תרגיל בית 5 – מבוא מורחב למדעי־המחשב

שחר פרץ

15 באוגוסט 2024

(1) ....

(2) ....

ג. נניח בשלילה קיום גנרטור g המייצא את קבוצת איברי g הגדולים מ־0; אזי לכל f גנרטור, $g(f)$ מוגדר היטב. בפרט, עבור $f = \lambda n \in \mathbb{N}.n+1$. אך, $\mathrm{Im}(f) = \mathbb{N}, \forall n \in \mathbb{N}.n > 0$, כלומר $g(f) = \mathbb{N}$ וסה"כ הפלט אינסופי ובפרט הגנרטור חסר השהייה סופית – סתירה לקיומו.

ה. יהי גנרטור $G(g_1, g_2)$ המקבל שני גנרטורים ומחזיר את החיתוך שלהם; נתבונן ב־$g_1 = \lambda n \in \mathbb{N}.0,\ \ g_2 = \lambda n \in \mathbb{N}.1$. ידוע לנו שהחיתוך ריק, אך האלגוריתם יאלץ לעבור על כל g_1 ו־g_2 על מנת לייצר את החיתוך, כלומר לעבור כל כמות בת־מנייה של מספרים, ובפרט אין־סופית. אזי, לא קיים גנרטור בעל השהייה סופית מתאים, כדרוש. נשים לב שההוכחה נכונה ללא תלות בנתון ש־g_1, g_2 בעלי השהייה סופית, שכן תחת ההנחה שקיימת השהייה, מעבר על $\aleph_0$ איברים יארך זמן אינסופי.

ו. נניח בשלילה קיום גנרטור בעל השהייה סופית G המקבל שני גנרטורים g_1, g_2 המייצרים את הסדרות $\{a_i\}_{i=1}^n, \{b_i\}_{i=1}^n$, ומייצר את סדרת האיברים המקיימים $a_i \neq b_i$. נבחר $a_1 = a_2 = \cdots = a_n = 0,\ \ b_1 = \cdots = b_n = 1$, לכן $\forall i \in \mathbb{N}.a_i = 0 \neq 1 = b_i$. בפרט, G מייצר את $\varnothing$. גם נבחר $n = \infty$, כי נתון בשאלה ששניהם סופיים באותו הגודל, או ששניהם אין־סופיים. אזי, הגנרטור G יאלץ לעבור כל $\{a_n\}$ ו־$\{b_n\}$ כדי להסיק זאת, ומשום ש־$n = \infty$, פעולה זו תיקח זמן אין־סופי – ובפרט, G אינו בעל השהייה סופית. סתירה.

(3) ....

ב. • צ.ל. קיום עץ בינארי מושלם $T = \langle V, E\rangle$ כאשר $|V| = n$ אמ"מ $\exists d \geq 1.n = 2^d - 1$.

הוכחה.

$\Longrightarrow$ נניח קיום $d \geq 1$ כך ש־$n = 2^d - 1$, ונוכיח קיום עץ $T = \langle V, E\rangle$ כך ש־$|V| = n$. נבחר, $V = \mathbb{N} \cap [1, n]$, ונבנה את E כך שה־root מתחבר לשני ילדיו, כל אחד מהילדים מתחבר לשני ילדים נוספים, וכן הלאה עד אשר מגיעים לילד ה־d־ים, שם עוצרים. סה"כ בכל "שכבה" של ילדים יהיו 2^i צמתים (כי כל שכבה תתפצל ל־2 בהבאה), וסה"כ נגיע במדויק למספר הצמתים שהצבנו לנו:

$$\sum_{i=1}^{d} 2^i = 2^d - 1 = n = |\mathbb{N} \cap [1, n]| = |V|$$

הערה: בנייה מפורשת ופורמלית של E אפשרית אך מיותרת בקונטסט זה. בנייה כזו נבנת כחלק מסעיף ג' בשאלה זו, אך ע"י אלגוריתם.

$\Longleftarrow$ נניח קיום $d \in \mathbb{N}$ כך ש־$n = 2^d - 1$, נוכיח באינדוקציה על d קיום עץ מושלם.

* בסיס: $d = 1$ אזי $n = 2^1 - 1 = 1$ מושלם באופן ריק;
* צעד: נניח על d קיום n עץ מושלם, בעבור $d' := d + 1$ יוצב $n' := 2^{d+1} = 2^d + (2^d - 1)$, ונוכיח קיום עץ מושלם בעבור n'. ידוע על קיום עת בגודל $n = 2^d - 1$ מושלם מה.א., בהתאם לאינדוקציה מהסעיף הקודם יש לו 2^{n-1} עלים, נבחר עץ עבורו לכל אחד מהעלים יש שני ילדים העונה על התנאים של עץ מושלם (בהתאם להגדרת מסלול ומה.א.) וסה"כ 2^n עלים כלומר ובהתאם כלומר $n' = n + 1$ וסיימנו.

סה"כ הוכחנו את שתי הגרירות, כדרוש.

■

• עתה, יש להוכיח יחידות עץ זה.

הוכחה. יהי $n \in \mathbb{N}$, מהטענות לעיל קיים עץ בעל $2^n - 1$ צמתים, נסמן את המסלול הארוך ביותר לכל העלים ב־l. נניח בשלילה קיום עץ שונה ממנו, שנוכל לבטא כשינוי על העץ הקיים שלנו. נניח בשלילה שאפשר להזיז מספר אי זוגי של צמתים ונמצא כי יש עלה בעל ילד יחיד וזו סתירה, לכן נוכל להזיז מספר זוגי של צמתים בלבד. נזיז $2i$ צמתים, אם חזרנו לנקודת ההתחלה, סיימנו,

אם לאו, אזי קיימת לפחות צומת אחת בעלת 2 עלים חדשים, ומכאן שהמסלול אליה הוא $l+1$, ובאינדוקציה (לא אפרטה כי זה יחסית ברור מאליו) בהכרח קיימת צומת בעלת מסלול קצר יותר (ממנה "הבאנו את העלים") וזו סתירה. ■

ג. ננתח את סיבוכיות זמן הריצה של build_balanced. בתוכה, יש לולאה העוברת על n ה"שכבות" בעץ. כלומר, i משתנה החזרה בין $1 \le i \le n$. כתלות ב־i, הלולאה בפנים תבצע 2^i קריאות ל־BinarySearchTree.insert כאשר זאת יקח סיבוכיות של i, כמות שכבות שכבר נוספה. סה"כ, נבצע $\sum_{i=1}^{n} 2^i \cdot i = \frac{n(n+1)(n+2)}{6} = \boldsymbol{\Theta(n^3)}$ פעולות.

ד. נתח את סיבוכיות הפונקציה subtree_sum. הפונקציה עוברת רקורסיבית מה־root לשני הצמתים אליה היא מחוברת, ומשום שאין מעגל בעץ בינארי לפי הגדרת עץ, אז נעבור על לא יותר מעל הצמתים בגרף. בגלל שיש קשר בין כל שני צמתים, אז נעבור לכל היותר על כל הצמתים בגרף. סה"כ נעבור בדיוק על כל הצמתים בגרף, ולא נבצע שום דבר פרט לקיראות רקורסיביות בסיבוכיות שאינה $O(1)$. משום שיש n צמתים, סה"כ הגענו לסיבוכיות של $\boldsymbol{\Theta(n)}$.

(4) ....

ב. יהי n אורך רשימה ו־k אורך רישא וסיפא אותו נשווה. ננתח בעבור המקרה הגרוע, שיתרחש כאשר תמיד ה־suffix־ים וה־prefix־ים יהיו זהים (כי כך השוואת הרשימות מעולם לא תגיע לתנאי העצירה שלה, בו מתקבלים שני תווים שונים) כלומר $\forall i, j. s_i[: k] = s_j[-k :]$. במקרה הזה, ניכנס ללולאה שתעשה n איטרציות, חיתוך ב־$O(k)$, ועוד n איטרציות שיכללו חיתוך במחיר $O(k)$. סה"כ $O(n(k+nk)) = O(nk+n^2k) = \boldsymbol{O(n^2k)}$.

ד. נניח שאין שתי מחרוזות עם אותו ה־prefix או ה־suffix, או שאלו שווים בין שתי מחרוזות, ונניח שהשוואת מחרוזות וחישוב hash מתבצע ב־$O(k)$ כאשר k אורך המחרוזת. ראשית נכנס ללולאה שתבצע חיתוך ו־insert, שניהם ב־$O(k)$, ונקבל מכאן $O(2nk)$. בלולאה הבאה נחשב חיתוך ב־$O(k)$ ו־hash כחלק מפעולת find, שתיארך $O(k)$ גם היא. סה"כ גם כאן $O(2nk)$, ונקבל $O(4nk) = \boldsymbol{O(nk)}$.

(5) ....

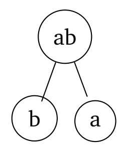
א. i **מספר העלים בעץ H:** ישנו עלה לכל תו: נניח בשלילה שלא כן, נפלג למקרים. אם מספר העלים גדול ממש ממספר התווים, אזי קיים עלה הנוצר מחיבור שני תווים (זו הדרך השנייה ליצור צומת לפי הגדרת עץ הפאמן) וזו סתירה לכך שהוא לא עלה. אחרת קיימים פחות תווים מעלים, כלומר קיים איזשהו תו שאינו עלה ומכאן שאינו בעץ, וזו סתירה. סה"כ התשובה היא $\boldsymbol{c = t}$.

ii **משקל השורש של העץ H:** משקל השורש הוא (אפשר להוכיח באינדוקציה) חיבור ערכי כל האותיות בא"ב. כל אות נוספת בקורפוס תוביל להעלאת ערך אות כלשהי ב־1, ולכן התשובה היא $\boldsymbol{c = n}$.

iii **גובה העץ H:** כפי שהוכח כאשר דיברנו על עצי חיפוש בינאריים, האורך המינימלי הוא בעבור $\log t + 1$ (כאשר העץ מאוזן, כמו בסעיף 1(ג)). עבור חסם עליון, העץ המקסימלי האפשרי אורכו t לפי הגדרה (עבור צורה כמו בסעיף 1(א)), והוכחנו את קיומו אם נבחר את הקורפוס להיות מספרי פיבונאצ'י. סה"כ $\boldsymbol{\log t + 1 \le c \le t}$.

iv **מספר הצמתים בעץ H:** טענה: $\boldsymbol{c = 2t - 1}$.

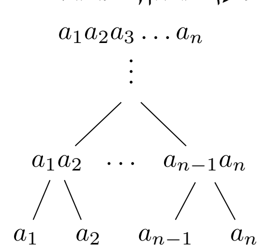
הוכחה. נוכיח באינדוקציה. בסיס, $t = 2$, העץ היחיד שיתכן הוא:



וסה"כ $c = 3 = 2 \cdot 2 - 1$ כדרוש. צעד: נניח באינדוקציה מלאה נכונות לעץ ל־$k < t$ תווים, ונוכיח בעבור עץ ל־t תווים. משום ש־$t \ge 3$ אזי נוכל לפלגו לשני תתים עצים לא ריקים עבור מספר תווים קטן ממש מ־t, שנסמן כי יש להם c_1, c_2 תווים בהתאמה. סה"כ נחברם בצומת כמותאר בעץ האפמן (שהוא עץ בינארי), ונקבל $c = c_1 + c_2 + 1$ תווים. נסמן את כמות התווים בהם k_1, k_2, לפיכך $k_1 + k_2 = t$ וגם $c_1 = 2k_1 - 1, c_2 = 2k_2 - 1$ וסה"כ $c = 2(k_1 + k_2 - 1) + 1 = 2t - 1$ כדרוש. ■

ב. $\boldsymbol{\Delta := ||C(a_n)| - |(C(a_1))|| = 0}$.

הוכחה. כדי להוכיח את הטענה, נוכיח שהעץ בצורה של עץ מאוזן, כלומר:



כבסיס, משום ש־$2|256$, אז נחלקם מהסדר $a_1 < \cdots < a_n$ הנתון בצורה של $\{a_1, a_2\}, \{a_3, a_4\}, \dots, \{a_{n-1}, a_n\}$. זאת, כי בשום שלב לא יתכן שאיבר יחיד a_i יהיה המינימום, כי לשם כך קיימים a_k, a_{k+1} כך ש־$a_k + a_{k+1} < a_i$ אך ממונוטוניות $a_1 \le a_k, a_2 \le a_{k+1}$,

ולכן:

$$a_n < 2a_1 < a_1 + a_2 \leq a_k + a_{k+1} \leq a_i < a_n$$

וסה"כ $a_n < a_n$ וזו סתירה. באינדוקציה, גם בעבור $a_i, a_{i+1}, a_{i+2}, a_{i+3}$ יצתמדו עם $a_{i+4}, \ldots a_{i+7}$, ובכלליות $a_i \ldots a_{i+j}$ מוצמד עם a_{i+j+1}, a_{2j}, כאשר $i \mid j$ ו־$\log j \in \mathbb{N}$, שני התנאים אפשריים כי $\log 256 \in \mathbb{N}$. סה"כ מהתנאים האלו קיבלנו עץ מאוזן.

משום שהוכח כי העץ עץ מאוזן, לפי הגדרה, אורך המסלול בין כל אחד מהעלים לשורש זהה, ובמקרה הזה, הוא $\log 256$. בפרט בעבור a_n, a_1. אורך קידוד האפמן הוא אותו האורך, ולכן $\ell = u = \log 256$, נעביר אגפים ונקבל $\boldsymbol{\ell - u = 0}$. ■

ג. 1. יהי c קוד חח"ע שאינו prefix-free ואינו uniquely-decodeable. יהיו שתי מחרוזות `text1, text2`. טענה שגוייה היא כי הדחיסה תניב בהכרח שתי מחרוזות בינאריות שונות.

הוכחה. נוכיח ע"י נתינת דוגמה נגדית. נבחר C קוד על $\{a, b, c\}$, המוגדר לפי $C(a) = 01, C(b) = 0, C(c) = 001$. נראה כי הוא מקיים את תנאי הרישא של הגרירה. בבירור, הקוד חח"ע. הוא גם אינו prefix-free כי `C(a) == C(b) + "1"`. הוא גם אינו uniquely-decodeable כי עבור המחרוזת `001` ניתן יהיה לפרשה הן כ־`ba` והן כ־`c`. עתה נראה כי הדחיסה עלולה לפלוט תוצאות זהות עבור זוג מחרוזות שונה. נבחר `text1 = ba, text2 = c`. אין חזרות ולכן דחיסת LZW לא תבוא לידי ביטוי. סה"כ נקבל את המחרוזות בינאריות $0001, 0001$ שהן זהות, וזו סתירה לטענה. ■

2. יהי c קוד חח"ע שאינו prefix-free אך uniquely-decodeable. יהיו שתי מחרוזות `text1, text2`.

הוכחה. נוכיח כי הצפנתן שונה. המחרוזות שונות, ולכן קיים אינקס i ב־`text1[i] != text2[i]` כלשהו שונה בניהן. נפלג למקרים.

■