תרגיל בית 06 אביתר שמש 322623182

שאלה 1 סעיף א 1)

$V = \{S,V\}$

$\sum = \{0,1,2\}$

R = {S->V,V->1|V->0V22}

שאלה 1 סעיף א 2)

$V = \{S,V\}$

$\sum = \{x,y,z,*,+,(,),\varepsilon\}$

R = {S->V|S->ε|V->V*V|V->(V*V)|V->V+V,V->(V+V)|V->x,V->y|V->z}

שאלה 1 סעיף א 3)

$V = \{S_1,S_2\}$

$\sum = \sum_1 U \sum_2$

$R = R_1 U R_2 U \{S\text{->}S_1|S\text{->}S_2\}$

שאלה 1 סעיף א 4)

$V = \{S,V\}$

$\sum = \{0,1, \varepsilon \}$

R = {S->ε|S->V|V->0V1|V->1V0,V->V01|V->V10|V->01V|V->10V}

שאלה 1 סעיף ב 2)

סיבוכיות הזיכרון של הפונקציה generate_language במקרה הגרוע היא: אקספוננציאלי במקרה הגרוע.

כמו שאראה בסעיף ג', כמות הקריאות הרקורסיביות היא אקספוננציאלית באורך הקלט.

אפילו במקרה הטוב בו סיבוכיות הזיכרון של הפונקציה היא O(1) לכל צומת בעץ, סיבוכיות הזיכרון הכוללת היא אקספוננציאלית באורך הקלט.

שאלה 1 סעיף ב 3)

סיבוכיות הזמן של הפונקציה generate_language במקרה הגרוע היא: אקספוננציאלי במקרה הגרוע.

הקוד עובר על כל החלוקות האפשריות של מילה באורך k, ושולח על כל אורך שכזה 2 קריאות רקורסיביות, שתנאי העצירה הוא כאשר האורך הנשלח הוא 1. לכן, לכל קריאה רקורסיבית, יהיו (k-n)! קריאות ש n הוא האורך הנשלח בקריאה הראשונה. לכן, סיבוכיות זמן הריצה היא אקספוננציאלית באורך הקלט.

שאלה 1 סעיף ג 1)

מספר הדרכים להגיע למחרוזות באורך k שניתן ליצור ע"י הפרמטרים הנתונים, כולל חזרות

שאלה 1 סעיף ג 2)

לכל היותר לינארית ב-K. אורך המסלול בעל כמות הקריאות הרקורסיביות הארוך ביותר אורכו k-1.

בכל קריאה, הזיכרון גדל ב- O(1) ולכן בסה"כ O(k).

שאלה 1 סעיף ג 3)

לכל היותר פולינומיאלית ב-K.

יש k-1 קריאות רקורסיביות, בכל אחת מהן O(1) עבודה.

לכן:

$$\sum_{i=1}^{k} k-i=\sum_{i=1}^{k} k-\sum_{i=1}^{k} i=k^2-\frac{k(k-1)}{2}=\frac{k^2+k}{2}=O(k^2)$$

שאלה 3 סעיף א)

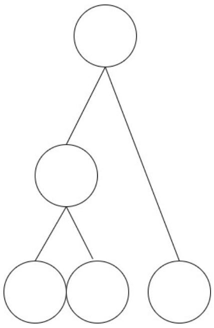
מספר העלים בעץ H – t. בסופו של העץ כל עלה הוא אות מהקורפוס. לכן, עבור קורפוס בעל t תווים, מספר העלים בעץ היא t.

משקל השורש של העץ H – n. השורש של העץ מכיל את כל האותיות בקורפוס, והערך שלו הוא סכום של כמות הפעמים שכל אחד מהם מופיע. לכן, משקל השורש של העץ H הוא כאורך הקורפוס, n.

גובה העץ H – בין $\lceil \log(t) \rceil$ ל t-1

כאשר כל תו ב-corpus מופיע כמות שווה של פעמים לכל השאר נקבל עץ מאוזן(לכל כמות צמתים שהיא חזקה של 2), כאשר $\log(t)$ אינו מספר שלם, אנחנו מוסיפים צמתים מעבר לעץ המאוזן אך לא מגדילים את הגובה ביותר מאחד. לכן, $\lceil \log(t) \rceil$.

במקרה בו נסדר את כמות הפעמים שכל תו מופיע מהקטן לגדול, אם כל תו מופיע יותר פעמים מסכום הפעמים שכל התווים לפניו מופיעים, נקבל עץ כזה:



גובהו הוא t-1, וזה המקרה המקסימלי, כי לא יכול להיות עץ בגובה t.

מספר הצמתים בעץ H – 2t-1. בהתחלה מאחדים 2 צמתים לצומת אחת.

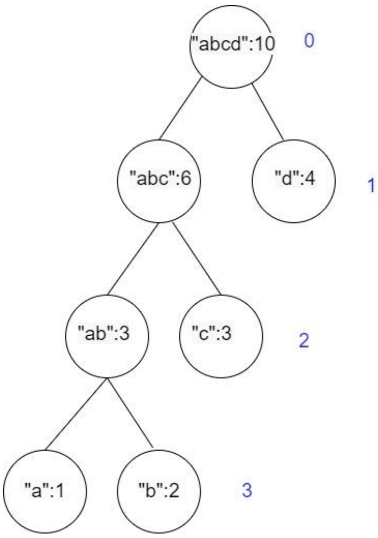
כעת בעץ יש t+1 צמתים. נאחד עם צומת נוסף, וכעת t+2.

לכן, בכל איחוד של 2 צמתים, תתווסף צומת אחת לכמות הצמתים הכוללת.

נמשיך לאחד עם t-3 הצמתים שנותרו, ולבסוף נקבל: t+2+t-3 = 2t-1.

לא הנחנו שום הנחה על העץ מלבד כמות הצמתים ההתחלתית שלו, ולכן, לכל עץ עם t תווים, יש 2t-1 צמתים.

שאלה 3 סעיף ב 1) הפרכה



corpus – "abbcccdddd"

char_count – {"a":1,"b":2,"c":3,"d":4}

העץ שיבנה + משקליו.

כמו שניתן לראות, המשקל של רמה 0 ורמה 1 שווה, אך הוא לא שווה למשקל של רמה 2 או רמה 3.

לכן, הופרך ע"י דוגמה נגדית

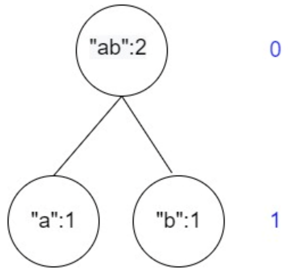
שאלה 3 סעיף ב 1) הפרכה

Corpus – "ab"

char_count – {"a":1,"b":1}

העץ שיבנה + משקליו.

לכן, כל רמה בעץ בעלת משקל שווה, ואין שתי רמות שונות שמשקלן שונה.



שאלה 3 סעיף ג 1)

סיבוכיות הזמן הממוצעת של char_count(corpus) היא O(n) גם כאשר |∑| אינו O(1).

עבור גודל הודעה מסויים, כמות האיטרציות שלנו תהיה n, והכנסה למילון היא O(1), לכן, O(n).

שאלה 3 סעיף ג 2)

סיבוכיות הזמן הממוצעת של build_huffman_tree(char_count_dict) מכילה מילון "אינסופי", לכן, בניגוד למקרה בו הנחנו ו |∑| = O(1), ואז הסיבוכיות הכוללת היא O(1), כאן הסיבוכיות לכל איטרציה תהיה שווה ל |∑|, וכמות האיטרציות תהיה כ |∑|. לכן $O(\sum^2)$.

שאלה 3 סעיף ג 3)

סיבוכיות הזמן הממוצעת של generate_hcode(htree,prefix ="") היא $|\sum|^2$. יש לנו במקרה הגרוע

$\sum^2$ עלים, ובכל צומת מבצעים O(1) עבודה, לכן, $O(\sum^2)$.

שאלה 3 סעיף ג 4)

סיבוכיות הזמן הממוצעת של compress(text,hcode) היא O(m) גם כאשר |∑| אינו O(1).

אנחנו עוברים על כל תו בהודעה, ושולפים מהמילון מה הערך האפמן עבורו. לכן O(1) עבודה בכל צומת.

סה"כ m צמתים, לכן O(m).

שאלה 3 סעיף ג 5)

סיבוכיות הזמן הממוצעת של decompress(text,hcode) כאשר |∑| אינו O(1) היא O(b*∑)

על כל ביט במחרוזת שלנו בביטים(שאורכה b) אנחנו מבצעים O(∑) עבודה כדי להגיע לעלה הרלוונטי, ולכן O(b*∑).

שאלה 4 סעיף א)

מחרוזת עבורה שתי הפונקציות ייתנו פלט זהה:

abcabcabcd

הפלט שיודפס:

['a', 'b', 'c', [3, 6], 'd']

שאלה 4 סעיף ב)

מחרוזת שהפונקציה החדשה דוחסת טוב יותר מדחיסת LZW:

abbabbbaba

אורך הדחיסה עבור האלגוריתם החדש: 66

['a', 'a', 'c', 'a', 'a', [5, 4], 'c']

אורך הדחיסה עבור האלגוריתם הסטנדרטי: 68

['a', 'a', 'c', 'a', [1, 3], 'c', 'a', 'c']

שאלה 4 סעיף ג)

לא קיימת מחרוזת שכזו.

האלגוריתם החדש, בעצם מחשב באופן רקורסיבי את הדחיסה היעילה ביותר מבחינת דחיסה.

לכן, הוא האלגוריתם שמבחינת אורך דחיסה, הוא האופטימלי ביותר.

אבל, הסיבה שלא משתמשים בו, היא שכמות הקריאות הרקורסיביות פר תו במחרוזת הוא ענק, בניגוד ל LZW, בו פשוט ברגע שמתחילים לראות קריאה, ממשיכים איתה עד שיש הפרה. אז אמנם, הוא יעיל יותר מבחינת זיכרון, אבל מבחינת זמן ריצה הוא בזבזני בהרבה.

שאלה 5 סעיף א 1)

הטענה נכונה.

המרחק ⌊(d-1)/2⌋ אומר שכל שגיאה בטווח זה, נוכל לזהות ולתקן.

לכן, אם ניקח איבר בטווח, אם הוא לא שייך לתמונה אז בחיתוך יהיו 0 איברים, בעוד אם ניקח איבר ששייך לתמונה יהיה איבר אחד בחיתוך.

לכן, יש לכל היותר איבר אחד והטענה נכונה.

שאלה 5 סעיף א 2)

הטענה נכונה.

אם לוקחים איבר בתמונה, מהגדרת d, אם ניקח רדיוס שקטן מ- d, יהיה רק כדור אחד בטווח זה, ולא יכולים להיות 0 איברים כי לקחנו איבר בתמונה.

לכן, הטענה נכונה.

שאלה 5 סעיף א 3)

הטענה לא נכונה.

אם ניקח פונקצייה חח"ע כאשר k=2,n=8(לדוג' הוספת כל תו בקוד שלוש פעמים נוספות), אז אם ניקח איבר בטווח, לא בהכרח יהיה חיתוך עם התמונה של c(2^2 איברים בתמונה, לעומת 2^8 איברים בטווח, ולכן גודל החיתוך יכול להיות קטן שווה ל-1, והטענה אינה נכונה.

שאלה 5 סעיף א 4)

הטענה לא נכונה.

נפריך באמצעות דוגמה נגדית.

.$k=2,n=3$

נגדיר:

000<-00

011<-01

101<-10

110<-11

הפונקציה חח"ע, והמרחק בין כל שני איברים בתמונה היא 2, לכן מרחק האמינג הוא 2, לכן הטענה לא נכונה.

שאלה 5 סעיף א 5)

הטענה נכונה.

מכך שהפונקציה חח"ע ו- $k=n$, הפונקציה היא זיווג(חח"ע ועל). לכן, ההפרש בין כל שני איברים בתמונה הוא לכל הפחות 1(לא יכול להיות 0 אחרת הם שווים), ויכול להיות 1 אם מחליפים ביט ספציפי מ -0 ל-1 או להיפך, ולכן המרחק הוא בדיוק 1.

שאלה 5 סעיף ב 1)

פונקציית הפענוח D תמיר את התו הראשון למחרוזת באורך 2 לפי בסיס בינארי, ותשמיט את הספרה האחרונה(תו הזוגיות).

הקוד טריוויאלי.

$C(101) = 100$

$C(100) = 101$

מרחקן הוא 1.

שאלה 5 סעיף ב 2)

פונקציית הפענוח D תמיר את התו הראשון למחרוזת באורך 2 לפי בסיס בינארי, ותשמיט את הספרה האחרונה(תו הזוגיות).

הקוד טריוויאלי.

$C(101) = 100$

$C(100) = 101$

מרחקן הוא 1.