

הפרכה - נראה דוגמה נגדית.



נשים לב שהאותיות B ו-D זהות, לכן הקוד לא ייחודי כי לפי ההגדרה ל-UD לא ייתכן ויש מילות קוד בעלות אותה המחרוזת, כלומר אין כאן ייחודיות.

פתרון ב' (לא פתרון סופי - לא פתרתי כי לא ריאלי למבחן פרונטלי).

הוכחה: נניח בשלילה שלא, ואם כך, כל מילות הקוד באורך לפחות 2. כיוון שעץ זה הינו עץ מלא, יש לפחות 4 עלים, כלומר 4 מילות קוד. נניח שהתו המדובר בשאלה הוא α .

נפצל למקרים: אם יש תו β בעל הסתברות גבוהה מזו של α , אז יש רק אחד כזה, משום ש:

$$\Pr(\alpha) = \frac{2}{5} + \varepsilon, \Pr(\alpha) = \frac{2}{5} + \varepsilon + \delta$$

$$\Pr(\alpha \cup \beta) = \frac{4}{5} + 2\varepsilon + \delta$$

ולכן לתווים הנותרים יהיו הסתברות קטנה ממש מ- $\frac{1}{5}$.

נרצה להוכיח של: β יש תו יחיד ול- β יהיה תו יחיד אמ"מ השלב האחרון הוא יחובר להסתברות נוספת שביחד יניבו את ההסתברות 1.

- למה זה? כי אחרת אם הוא מחובר בשלב מקדים יותר, והסתברותו נוספת פעמיים לפחות לאיזשהי הסתברות אחרת, אז עומקו יהיה לפחות 2 בעץ.

אם כך, נניח בשלילה שהתו נוסף בשלב מוקדים לשלב האחרון. **להמשיך...**

גרסה אחרת:

יש פה כיסוי של כל המקרים כי נתון יותר משתי חמישיות. כלומר התו מופיע בתדירות שלוש חמישיות או בתדירות של ארבע חמישיות. בגלל שנתון גם תדירות, אז ידוע שגודל הטקסט הוא חמש.

עבור המקרה שהתו מופיע בתדירות ארבע חמישיות, נותר תו נוסף שמופיע בתדירות של חמישית.

שהטווח מהאנתרופיה הוא בעצם ביט אחד עבור מילת קוד. קידוד מילת קוד עבור תו לפי כמות האינפורמציה תיתן מספר מסוים. קידוד לפי האפמן ייתן מילת קוד כך שהפרש האורכים לא יעלה אחד.

אם לוקחים את הנוסחא של כמות האינפורמציה עבור כל מספר בטווח הזה אז נקבל מספר קטן מאחד. זוהי בעצם האנטרופיה. למשל עבור $3/5$ מקבלים כמות אינפורמציה של 0.7369. אז קידוד לפי האפמן ייתן מילת קוד באורך אחד

ככה עבור כל הסתברות שנמצאת בטווח הזה. אם נניח תיקחי את $2/5$ בקצה ותעשי לו נוסחא של כמות האינפורמציה אז ייצא 1.32 אז קידוד לפי האפמן ייתן לך מילת קוד באורך 2 מכיוון שהפרש האורכים של מילות הקוד קטן מאחד

כל ההסתברויות בטווח יתנו מילת קוד באורך אחד

גם המרחק של הפמן מהאנתרופיה הוא הפרש של אחד לכל היותר

סליחה. כמות האינפורמציה

שאלה 2

בשאלה זו אין צורך להתייחס ל-prelude.

ברצוננו לדחוס את הטקסט loseless באמצעות Dynamic Huffman Coding ובאמצעות אלגוריתם Forward Looking שנלמדו בכיתה.

0	1	2	3	4	5	6	7
L	o	s	e	l	e	s	s

1. נניח כי אנו נמצאים באינדקס 3 בקובץ. יש לשים לב שהאינדקסים ממוספרים מ-0. כיצד נראה העץ Huffman הדינמי בשלב זה?
2. כיצד ישתנה העץ Huffman הדינמי באינדקס 4? הסבר. מה מילת הקוד של האות l בשלב זה?
3. כיצד ישתנה העץ Huffman הדינמי באינדקס 5? הסבר.
4. כיצד יראה העץ Huffman הדינמי באינדקס 4 כאשר פועלים לפי Forward looking? הסבר. מה מילת הקוד של האות l בשלב זה?

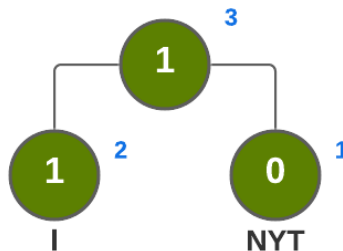
פתרון א'

נציג את שלבי הכניסות בעץ ונזכור כי צריך לשמור על תכונת האחים, כלומר צריך לשמור על סדר עולה של האינדקסים בין כל מספרי הצמתים, אם יש חריגה, נתקן את העץ בשביל לשמור על תכונת האחים. בנוסף, כל הכנסה של תו חדש לעץ מתבצע בעזרת גישה לצומת NYT.

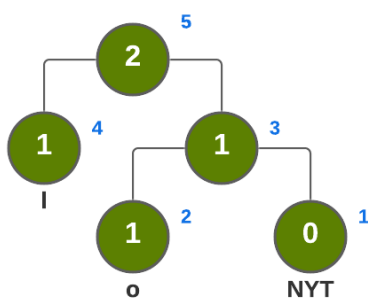
מתחילים עם צומת ריקה:



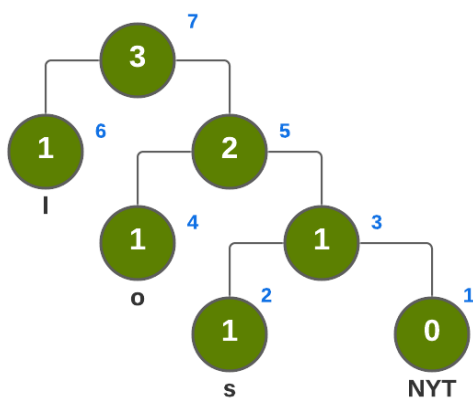
נכניס את התו הראשון l בצורה הבאה:



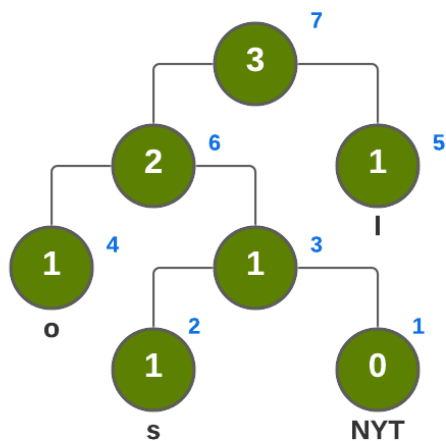
נכניס את התו השני **o** בצורה הבאה:



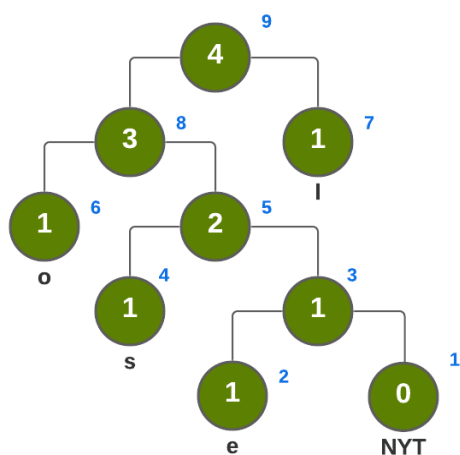
נכניס את התו השלישי **s** בצורה הבאה:



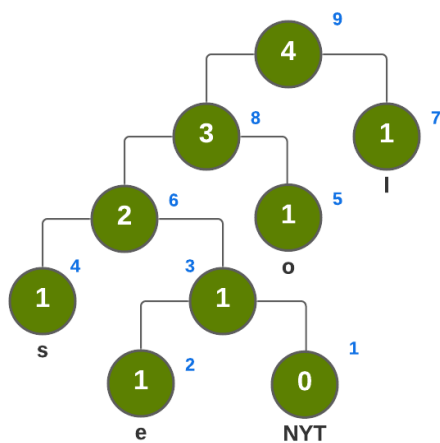
כעת ניתן לראות שיש כאן הפרה של תכונת האחים, כלומר צומת מספר 5 גדולה יותר מצומת מספר 6 וזה לא תקין, לכן, נבצע החלפה בין 2 תתי העצים האלה ונקבל:



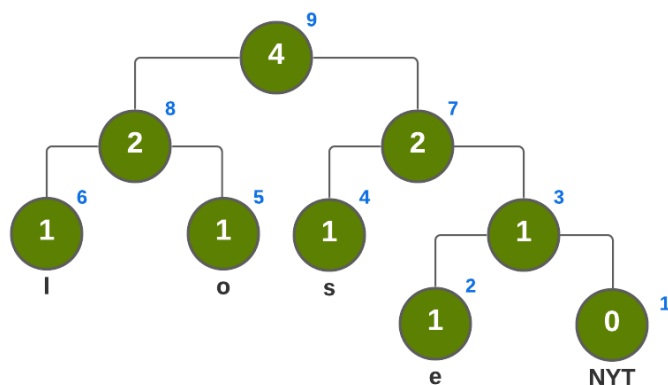
נכניס את התו הרביעי e בצורה הבאה:



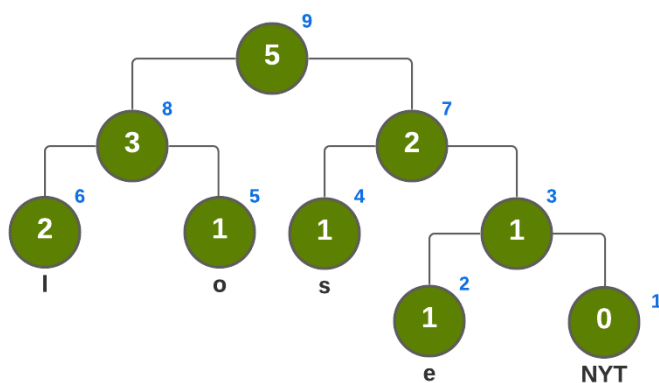
כעת, נשים לב שיש לנו כאן הפרה של תכונת האחים, צומת מספר 5 גדול יותר מ-6 וגם מ-7. נבצע כאן 2 הזזות, בשלב הראשון נבצע החלפה בין צומת 5 לצומת 6:



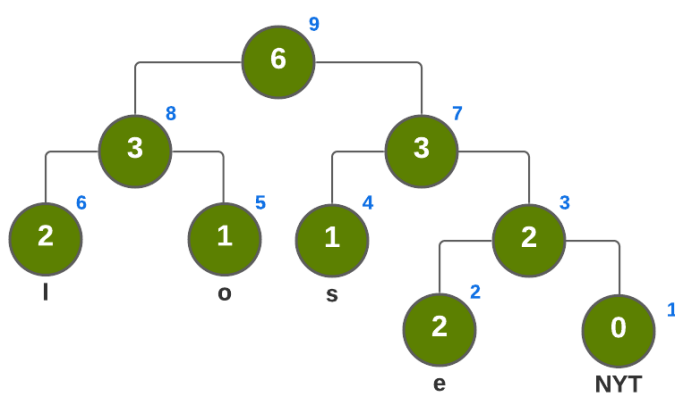
ובשלב השני נבצע הזזה בין צומת 6 המעודכנת לצומת 7 בצורה הבאה:



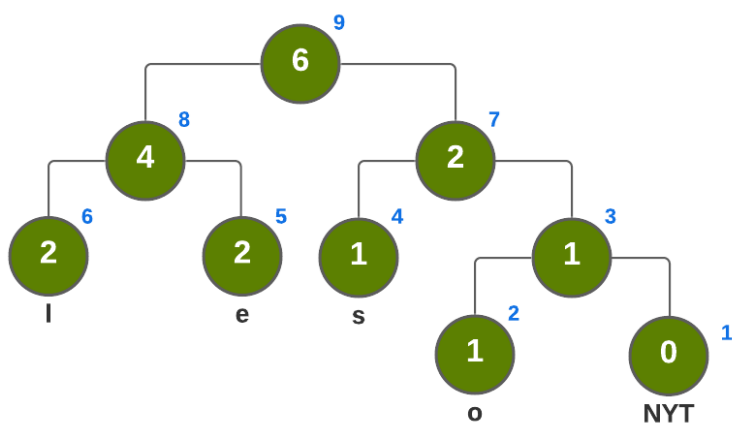
כעת נכניס את התו החמישי **l** בצורה הבאה:



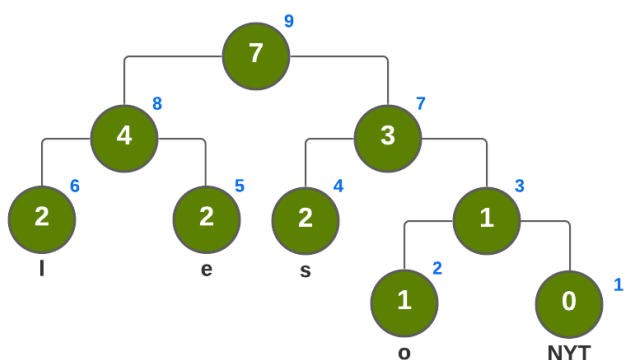
נכניס את התו השישי **e** בצורה הבאה:



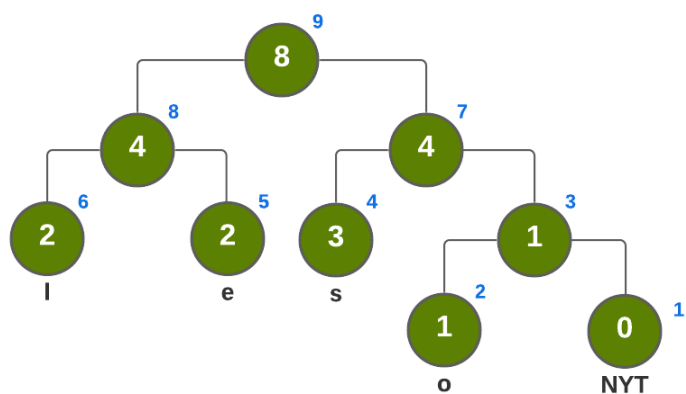
כעת נשים לב כי יש כאן שוב הפרה של תכונת האחים בין הצומת 3 לצמתים 4 ו-5. לכן, נבצע החלפה בין צומת מספר 2 לצומת מספר 5 באופן הבא:



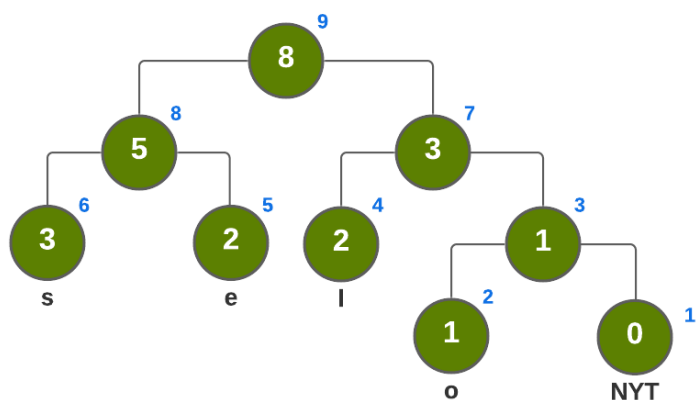
נכניס את התו השביעי s בצורה הבאה:



נכניס את התו השמיני והאחרון s בצורה הבאה:



נשים לב שיש כאן שוב הפרה של תכונת האחים, לכן נבצע החלפה בין צומת מספר 4 לבין צומת מספר 6:



פתרון א':

הציור של המצב כבר קיים בתיאור למעלה.

פתרון ב':

ההסבר נמצא בציור למעלה, מילת הקוד של א במצב זה היא 00 לפי המסלול על העץ.

פתרון ג':

ההסבר נמצא בציור למעלה.

פתרון ד':

לא נלמד הסמסטר.

שאלה 3

אחד מהמימושים שלמדנו עבור קידוד ארתמטי היה Incremental Coding. נתון הא"ב $\{A, B, C\}$ בהסתברות $\{0.8, 0.05, 0.15\}$ בהתאמה, כאשר האינטרוול מחולק לפי סדר לקסיקוגרפי ו- A תופס את תת האינטרוול הפותח ב-0, וכו'... מה יהיה הקובץ הדחוס ע"פ מימוש זה עבור ההודעה $ACBA$. יש לפרט את השלבים באמצעות הטבלה המצורפת.

הערות:

1. מספר השורות בטבלה אינו מעיד על מספר השלבים. באפשרותך להוסיף או להחסיר שורות.
2. יתכן וקיימות עמודות לא רלוונטיות לשלב מסויים (ולכן תתכנה עמודות שאינן מלאות).

הסבר:	הפלט לקובץ	scaling האינטרוול לאחר	האינטרוול	התו בהודעה

לא נלמד הסמסטר

מבחן 2022 - סמסטר א' - מועד א'

שאלה 1

הוכח או הפרך -

- א. נתון $C = \{A = 0, B = 01, C = 011, D = 111\}$. האם הקוד שלם?
- ב. נתון קוד C שהוא UD , בהינתן תו s עם הסתברות p_s אורך מילת הקוד שמתאימה ל- s גדולה או שווה לכמות האינפורמציה של s כלומר $|c_s| \geq [I(s)] = \lceil -\log_2(p_s) \rceil$.

פתרון א'

הוכחה.

בשביל לבדוק האם קוד C הוא קוד שלם צריך לוודא שהקוד הוא UD וגם שמתקיים: $K(C) = 1$.

נבדוק שהקוד UD : בעזרת $UD - Test$

- נאתחל רשימה $L = \{0, 01, 011, 111\}$ של כל מילות הקוד המקוריות.
- נעבור על כל זוגות המילים ונכניס את ה- $dangling suffix$ שנוצר מהשילוב שלהם לתוך L .
 - עבור 0, 01 נכניס 1.
 - עבור 01, 011 נכניס 1.
 - עבור 0, 011 נכניס 11.

$$L = \{0, 1, 11, 01, 011, 111\}$$

- נעבור שוב על כל הזוגות:
 - עבור 11, 111 נכניס 1.
 - עבור 0, 01 נכניס 1.
 - עבור 1, 11 נכניס 1.
 - עבור 1, 111 נכניס 11.
- הגענו למצב שבו הכנסנו שוב מילות קוד שהם לא מקוריות, לכן אפשר לעצור כאן והקוד C הוא UD .

נבדוק את תנאי שוויון קראפט:

$$K(C) = \sum_{i=1}^4 2^{-|c_i|} = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8} = 1$$

ולכן הקוד C הוא קוד שלם.

פתרון ב'

הפרכה.

נציג דוגמה נגדית עם הקוד $C = \{A = 0, B = 01, C = 011, D = 111\}$ על ההודעה $ABACD$ כך

שהסתברויות הן: $\Pr(A) = \frac{2}{5}, \Pr(B) = \Pr(C) = \Pr(D) = \frac{1}{5}$.

אז בהינתן תו A עם הסתברות $\frac{2}{5}$, נציג את אורך מילת הקוד: $|C_A| = 1$.

נציג את כמות האינפורמציה של A כלומר: $I(A) = \lceil -\log_2(p_s) \rceil = \lceil -\log_2\left(\frac{2}{5}\right) \rceil = 2$

קיבלנו:

$$|c_A| \geq [I(A)]$$

$$1 \geq 2$$

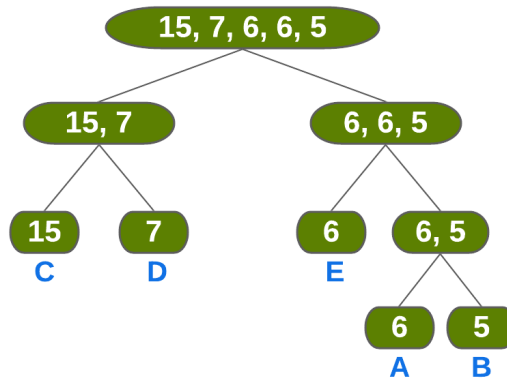
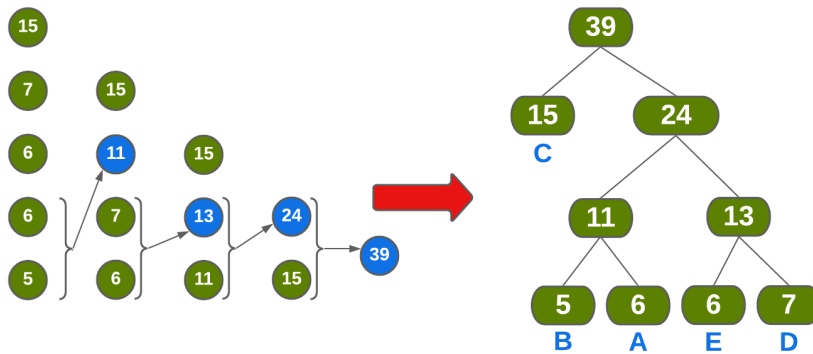
כלומר, מתקיים ש: $|c_A| < [I(A)]$ בשונה ממה צריך להראות ולכן לא ניתן להוכיח.

שאלה 2

נתון א"ב $\Sigma = \{A, B, C, D, E\}$ שבו האותיות מופיעות עם השכיחויות 6, 5, 15, 7, 6 בהתאמה.
א. בנה/י קוד שאננו פאנו.

ב. בדקו האם הקוד שקיבלתם הוא אופטימלי.

ג. קודדו את ההודעה לפי קוד פיבונאצ'י *Fib1* עם ההסתברויות הנתונות.

פתרון א'**פתרון ב'**

כדי לבדוק האם קוד הוא קוד אופטימלי נחשב את אורך מילת הקוד הממוצעת עבור הקוד C וגם ניצור באותו שיטה קוד הפמן C' וגם עבורו נחשב את אורך מילת הקוד הממוצעת ואם $E(C, P) \leq E(C', P)$ אז הקוד C הוא קוד אופטימלי כיוון שקוד

הפמן הוא תמיד קוד אופטימלי (בשונה מקוד שאננו פאנו של תמיד אופטימלי ולכן נדרשת כאן בדיקה שכזאת).

$$E(C, P) = \sum_{i=1}^n p_i \cdot |c_i| = \frac{6}{39} \cdot 3 + \frac{5}{39} \cdot 3 + \frac{15}{39} \cdot 2 + \frac{7}{39} \cdot 2 + \frac{6}{39} \cdot 2 = \frac{89}{39} \approx 2.282051$$

$$E(C', P) = \sum_{i=1}^n p_i \cdot |c_i| = \frac{6}{39} \cdot 3 + \frac{5}{39} \cdot 3 + \frac{15}{39} \cdot 1 + \frac{7}{39} \cdot 3 + \frac{6}{39} \cdot 3 = \frac{29}{13} \approx 2.23076$$

לכן קוד C' אופטימלי יותר ומכאן הקוד שקיבלנו לפי שאננו פאנו לא קוד אופטימלי.

פתרון ג'

קודדו את ההודעה לפי קוד פיבונאצ'י $Fib1$ עם ההסתברויות הנתונות.
 $\Sigma = \{A, B, C, D, E\}$ שבו האותיות מופיעות עם השכיחויות 6, 5, 15, 7, 6.
 נמיון את ההסתברויות שלנו מהגדול לקטן: $\{C = 15, D = 7, A = 6, E = 6, B = 5\}$.
 אנחנו ניתן מילות קוד קצרות יותר לתווים בעלי ההסתברות הגבוהה כדי לקודד באופן אופטימלי.

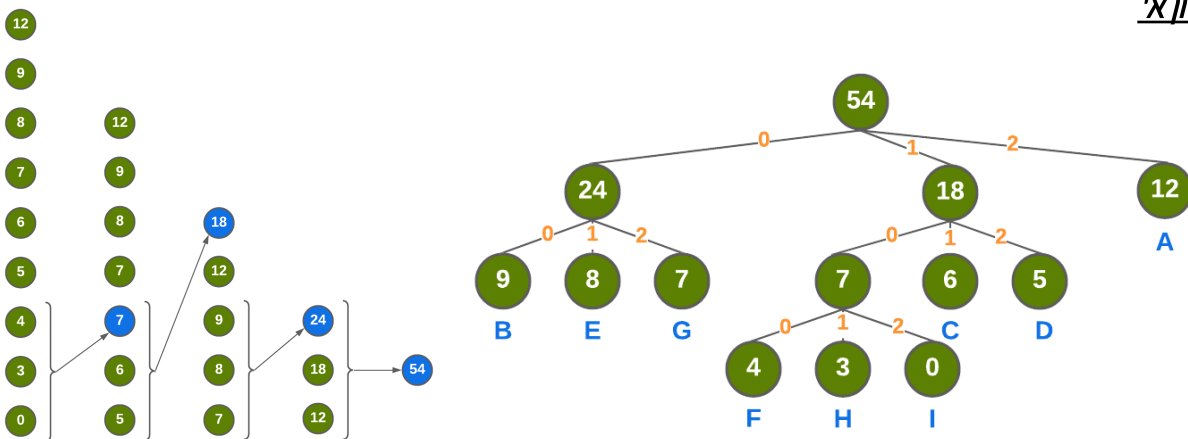
אינדקס	תו	הסתברות	פיבונאצ'י בינארי	$Fib1$
1	C	$\frac{15}{39}$	1	11
2	D	$\frac{7}{39}$	10	011
3	A	$\frac{6}{39}$	100	0011
4	E	$\frac{6}{39}$	101	1011
5	B	$\frac{5}{39}$	1000	00011

שאלה 3

נתונה ההודעה הבאה מעל הא"ב $\Sigma = \{a, b, c, d, e, f, g, h\}$ שבו האותיות מופיעות עם השכיחויות 12, 9, 6, 5, 8, 4, 7, 3 בהתאמה.

א. בנה/י עץ הופמן טרינרי.

ב. בנה מילות קוד טרינריות עבור האותיות, ואח"כ צור/י את מילות הקוד הבינריות על ידי החלפת שלושת התווים הטרינריים על ידי 0, 10, 11 בהתאמה.
דאג/י לבצע את ההחלפה בצורה הטובה ביותר מבחינת הדחיסה.

פתרון א'**פתרון ב'**

כדי להמיר למילות קוד בינאריות אנחנו נרצה לקודד עם האורכים הכי קצרים שאפשר ככל שניתן. אם נסתכל על הציור של העץ, תמיד נתבונן ברמה הכי נמוכה ונשאל את עצמנו באיזה צורה הכי נכון לבחור את 0, 10, 11 כך שנקבל את מספר הסיביות הקצר ביותר עד לעלים ברמה התחתונה ביותר. לכן נבצע את ההמרה הבאה: 0: 0, 1: 10, 2: 11.
נייצג את מילות הקוד באופן הבא:

מילות קוד בינאריות	מילות קוד טרינאריות	תו
11	2	A
00	00	B
1010	11	C
1011	12	D
010	01	E
1000	100	F
011	02	G
10010	101	H

שאלה 4

נתון הא"ב $\Sigma = \{a, b, c, d\}$ עם ההסתברויות: $P = \{p_a = 0.2, p_b = 0.2, p_c = 0.2, p_d = 0.4\}$.
א. נתון כי ההודעה T_1 , בעלת 5 תווים קודדה באמצעות דחיסה אריתמטית סטטית ל-0.14433.

פענח/י את ההודעה. הראה/י את כל השלבים והחישובים.

ב. בדחיסה אריתמטית התייחסנו לגודל האינטרוול האחרון. מה גודלו ע"פ הדחיסה של סעיף א'?

ג. נתונה ההודעה $T_2 = dabbc$ שקודדה באמצעות דחיסה אריתמטית **אדפטיבית**.

המאותחלת, כרגיל, בהתפלגות אחידה. מה גודלו של האינטרוול האחרון ע"פ דחיסה זו?
 שים/י לב, אין צורך לחשב את ההודעה הדחוסה עצמה.

פתרון א'

תחילה נסדר את הנתונים:

תו	low	high	range
a	0	0.2	0.2
b	0.2	0.4	0.2
c	0.4	0.6	0.2
d	0.6	1	0.4



נתון לנו ההודעה: $E = 0.14433$.

נפעל לפי השלבים של האלגוריתם: $E = \frac{encoded-low}{range}$, בנוסף נתון לנו שזה עבור 5 תווים.

• התו הראשון הוא **a** כי $E = 0.14433$ נמצא בקטע שלו.

$$E = \frac{0.14433-0}{0.2} = 0.72165 \quad \circ \text{ נעדכן:}$$

• התו השני הוא **d**.

$$E = \frac{0.72165-0.6}{0.4} = 0.304125 \quad \circ \text{ נעדכן:}$$

• התו השלישי הוא **b**.

$$E = \frac{0.304125-0.2}{0.2} = 0.520625 \quad \circ \text{ נעדכן:}$$

• התו הרביעי הוא **c**.

$$E = \frac{0.520625-0.4}{0.2} = 0.603125 \quad \circ \text{ נעדכן:}$$

• והתו האחרון הוא **d**.

$$E = \frac{1}{128} \quad \circ$$

ההודעה שפיענחנו היא: $T_1 = adbcd$.

פתרון ב'

$$r = \prod_{i=1}^k p(m_i) = \Pr(a) \cdot \Pr(d) \cdot \Pr(b) \cdot \Pr(c) \cdot \Pr(d) =$$

$$= 0.2 \cdot 0.4 \cdot 0.2 \cdot 0.2 \cdot 0.4 = \frac{4}{3125} = 0.00128$$

פתרון ג'

נתונה ההודעה $T_2 = dabbc$ שקודדה באמצעות דחיסה אריתמטית **אדפטיבית**. המאותחלת, כרגיל, בהתפלגות אחידה. מאחר ומדובר בדחיסה אריתמטית אדפטיבית, ההסתברויות משתנות כל הזמן בכל מעבר של תו, לכן, נתחיל בהסתברות אחידה על כל התווים ובהמשך הדרך נשנה את ההסתברויות בהתאם לתדירויות שלהם.

$$\Pr(d) \cdot \Pr(a) \cdot \Pr(b) \cdot \Pr(b) \cdot \Pr(c) = \frac{1}{4} \cdot \frac{1}{5} \cdot \frac{1}{6} \cdot \frac{2}{7} \cdot \frac{1}{8} = \frac{1}{3360}$$

שאלה 5

- א. בנו את הטבלה של המצב הנוכחי "ATGACGA" עם $k_{max} = 3$.
- ב. הוסיפו את התו T בסוף הקלט, מה ההסתברות ל-T הזה ללא EP.
- ג. מה ההסתברות ל-T הזה עם EP.

פתרון א'

k=0			k=1			k=2			k=3		
הקשר	תדירות	הסתברות	הקשר	תדירות	הסתברות	הקשר	תדירות	הסתברות	הקשר	תדירות	הסתברות
ε	$A = 3$	$\frac{3}{11}$	A	$T = 1$	$\frac{1}{4}$	AT	$G = 1$	$\frac{1}{2}$	ATG	$A = 1$	$\frac{1}{2}$
				$C = 1$	$\frac{1}{4}$		$\$ = 1$	$\frac{1}{2}$		$\$ = 1$	$\frac{1}{2}$
	$T = 1$	$\frac{1}{11}$		$\$ = 2$	$\frac{2}{4}$						
			T	$G = 1$	$\frac{1}{2}$	TG	$A = 1$	$\frac{1}{2}$	TGA	$C = 1$	$\frac{1}{2}$
	$G = 2$	$\frac{2}{11}$		$\$ = 1$	$\frac{1}{2}$		$\$ = 1$	$\frac{1}{2}$		$\$ = 1$	$\frac{1}{2}$
							$C = 1$	$\frac{1}{2}$			
	$C = 1$	$\frac{1}{11}$	G	$A = 2$	$\frac{2}{3}$	GA	$\$ = 1$	$\frac{1}{2}$	GAC	$G = 1$	$\frac{1}{2}$
				$\$ = 1$	$\frac{1}{3}$		$G = 1$	$\frac{1}{2}$		$\$ = 1$	$\frac{1}{2}$
	$\$ = 4$	$\frac{4}{11}$					$\$ = 1$	$\frac{1}{2}$			
			C	$G = 1$	$\frac{1}{2}$	AC			ACG	$A = 1$	$\frac{1}{2}$
				$\$ = 1$	$\frac{1}{2}$		$A = 1$	$\frac{1}{2}$		$\$ = 1$	$\frac{1}{2}$
							$\$ = 1$	$\frac{1}{2}$			

פתרון ב'

- אנחנו נרצה להכניס את T בלי עקרון ההרחקה כלומר עבור המצב $ATGACGAT$.
- בהקשר $k_{max} = 3$ אין לנו את CGA ולכן נעביר \$ בהסתברות $\Pr(\$) = 1$.
 - בהקשר $k = 2$ אז עבור GA נשים לב ש T לא שייך לשם ולכן נעביר \$ בהסתברות $\Pr(\$) = \frac{1}{2}$.
 - בהקשר $k = 1$ אז עבור A נשים לב ש T שייך לשם ולכן נעביר T בהסתברות $\Pr(T) = \frac{1}{4}$.
- לכן סה"כ נקבל:
- $$\$. (1), \$(\frac{1}{2}), T(\frac{1}{4}) = \lceil -\log_2(1) - \log_2(\frac{1}{2}) - \log_2(\frac{1}{4}) \rceil = 0 + 1 + 2 = 3$$

פתרון ג'

עקרון ההרחקה אומר שאם התו נמצא בהקשר גבוהה יותר ולא השתמשנו בו אז בטוח גם לא נשתמש בו בהקשר נמוך יותר ולכן נוכל להוריד אותו וההסתברות שלו תהיה פשוט 0.

בנוסף אנו יודעים כי לא ניתן להפעיל את עקרון ההרחקה על k_{max} .

אנחנו נרצה להכניס את T עם עקרון ההרחקה כלומר עבור המצב $ATGACGAT$.

- בהקשר $k_{max} = 3$ אין לנו את CGA ולכן נעביר \$ בהסתברות $\Pr(\$) = 1$.
- בהקשר $k = 2$ אז עבור GA נשים לב ש T לא שייך לשם ולכן נעביר \$ בהסתברות $\Pr(\$) = \frac{1}{2}$.
אך בגלל שלא השתמשנו ב- C אחרי GA אז בהקשר הנמוך הבא אנחנו בוודאי גם לא נשתמש בו.
- בהקשר $k = 1$ אז עבור A נשים לב ש T שייך לשם ולכן נעביר T בהסתברות $\Pr(T) = \frac{1}{3}$.
נשים לב כי בהקשר $k = 2$ לא השתמשנו ב- C ולכן בהקשר $k = 1$ אז $C = 0$.
כלומר:

k=1		
הקשר	תדירות	הסתברות
A	$T = 1$	$\frac{1}{1+0+2} = \frac{1}{3}$
	$C = 0$	$\frac{0}{1+0+2} = 0$
	$\$ = 2$	$\frac{2}{1+0+2} = \frac{2}{3}$

לכן סה"כ נקבל:

$$\$(1), \$(\frac{1}{2}), T(\frac{1}{3}) = \lceil -\log_2(1) - \log_2(\frac{1}{2}) - \log_2(\frac{1}{3}) \rceil \approx 0 + 1 + 1.584 \approx 2.584$$

אפשר לראות שעקרון ההרחקה יותר טוב כי הוא שיפר את הקידוד עם פחות סיביות $3 > 2.584$.

שאלות שעוד לא הופיעו במבחנים

השאלות מקורם דחיסת נתונים ב' אבל נלמדו גם אצלנו החל מ-2022 (היה שאלה מהקורס הזה שהופיע במועד 2022 שלנו).

שאלה 1

אלגוריתם PPM:

הנחי כי $k_{\max}=2$

א. מלא/י את הטבלה הבאה עבור ההודעה assanissimassa:

K=0			K=1			K=2		
CONTEXT	COUNTS	P	CONTEXT	COUNTS	P	CONTEXT	COUNTS	P

ב. נניח א"ב בן 28 תווים (26 תווי אנגלית, תו רווח ו-\$) ונניח ש-14 תווי המחרוזת

assanissimassa כבר קודדו.

עבור התווים הבאים, אם נתון שהם התו הבא בהודעה (משורשר לסופה), עליך לציין איזה תווים מקודדים במקרה זה, מה הסתברותם ובאיזה הקשר הם מקודדים. בכל סעיף חשב את כמות האינפורמציה של התוכן המקודד.

a. אם התו הבא הוא n?

b. אם התו הבא הוא s?

c. אם התו הבא הוא m?

d. אם התו הבא הוא d?

ג. בכל סעיף בדוק/י האם הפעלת עקרון ההרחקה, Exclusion Principal, תייעל את הדחיסה?

הסביר/י.

פתרון 1

הנחי כי $k_{\max}=2$

א. מלא/י את הטבלה הבאה עבור ההודעה :assanissimassa

K=0			K=1			K=2		
context	counts	p	context	counts	P	context	Counts	P
a	4	4/19	a	s=2 n=1 \$=2	2/5 1/5 2/5	as	s=2 \$=1	2/3 1/3
s	6	6/19	s	s=3 a=2 i=1 \$=3	3/9 2/9 1/9 3/9	ss	a=2 i=1 \$=2	2/5 1/5 2/5
n	1	1/19	n	i=1 \$=1	1/2 1/2	sa	n=1 \$=1	1/2 1/2
i	2	2/19	i	s=1 m=1	1/4 1/4	an	i=1 \$=1	1/2 1/2

				\$=2	1/2			
m	1	1/19	m	a=1 \$=1		ni	s=1 \$=1	1/2 1/2
\$	5	5/19				is	s=1 \$=1	1/2 1/2
						si	m=1 \$=1	1/2 1/2
						im	a=1 \$=1	1/2 1/2
						ma	s=1 \$=1	1/2 1/2

a.

$$-\log(1/2)=1 \text{ bit}$$

$$-\log(1/2)+-\log(2/5)=1+1.32 = 2.32 \text{ bit} \quad \mathbf{b.}$$

ניתן לשפר על ידי Exclusion – $-\log(1/2)=1$ – במקום 1.32.

$$-\log(1/2)+-\log(2/5)-\log(1/19)=1+1.32+4.25 = 6.57 \text{ bit} \quad \mathbf{c.}$$

$$-\log(1/2)+-\log(2/5)-\log(5/19)+\log(28)=1+1.32+1.93+4.8 = 9.05 \text{ bit} \quad \mathbf{d.}$$

ניתן לשפר על ידי Exclusion – $-\log(5/12)=1.26$ – במקום 1.93

בהקשר -1 מורידים את כל חמשת האותיות שנראו, ולכן d יכול להיות מקודד

בהסתברות $1/(28-5)=0.043$ עם 4.52 סביות במקום 4.8.

שאלה 2

יהי $BWT(T) = (ooppomm, 5)$ עבור טקסט נתון T .
 א. מצא/י את T . הסבר כל שלב בטרנספורמצית Burrows–Wheeler ההפוכה.

פתרון 2

א. ממיינים בסדר לקסיקוגרפי לא יורד את התווים ב- $L=ooppomm$, ומקבלים $F=mmooopp$.
 נקבל ש: $T=opompomp$.

$K=0$ $\underline{P}$ $\underline{K=1}$ $\underline{P}$ $\underline{K=2}$ $\underline{P}$
 $Z=4$ $4/13$ $Z | X=3$ $3/6$ $ZX | \begin{matrix} Z=1 \\ X=1 \\ S=2 \end{matrix}$ $\begin{matrix} 1/4 \\ 1/4 \\ 2/4 \end{matrix}$
 $Y=1$ $1/6$ $S=2$ $2/6$ $ZY | \begin{matrix} Z=1 \\ S=1 \end{matrix}$ $\begin{matrix} 1/2 \\ 1/2 \end{matrix}$
 $X=4$ $4/13$ $X | Z=1$ $1/6$ $XZ | \begin{matrix} Y=1 \\ S=1 \end{matrix}$ $\begin{matrix} 1/2 \\ 1/2 \end{matrix}$
 $Y=1$ $1/6$ $X | \begin{matrix} Y=1 \\ S=3 \end{matrix}$ $\begin{matrix} 1/6 \\ 3/6 \end{matrix}$ $XY | \begin{matrix} Z=1 \\ S=1 \end{matrix}$ $\begin{matrix} 1/2 \\ 1/2 \end{matrix}$
 $Y=2$ $2/13$ $Y | Z=2$ $2/3$ $YZ | \begin{matrix} X=2 \\ S=1 \end{matrix}$ $\begin{matrix} 2/3 \\ 1/3 \end{matrix}$
 $S=3$ $3/13$

(א) נאמן למה הוא חסר x אתה חייב לחזק במוקרה זה, מהי המושגית?

נקודת המסתברות של x מהוקדל $k=2$ $\rightarrow -\log\left(\frac{1}{2}\right) = 1 \text{ bit}$ $\frac{4}{6}$

סוגי ההתקדק (הוקדק $\frac{1}{2}$ מתקדק 3) $\Rightarrow$ $\left(\frac{1}{2}\right) \rightarrow$ $\frac{1}{4}$ בהסתברות $\frac{1}{4}$

במקרה זה עקרון ההתקדק אנו ולפי זה הוא הוא חסר רק $\frac{1}{4}$ $\frac{6}{6}$

$k_{max} - 1$ $\leftarrow$ למטה $\leftarrow$ כיוון שמאלין אם יש כסר המסתברות
אם אין אלוה' אלוה' מרסנית.

(ב) הוסף את y ופדוק ספס אלה:

$\# \left(\frac{1}{2}\right), \left(\frac{1}{6}\right) \rightarrow -\log\left(\frac{1}{2}\right) + -\log\left(\frac{1}{6}\right)$

$\downarrow \checkmark$ $\downarrow \checkmark$
 $k_{max}=2$ $k_{max}=1$

היחס, בצד את עקרון ההתקדק - (ל.פ. 2) $\rightarrow$ כחלק של 1 $\frac{5}{6}$

נאמן מהוקדל את x ! $\rightarrow$ כיוון שאתה כסר יודע שיש לא הומוס האלו.

$\# \left(\frac{1}{2}\right), \left(\frac{1}{4}\right) \rightarrow -\log\left(\frac{1}{2}\right) + -\log\left(\frac{1}{4}\right)$

$\downarrow$
מהי המין המוקדק?
(מהוקדל x ! $\rightarrow$ $\frac{1}{4}$ $\rightarrow$ $\frac{1}{2}$)

שאלה 4

שאלה זו דנה באלגוריתם PPM.

הנח/י כי $k_{\max} = 3$.

נניח א"ב בן 5 תווים $\{ 'a', 'b', 'c', 'd', '$' \}$, כאשר '\$' הוא תו מיוחד כנלמד בכתה. כמו כן, נניח ש-10 תווי המחרוזת `cacbcaabca` כבר קודדו באמצעות אלגוריתם PPM.

א. נתון שהתו הבא בהודעה (משורשר לסופה) הוא `a`. איזה תווים מקודדים במקרה זה? מה הסתברותם? הסברי/י.

ב. איך תשתנה תשובתך במידה ומופעל עקרון ההרחקה - Exclusion Principle?

ג. כיצד תשתננה תשובותיך לסעיפים א' ו-ב' במידה ובמקום `a` יהיה התו `b`?

ד. כיצד תשתננה תשובותיך לסעיפים א' ו-ב' במידה ובמקום `a` יהיה התו `d`?

פתרון 4

חסר

שאלה 5

תהי $T = \text{abcabcabc}$ מחרוזת מעל הא"ב $\Sigma = \{a, b, c\}$.

א. חשבי את BWT-Burrows-Wheeler's Transform על T .

פתרון 5

חסר

שאלה 7

1. נניח כי נתונה ההודעה T הבאה:

$T = \text{misses_miss_missouri}$

א. חשבי את $S = \text{BWT}(T)$

ב. חשבי את $R = \text{MTF}(S)$ כאשר $\Sigma = \{ _, e, i, m, o, r, s, u \}$

ג. הפעל/הפעילי את אלגוריתם הפמן על התוצאה

ד. הראה/הראי כיצד המפענח מחשב את T על סמך BWT.

פתרון 7

① $T = \text{misses_miss_missouri}$

	F	L
0	miss_missouri	misses
1	missouri	misses_miss
2	es_miss_missouri	miss
3	i	misses_miss_missou
4	i	ss_missouri
5	i	sses_miss_missouri
6	i	ssouri
7	miss_missouri	misses
8	misses_miss_missou	r
9	missouri	misses_miss
10	ouri	misses_miss_miss
11	r	misses_miss_missou
12	s	miss_missouri
13	s	missouri
14	s	es_miss_missouri
15	s	ouri
16	s	s_missouri
17	s	ses_miss_missouri
18	s	souri
19	u	r

$\text{BWT}(T) = (\text{sssrmmm-i-suesssio}, 8)$

② $\Sigma = \{ _, e, i, m, o, r, s, u \}$

$S = \text{sssrmmm-i-suesssio}$

MTF-list:

	0	1	2	3	4	5	6	7
-	e	i	m	o	r	s	u	
s	-	e	i	m	o	r	u	
r	s	-	e	i	m	o	u	
m	r	s	-	e	i	o	u	
-	m	r	s	e	i	o	u	
i	-	m	r	s	e	o	u	
-	i	m	r	s	e	o	u	

3

⊗ נתן את המילה קיבלנו: e i i i m m m o r s s s s s s s u

סה יהיה הצורה F

ס- מקום המל באינדקס i ב-F

בצורה L

⊗ ככה שכל האינדקס שקבלנו (8), נחשב מהי המילה המקורית לפי האפשרות

misses-miss-missouri : מילה

	S	F	L
0	7	- ← S	g
1	9	- ← S	l
2	12	e ← S	e
3	8	i ← r	t
4	16	i ← M	i
5	17	i ← M	b
6	18	i ← M	n
7	4	M ← -	h
8	5	(M)	i a
9	6	M ← -	m
10	19	0 ← S	g
11	3	r ← u	s
12	0	S ← e	
13	1	S ← S	

Page 3 / 5

10	19	0	←	S	g
11	3	r	←	u	s
12	0	S	←	e	f
13	1	S	←	S	k
14	2	S	←	S	d
15	10	S	←	S	p
16	13	S	←	i	j
17	14	S	←	i	c
18	15	S	←	i	o
19	11	4	←	0	r

שאלה 8

2. PPM –

הנח' כי $k_{max}=3$ א. מלא/י את הטבלה הבאה עבור ההודעה `T = misses_miss_missouri`.

K=-1	K=0		K=1		K=2		K=3	
	context	counts	context	counts	context	counts	context	counts

ב. בהמשך לאותה ההודעה ואותו k_{max} , מהם התווים המועברים למפענח על מנת לקודד את ההודעה ובאיזה הקשר הם מקודדים (לאיזה k התו מתייחס). כלומר, מלא/י את הטבלה הבאה (יש להוסיף שורות לטבלה במידת הצורך):

Encoded symbol	Order

ג. האם קיים תו בהודעה T הגורם להפעלת "עקרון ההרחקה" – Exclusion Principle? אם כן, מהו התו, ומהן ההסתברויות לפני ואחרי הפעלת העקרון? אם לא, הסבר מדוע עקרון זה אינו מופעל בהודעה זו.

פתרון 8

misses - miss - missouri

le ②

k=-1	k=0		k=1		k=2		k=3	
	context	counts	context	counts	context	counts	context	counts
	M=3	M	i=3 B=1	Mi	S=3 B=1	Mis	S=3 B=1	
	i=4	i	S=3 B=1	is	S=3 B=1	iss	e=1 - =1 O=1 B=3	
	S=7	S	S=3 e=1 - =2 O=1 B=4	ss	e=1 - =1 O=1 B=3	sse	S=1 B=1	
	e=1	e	S=1 B=1	se	S=1 B=1	ses	- =1 B=1	
	- =2	-	M=2 B=1	es	- =1 B=1	es-	M=1 B=1	
	O=1	O	U=1 B=1	S-	M=2 B=1	S-M	i=2 B=1	
	U=1	U	r=1 B=1	-M	i=2 B=1	-Mi	S=2 B=1	
	r=1	r	i=1 B=1	So	U=1 B=1	SS-	M=1 B=1	
	B=8			0U	r=1 B=1	SSO	U=1 B=1	
				Ur	i=1 B=1	SOU	r=1 B=1	
						OUR	i=1 B=1	

①		②		③		④		⑤		⑥		Σ	
ES	Order	\$		\$		-	-1	\$		\$		r	-1
\$	3	\$	3	\$	1	\$	-1	\$	3	\$	0	\$	3
\$	2	\$	2	\$	0	\$	3	\$	2	\$	-1	\$	2
\$	1	\$	1	e	-1	\$	2	-	1	\$	3	\$	1
\$	0	\$	0	\$	3	\$	1	\$	3	\$	2	\$	0
M	-1	S	-1	\$	2	M	0	M	2	\$	1	\$	0
\$	3	\$	3	\$	1	\$	3	i	3	\$	0	U	-1
\$	2	\$	2	\$	0	\$	2	S	3	\$	-1	\$	3
\$	1	\$	1	\$	3	i	1	S	3	\$	3	\$	2
\$	0	S	0	\$	2	\$	3	\$	3	\$	2	\$	1
i	-1	\$	3	\$	1	S	2	\$	2	\$	1	\$	0
		\$	2	\$	0	S	3	\$	1	\$	0		

בגד

[misses - miss - miss ...]

ע כן, הו 0.

נחשב בטבלה רק עם המשתנים שונים (לפני פשיטה ב 0)

k=0	k=1	k=2	k=3
m=3 i=3 s=7 e=1 - = 2	s=3 e=1 - = 2 0 = 1	ss e=1 - = 1 0 = 1	iss e=1 - = 1 0 = 1
\$ = 5 → 6	\$ = 3 → 4	\$ = 2 → 3	\$ = 2 → 3

(השינויים לאחר הוספת 0 לטבלה)

כאן כדי לקודד 0 נשים אם כי לא קיים לנו iss בטבלה, גבו שאלו קיים
 0 בהקשר 3-א ובדי לרצה בהקשר נשמה \$ כנף לאבי הקשרים 2, 1, 2
 ולכן ~~הקשרים~~ הקידוד של 0 יראה כך: \$ \$ \$ 0
 כזה נחשב את ההסתברות:

$$P = \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{3} \cdot \frac{1}{2} \cdot \frac{1}{4} = 0.00283$$

$$EP = \frac{1}{2} \cdot \frac{1}{4} \cdot \frac{1}{4} \cdot \frac{1}{2} = 0.00625$$

ה- p מחושב כך:

מספר ~~המקרים~~ שאנו רוצים לשלוח חלקי סה"כ מספר המופעים של האותיות (באותה שורה וצמודה)
 וצוד כמות ה- \$. (שכיחות חלקי סכום שלכוחות)

ה- p מחושב כך:

מספר ~~המקרים~~ שאנו רוצים לשלוח חלקי סה"כ מספר המופעים של האותיות (באותה שורה וצמודה)
 וצוד כמות ה- \$. (שכיחות חלקי סכום שלכוחות)

אם נסתכל על ה \$ הרביעי * מדובר פה על המקרה בו ההקשר הוא 0
 ולכן יש לנו \$ = 5 וסכום השכיחות הוא 21

$$\frac{3}{3} + \frac{3}{3} + \frac{7}{5} + \frac{1}{5} + \frac{3}{5} + \frac{5}{5} = 21$$

הקשר

הסתברות צמוד \$ זה היא $\frac{5}{21}$

ה- EP מחושב כך:

בהקשר באורך 2 ניתן להוידא את e, - אחרת היינו יכולים להשתמש בהקשר
 3 אך אחרת כי זה לא אפשרי ולכן נחשב את ההסתברות ללא הלייפ
 הללו ולכן בהקשר 2 נשאר לנו רק \$ ולכן ההסתברות היא 1.
 וכן העאור לאבי שאר ההקשרים.

שאלה 9

קודד בשיטת הדחיסה *Binary Run Length* את המחרוזות הבינאריות הבאות כאשר גודל הבלוק $k = 3$.

א. 0 1 1 1 1 0 0 1 1 1 0 0 0 0 0 0 0 1

ב. 0 1 1 1 1 0 0 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0

פתרון א'

נתון לנו בלוק מקסימלי $k = 3$, כל בלוק יכול להחזיק עד: $2^k - 1 = 8 - 1 = 7$ רצפים.

קודדו את המחרוזת הבאה: 0 1 1 1 1 0 0 1 1 1 0 0 0 0 0 0 0 1

↓

1, 4, 2, 3, 8, 1

נשים לב שאנחנו בבעיה כי גודל כל בלוק הוא לכל היותר 3 סיביות (עד המספר 7) אבל יש לנו כאן 8.

לכן, עבור נסתכל על המקרה הכי גדול $2^3 - 1 = 7$ כלומר 111 ואז נחשב $8 - 7 = 1$ כלומר 001.
לכן סה"כ נקודד 111 001.

↓

001 100 010 011 111 001 001

פתרון ב'

נתון לנו בלוק מקסימלי $k = 3$, כל בלוק יכול להחזיק עד: $2^k - 1 = 8 - 1 = 7$ רצפים.

קודדו את המחרוזת הבאה: 0 1 1 1 1 0 0 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0

↓

1, 4, 2, 3, 7, 8, 1

נשים לב שאנחנו בבעיה כי גודל כל בלוק הוא לכל היותר 3 סיביות (עד המספר 7) אבל יש לנו כאן 8.

בנוסף, המספר 7 כאן ותופס את המקום של ה-Escape code, כלומר האלגוריתם יחשוב שאנחנו קולטים מספר גדול מהבלוק כי עשינו 111 אבל בעצם בסה"כ רצינו לקודד את המספר 7.

לכן, עבור הספרה 8 נקודד 111 001 כלומר ה-Escape code וגם 001 $\Rightarrow 8 - 7 = 1$

ועבור הספרה 7 נקודד 111 000 כלומר "נקודד 7 ככה 111" ובשביל להגיד שזה לא אקסייפ אז $7 - 7 = 0$

↓

001 100 010 011 111000 111001 001

חלאס.