תורת הסיבוכיות - תרגול 4
מכונות טיורינג עם אוב ושאלת P=NP

מכונות עם אוב - תזכורת

מכונת טיורינג עם אוב זהה למכונת טיורינג רגילה פרט ליכולת נוספת שיש לה - היא מצויידת ב"סרט שאלות לאוב" שעליו היא יכולה לכתוב מילה, ואז להיכנס למצב מיוחד של "קריאה לאוב". מייד לאחר מכן תוכן סרט השאלה לאוב יימחק, ויוחלף ב־0 או 1, בהתאם לזהות האוב של המכונה. אם האוב של המכונה הוא עבור השפה L, אז על מילה w האוב יענה ב־1 אם ורק אם $w \in L$. שימו לב שאותה מכונה עשויה לעבוד עם מספר אובות שונים, והתנהגותה על אותו קלט תשתנה כתוצאה מהתשובות השונות שהם יספקו לה.

הגדרנו CLASS^A כמחלקת השפות המתקבלת על ידי מכונה מה"טיפוס" של המחלקה CLASS עם אוב לשפה A. כך למשל $\mathrm{P}^{\mathrm{SAT}}$ היא מחלקת כל השפות המתקבלות על ידי מכונה דטרמיניסטית בעלת זמן ריצה פולינומי וגישת אוב ל־SAT. קל לראות כי $\mathrm{NP} \cup \mathrm{coNP} \subseteq \mathrm{P}^{\mathrm{SAT}}$: עבור שפה ב־NP נפעיל את הרדוקציה שלה לפסוק φ ונענה כמו האוב עליו; עבור $L \in \mathrm{coNP}$ מתקיים על פי הגדרה ש־$\overline{L} \in \mathrm{NP}$ ולכן ניתן להכריע את $\overline{L}$ באמצעות מכונת $\mathrm{P}^{\mathrm{SAT}}$, ומכיוון שהיא דטרמיניסטית וניתן להחליף את המצבים הסופיים, ניתן להכריע כך גם את L.

כמו כן הגדרנו $\mathrm{CLASS}_1^{\mathrm{CLASS}_2}$ כמחלקת השפות המתקבלות על ידי מכונה מהטיפוס של המחלקה CLASS_1 עם אוב לשפה במחלקה CLASS_2.

מספר דוגמאות:

1. $\mathrm{P}^{\mathrm{NP}} = \mathrm{P}^{\mathrm{SAT}}$ בזכות ה־NP־שלמות של SAT.
2. $\mathrm{P}^{\mathrm{P}} = \mathrm{P}$ כי ניתן לבצע סימולציה של האוב.
3. לא ידוע האם $\mathrm{NP}^{\mathrm{NP}} = \mathrm{NP}$ - כפי שראינו, זה בפרט יגרור ש־coNP⊆NP (הכוח הנוסף נובע מכך שהאוב עונה תשובה דטרמיניסטית).
4. $\mathrm{PSPACE}^{\mathrm{PSPACE}} = \mathrm{PSPACE}$ - שוב, על ידי סימולציה של האוב.
5. $\mathrm{P}^{\mathrm{TQBF}} = \mathrm{PSPACE}$ - נובע מהשלמות של TQBF ביחס לרדוקציות שניתנות לחישוב ב־P.

הוכחה כי לא ניתן להכריע את שאלת $\mathrm{P} = \mathrm{NP}$ באמצעות לכסון

שיטת הלכסון הוכיחה את עצמה ככלי רב עוצמה להפרדה בין מחלקות סיבוכיות (למשל, משפטי ההירכייה). "לכסון" הוא שם כללי למספר טכניקות (מתוחכמות יותר ופחות) בעלות עקרונות משותפים בסיסיים, כשהעיקרון הבסיסי ביותר הוא בכך שבלכסון אנו מתייחסים למכונות טיורינג כאל "קופסאות שחורות": כל שאנו מסתמכים עליו הוא העובדה שניתן לקודד מכונות טיורינג באמצעות מחרוזות סופיות, ולבצע סימולציה למכונת טיורינג, בהינתן הקידוד שלה, בלי תקורה משמעותית במשאבים. בשל פשטות ההנחות הללו, ההוכחות נותרות תקפות גם כאשר מפעילים אותן על מכונות טיורינג אשר חוזקו באמצעות אוב. במילים אחרות, אם ראינו למשל כי $\mathrm{DL} \neq \mathrm{PSPACE}$, אז גם לכל שפת אוב A יתקיים ש־$\mathrm{DL}^A \neq \mathrm{PSPACE}^A$. בניסוח קצר וקולע - הוכחה בלכסון עובדת **ביחס לכל אוב**.

נראה כעת שתי שפות אוב A, B כך ש־$\mathrm{P}^A = \mathrm{NP}^A$ ולעומת זאת $\mathrm{P}^B \neq \mathrm{NP}^B$. תוצאה זו (של Baker, Gill, Solovay מ־75) מראה כי לכסון **לא יכול להוכיח** כי $\mathrm{P} = \mathrm{NP}$ או $\mathrm{P} \neq \mathrm{NP}$ (אחרת לא היה ניתן לבצע "הפרדה" שכזו באמצעות אובות). זו אחת מהעדויות המוקדמות לכך שבעיית P=NP היא קשה יחסית (מאז נמצאו עדויות נוספות לקושי של הבעיה, שלא נציג כאן).

הוכחת החלק הקל

בתור A מספיק לבחור שפה חזקה כל כך שהיא מטשטשת את ההבדלים בין P ו־NP. שפה מושלמת לצורך העניין היא TQBF. מתקיים:

$$\mathrm{P}^{\mathrm{TQBF}} \subseteq \mathrm{NP}^{\mathrm{TQBF}} \subseteq_{(1)} \mathrm{NSPACE} \subseteq_{(2)} \mathrm{PSPACE} \subseteq_{(3)} \mathrm{P}^{\mathrm{TQBF}}$$

מעבר 1 נובע מכך ש־$\mathrm{TQBF} \in \mathrm{PSPACE}$ ולכן מכונת NPSPACE יכולה פשוט לסמלץ את האוב.

מעבר 2 נובע ממשפט סביץ'.

מעבר 3 נובע מהטענה שהבאנו כדוגמה קודם.

משרשרת ההכלות הללו נובע שכולן שוויונות, ולכן בפרט $\mathrm{P}^{\mathrm{TQBF}} = \mathrm{NP}^{\mathrm{TQBF}}$.

הוכחת החלק הקשה

כעת עלינו למצוא שפה B כך שעבורה יהיה קל יחסית להוכיח כי $\mathrm{P}^B \neq \mathrm{NP}^B$. זה מחזיר אותנו שוב לשאלה הבסיסית - איך אפשר להוכיח ששתי מחלקות שכאלו שונות זו מזו? והתשובה היא שכרגיל, בלכסון. אם כן, נציג B שהוכחת $\mathrm{P}^B \neq \mathrm{NP}^B$ בלכסון כן אפשרית **עבורה**, ובכך נוכיח שהוכחת הטענה הכללית יותר $\mathrm{P} \neq \mathrm{NP}$ בלכסון היא בלתי אפשרית. במילים אחרות, כדי להוכיח שאי אפשר להוכיח משהו בלכסון, משתמשים בלכסון.

לכל שפה B אפשר לבנות את שפת "כל המילים האונריות שאורכן כאורך מילים מ־B". פורמלית: $U_B = \{1^n | B \cap \Sigma^n \neq \emptyset\}$.

בבירור $U_B \in \mathrm{NP}^B$ לכל B: בהינתן קלט 1^n המכונה תנחש מילה באורך n, תשאל את האוב של B עליה ותענה כמותו. אם כן, אם נצליח לבנות B כך ש־$U_B \notin \mathrm{P}^B$, סיימנו.

בניית B תהיה כאמור בלכסון. המטרה היא לבנות את B כך שכל מכונה דטרמיניסטית פולינומית עם אוב ל־B טועה על מילה כלשהי מתוך U_B. יתרון משמעותי אחד שלנו על פני לכסונים "רגילים" הוא שאיננו צריכים לבנות מכונה שמכריעה את B; אנחנו יכולים להתבונן "מגבוה" על התנהגות כל המכונות בעולם ולקבוע את המילים ב־B באופן זה. אם כן, מדוע לכסון נאיבי לא עובד? נקבע מספור של כל מכונות הטיורינג כשהן מסודרות בסדר לקסיקוגרפי: $M_1, M_2, M_3, \ldots$. מדוע לא ניתן פשוט לקבוע ש־$1^n \in B$ אם ורק אם M_n דוחה את הקלט 1^n?

הבעיה היא בכך ש־M_n היא בעלת גישת אוב ל־B, ולכן ההתנהגות של M_n על כל קלט שהוא תלויה בהגדרה הספציפית של B. מכיוון שאנו מנסים להגדיר את B באמצעות "היפוך" של ההתנהגות של M_n על קלט מסויים, נוצרת כאן תלות מעגלית שיש לשבור איכשהו. בפרט, ההצעה שלמעלה כושלת מכיוון ש־M_n על קלט 1^n יכולה פשוט לשאול את האוב על 1^n ולענות כמוהו. מכאן שנדרשת שיטת לכסון מחוכמת יותר. ניעזר בכך שהמכונות שאנו רוצים לתקוף הן אלו שרצות בזמן פולינומי, ובזמן פולינומי לא ניתן לשאול את האוב **יותר מדי** שאלות.

אם כן, נבנה את B בשלבים, שנמספר בתור $i = 1, 2, 3, \ldots$. בתחילת הבנייה, $B = \emptyset$. במהלך הבנייה תישמר תמיד התכונה הבאה: קיים n טבעי כך שכל מילה w שעבורה כבר החלטנו אם $w \in B$ או $w \notin B$ היא מאורך $|w| < n$ (בהתחלה $n = 0$ כמובן).

כדי לפשט את החיים לעצמנו נניח כי כל מכונת טיורינג מופיעה אינסוף פעמים במניה (זוהי דרך אחרת לנסח את התעלול הישן של פירוש כל מחרוזת w כ־$1^k 0 \langle M \rangle$).

בשלב i ניקח את ה־n שקיומו מובטח על ידי התכונה, ונריץ את M_i על הקלט 1^n למשך כמות אקספוננציאלית של צעדים - $\frac{2^n}{10}$. אם M_i שואלת את האוב על מילים שכבר החלטנו האם הן שייכות או שאינן שייכות ל־B, נענה בהתאם להחלטה שלנו. אם M_i שואלת על מילים שטרם החלטנו לגביהן, נענה שהן אינן ב־B ונסמן לעצמנו שאותן מילים אינן ב־B.

אם M_i סיימה את ריצתה על 1^n וקיבלה, אין צורך לעשות כלום - כרגע אף מילה מאורך n אינה ב־B ולכן תשובת M_i על 1^n שגויה.

אם לעומת זאת M_i דחתה את 1^n, אנחנו חייבים לעשות דבר מה שממנו ינבע ש־$1^n \in U_B$ ולכן M_i טועה. כלומר, אנו רוצים לגרום לכך שמילה **כלשהי** מאורך n תהיה ב־B. הבעיה היא ש־M_i עשויה הייתה לשאול את האוב על חלק מהמילים מאורך n (וכזכור, על כולן אמרנו לאוב לענות בשלילה) - אבל מכיוון ש־M_i רצה רק $\frac{2^n}{10}$ צעדים ויש 2^n מילים מאורך n, מובטח לנו שקיימת מילה מאורך n שעליה M_i לא שאלה את האוב; נוסיף את המילה הזו ל־B ובכך נבטיח ש־M_i טועה.

נשים לב כי כעת קיים n' חדש הגדול מאורך כל המילים שהכרענו לגביהן, מאחר ו־M_i ביצעה רק מספר סופי של שאילתות. לכן התכונה שתיארנו בתחילת ההוכחה נשמרת.

כעת הושלם תיאור הלכסון. נשים לב שבניית B **איננה** אלגוריתמית - תיארנו תהליך שאינו מסתיים אף פעם. עם זאת, הוא מגדיר באופן חד משמעי שפה B (ייתכן שיש מילים שאף שלב בתהליך לא מכריע לגביהן - במקרה זה קובעים שהן אינן ב־B). ושפה זו היא השפה המבוקשת. כדי לראות שהבניה עובדת, ניקח מכונה M פולינומית עם פולינום $p(n)$ וניקח i גדול דיו כך ש־$M = M_i$ וכמו כן $p(i) < \frac{2^i}{10}$; בסיבוב ה־i בבניה, M_i תרוץ על קלט $n \geq i$ (גדול מ־i שכן כל סיבוב קובע את גורלן של המילים מגודל אחד מסויים לפחות) במשך מספר גדול דיו של צעדים ולכן תעצור, ולכן תענה על 1^n תשובה שגויה ביחס לשפה U_B, כנדרש.