בס"ד, תשרי תשע"ו

מגיש :יוחנן חאיק

תרגיל מס' 6 בניתוח אלג' – מושגים בסיסיים ב- NP

1) (א) הוכיחו כי אם $L1\in P$ ו- $L2\in P$ אזי $L1\cup L2\in P$.
(ב) מהו סדר גודל זמן הריצה הנחוץ לאלג' A3 שפותר בעיה זו אם ידוע כי זמן הריצה של A1 הפותר את L1 הינו $O(n^{k1})$ וזמן הריצה של A2 הפותר את L2 הינו $O(n^{k2})$? הסבירו!

א. נתון לנו כי $L_1 \in P$, כלומר על פי ההגדרה, קיים אלגוריתם פולינומי A_1 המכריע את השפה L_1.
כלומר לכל מילה $x\in \{0, 1\}^*$, A_1 יכול להגיד האם $x \in L_1$ וכל זה תחת זמן פולינומיאלי.
ואותו כנ"ל – $L_2 \in P$, כלומר על פי ההגדרה, קיים אלגוריתם פולינומי A_2 המכריע את השפה L_2.
כלומר לכל מילה $x\in \{0, 1\}^*$, A_1 יכול להגיד האם $x \in L_2$ וכל זה תחת זמן פולינומיאלי.
. כעת נגדיר שפה חדשה $L_1 \cup L_2$ = **L_3**.
בעבור השפה החדשה, נגדיר אלגוריתם חדש A_3, שעובד באופן הבא:
בהינתן מילה $x \in \{0,1\}]^*$
1. A_3 מריץ את אלגוריתם A_1 על x.
אם A_1 מחזיר 1, נמשיך ל-A_2.
2. אחרת, נחזיר 0 ונסיים.
3. נריץ את A_2 על x.
אם A_2 מחזיר 1, נחזיר 1 ונסיים.
3. אחרת – נחזיר 0 ונסיים.
2. **הובחת נכונות** – נניח ש $x \in L_3$, כלומר $x \in L_1 \cap L_2$, כלומר $x \in L_1$ וגם $x\in L_2$.
אם כך, יוצא לנו שאם $x \in L_1$ אזי בשורה הראשונה באלגוריתם, אנחנו נחזיר 1 (אמת) ונמשיך הלאה. ורק אם $x \in L_2$ אז בשורה השניה באלגוריתם, אנחנו נחזיר 1 ונסיים.
עכשיו נראה מה קורה אם $x \notin L_3$. במקרה כזה, מובן לנו שהוא גם לא משתייך לחיתוך השפות, או אפילו לא לשתיהן. במקרה זה כבר בשורה הראשונה נקבל ערך 0 ונצא.
3. **זמני ריצה** – נתון לנו שזמני הריצה של שתי השפות L_1, L_2 הם פולינומיאליות ושייכות למחלקה P. כך שגם אם נצטרך להריץ את שני האלגוריתמים אחד אחרי השני, זה יצא לנו זמן פולינומי בפול 2. מה עדיין מגדיר לנו את זה בזמן פולינומיאלי, מש"ל.

ב. במקרה הגרוע/ הטוב (תלוי בנקודת המבט), נצטרך להריץ את שני האלגוריתמים אחד אחרי השני, מה שייצור לנו בעצם חיבור של שני זמני הריצה – $O(n^{k_1}+n^{k_2})$ כאשר מדובר בחיבור ולא בגדול מביניהם.

2) (א) אם $L\in P$ הוכיחו כי $L^K\in P$ עבור K שלם אי-שלילי כלשהו.
(ב) מהו סדר גודל זמן הריצה הנחוץ לאלג' A2 שפותר בעיה זו אם ידוע כי זמן הריצה של A1 הפותר את L הינו $O(n^c)$? הסבירו!

(ג) נתון $L \in P$. הציעו אלגוריתם פולינומי שפותר את L^* וחשבו את זמן הריצה שלו אם זמן הריצה של האלגוריתם שפותר את L הוא הינו $O(n^c)$.

א. עלינו בעצם להוכיח סגירות לשרשור עבור השפה L.

עבור כל מילה $x \in \{0,1\}^*$, נתון לנו שהשפה L מכריעה אותה בזמן-פולינומי ביחס לאורך הקלט. מאחר ש $|x|$ הוא מספר סופי, כל הכפלה/שרשור k פעמים עדיין משאיר את זמן הריצה ביחס פולינומיאלי מאחר, האלגוריתם פשוט יצטרך לעשות את הפעולה שוב ושוב, כל שהזמן יהיה הכפלה של k ביחס לביצוע המקורי.

ב. $O(kn^c)$ – מאחר שנשארנו תחת זמן-פולינומיאלי, אז יש לנו את זמן הפתרון המקורי שעובד פעם אחר פעם, kפעמים ולכן מכפיל את זמן הריצה.

ג. נתנונה לנו מילה $w=\{x_1...x_k\}$, השייכת ל-L^*.וכמו כן נתון לנו אלגוריתם A הפותר את $x \in L$ בזמן פולינומיאלי.

עבור כל $x_i \in w$, נעבוד בצורה הבאה:

אם $x_i = \varepsilon$, אדי נחזיר null

אחרת - נריץ את $A(x_i)$ ונפתור אותו, ונקדם את $i=i+1$.

הוכחת זמן ריצה – עבור על איטרציה על x_i אנחנו רצים $O(n^c)$.

נכפיל כל אחד כזה באורך המילה (כמות השרשורים) שהיא k, ונקבל $O(kn^c)$.

3) (א) הוכיחו כי שיחס הרדוקצי-ה הפולינומיאלית הינו יחס טרנזטיבי על שפות. כלומר אם ידוע כי $L1 \le_p L2$ ו- $L2 \le_p L3$ אזי מתקיים $L1 \le_p L3$.

(ב) אם הרדוקציה מ- L1 ל- L2 רצה בזמן $O(n^{k1})$ והרדוקציה מ- L2 ל- L3 רצה בזמן $O(n^{k2})$ מהו זמן הריצה של הרדוקציה מ- L1 ל- L3? הסבירו!

א. נתונה לנו שפה $L_1 \in NPH$ (על סמך זה שניתן לבנות על גביה רדוקציה).

כמו כן, נתון לנו כי $L_1 \le_p L_2$ – כלומר $x \in L_1 \leftrightarrow f'(x) \in L_2$ כלומר $L_2 \in NPH$.

כמו כן, נתון לנו כי $L_2 \le_p L_3$ – כלומר $x \in L_2 \leftrightarrow f''(x) \in L_3$ כלומר $L_3 \in NPH$.

טענה: $L_1 \le_p L_3$.

הוכחה:

נגדיר $L_1 \le_p L_3$ בעזרת פונקציית הרדוקציה r הבאה:

1. הפעל את f' על x.
2. הפעל את $f''(f'(x))$

מהנתונים ניתן לראות כי $x \in L_1 \leftrightarrow f'(x) \in L_2 \leftrightarrow f''(f'(x)) \in L_3$.

כלומר השורה הראשונה מוכרחת לקיים את הרדוקציה אמ"ם השורה השניה תשרה את הרדוקציה של התוצאה, ואם כן $x \in L_1 \leftrightarrow r(x) \in L_3$.

הוכחת זמן ריצה פולינומי:

שורה 1 – רצה תחת זמן פולינומיאלי.

שורה 2 – רצה תחת זמן פולינומיאלי.

מאחר ו-P סגורה לאיחוד ולשרשור, כל האלגוריתם רץ בזמן פולינומיאלי.

ב. $O(n^{k_1}+n^{k_2})$ כאשר מדובר על חיבור מלא (לא הגבוה מביניהם).

על מנת לבצע את הרדוקציה, על הפונקציה לעשות את כל חלקי הרדוקציה בכל שלב, ולכן יהיה מדובר בחיבור הזמנים.

4) הוכיחו או הביאו דוגמה נגדית (או לחילופין הסבירו למה זה לא חייב להתקיים):

(א) אם $L1 \in NPC$ ו- $L2 \in NPC$ אזי $L1 \le_p L2$ ו- $L2 \le_p L1$

(ב) אם $L1 \in P$ ו- $L2 \in P$ אזי $L1 \le_p L2$ ו- $L2 \le_p L1$

(ג) אם $L1 \in NP$ ו- L2 קשה ב- NP אזי $L2 \le_p L1$

(ד) אם $L1 \in NP$ אזי $CO\text{-}L1 \le_p TAUT$ (להזכירכם CO-L1 הינה הבעיה המשלימה ל- L1)

ה) אם L1 קשה ב- NP ו- L2 ב- NP אזי $L1 \cap L2 \in NP$

א. נכון. כל שפה $L \in NPC$ צריכה לקיים שני תנאים – 1. $L \in NP$. 2. $L \in NPH$ (כלומר- ניתן לעשות לשפה זו רדוקציה מכל שפה ב-NP). ומאחר ששתי השפות הן שייכות לNPC, ניתן לעשות אליה רדוקציה מכל שפה ב-NP, ומאחר ששתי השפות נמצאות גם ב-NP, זה אומר שניתן לעשות מכל אחת לשניה רדוקציה.

ב. נכון. מאחר ואנחנו יכולים **לפתור** כל אחת מהשפות תחת זמן אלגוריתמי, אלגוריתם הרדוקציה מאחד לשני יהיה פשוט 1. פתור את הפונקציה (תחת זמן פוליינומיאלי) 2. העבר את התשובה לשפה השניה.

ג. לא מחויב. אם $L_1 \in NPC$ – אזי היא גם קשה וגם ב-NP, ואז זה יכול להתקיים.

אחרת, לא נכון. כי אם ניתן יהיה לעשות רדוקציה כמו המתואר השפה L_1 תהיה חייבת להיות NPC, ויצאנו מנקודת הנחה שהיא אינה כזו.

ד. לא נכון. הגדרנו את coNP להיות המחלקה שניתן לאמת את אי הנכונות תחת זמן פולינומיאלי, ובשם שבדיקת הנכונות היא לא טאוטולוגיה, גם בדיקת אי הנכונות המשלימה אינה טאוטולוגיה.

ה. לא נכון. השפות ב-NP סגורות תחת חיתוך.

אך אם L_1 קשה בNP אך אינה מוכלת בה, גם החיתוך ביניהם הוא לא מוגדר לנו.

בהצלחה!