1

מושגים בסיסיים בתורת הסיבוכיות

בעיות אופטימיזציה
מצאו פתרון גדול ביותר
או קטן ביותר

בעיות הכרעה/החלטה
האם קיים פתרון לבעיה

דוגמאות

1. נתונה רשת זרימה, מצאו בה זרימה מקסימלית
2. נתון גרף דו-צדדי, מצאו בו זוג מקסימלי
3. נתון גרף מכוון קשיר וממושקל, מצאו בו עץ פורש מינימלי

1. נתון גרף - האם קיים בו מעגל אוילר
2. נתון גרף ומספר k. האם קיים בגרף מסלול פשוט בעל k קשתות לפחות
3. נתון גרף דו-צדדי, האם קיים בו זוג מושלם (שבו כל הצמתים מזווגים).

הגדרות: בעיה קלה היא בעיה שניתן לפתור בזמן ריצה שהוא פולינום באורך הקלט. בעיה קשה היא בעיה שאינה קלה.

דוגמאות:

בעיות קלות: מיון, מציאת מסלולים קצרים ביותר, חישוב זרימה מקסימלית ברשת.

בעיות קשות: מציאת כל המסלולים הפשוטים בגרף נתון (בעיה זו קשה כי מספרם יכול להיות אקספוננציאלי).
מציאת כל הפרמוטציות של n מספרים.

מטרת תורת הסיבוכיות היא להשוות קושי של בעיות ולחלק אותן למחלקות.

לכל בעית אופטימיזציה ניתן להתאים באופן טבעי בעיית החלטה.

דוגמאות:

בעיית אופטימיזציה	בעיית ההחלטה המתאימה
1. מצאו מסלול קצר ביותר בין s ל-t	1. נתון k, האם קיים מסלול בין s ל-t באורך k לכל היותר
2. מצאו זרימה מקסימלית ברשת G	2. האם ניתן למצוא זרימה בגודל M לפחות ברשת נתונה עבור M נתון?

בעיית ההחלטה אינה קשה מבעיית האופטימיזציה (כי פתרון של בעיית האופטימיזציה נותן מיידית פתרון לבע"ת ההחלטה).
נשתמש בתכונה זו ונתמקד במטרה להראות שבעיות הכרעה מסוימות הן קשות באופן יחסי.

קידוד של בעיות

ניתן לקודד בעיות בצורה קונקרטית ע"י שימוש בשפות פורמליות. בעיה תקודד כמילה בינרית.
בעיה מופשטת - למשל "נתון גרף G, האם הוא קשיר"
ניתן לקודד כשפה L שהיא תת קבוצה של שפת המילים הבינריות. לכל גרף ניתן לשייך מילה בינרית (המייצגת למשל את רשימות השכנות או מטריצת הסמיכויות)
השפה L תכיל את הקידודים שמייצגים גרף קשיר.
לכל מילה שלא מתאימה לגרף ניתן להניח למשל שהיא מייצגת את הגרף: • • (שאינו קשיר).
ובאופן כללי
$L = \{w :$ התשובה לבעיית ההכרעה על w היא "כן"$\}$

אלגוריתם הכרעה A

זהו אלגוריתם המקבל כקלט מילה x ונותן כפלט תשובה

"כן" או "לא" (כלומר 1 או 0)
השפה המזוהה ע"י A היא
$L_A = \{x : A(x) = 1\}$
כלומר המילים הבינריות שעליהן A עונה "כן".
A הוא אלג' הכרעה פולינומיאלי אם קיים קבוע ג
כך שזמן הריצה של A על הקלט x הוא $O(|x|^{ג})$.

בצורה לא פורמלית A הוא אלגוריתם "פותר" בע"ת הכרעה.
<u>מחלקת הסיבוכיות P</u> אוסף כל בעיות ההכרעה שקיים עבורן אלג' הכרעה פולינומיאלי.
(כלומר מחלקת הבעיות ה"קלות").

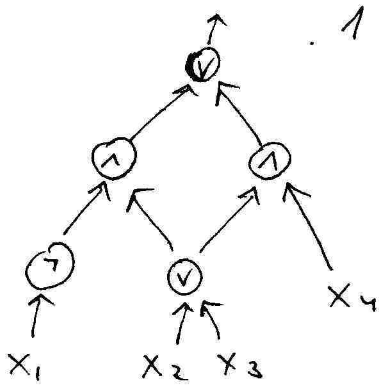
<u>אלגוריתם אימות פולינומיאלי A לשפה L</u>
A מקבל כקלט זוג מילים x,y, כאשר ל-x נתייחס כמילת קלט כמו קודם ואילו ל-y נתייחס כ"הוכחה" או "ניחוש".
כדי ש-A אכן יהיה אלג' אימות פולי' צריכים להתקיים שני התנאים הבאים (וצריך להיות קיים קבוע k עבורם)!
א. זמן הריצה של A פולינומיאלי ב-$|y|,|x|$ (אורכי x,y).
ב.1. אם $x \in L$ אז קיים y, כך שאורכו של y פולינומיאלי ב-x (כלומר $|y| \le |x|^k$)
וכך ש-$A(x,y) = 1$
ב.2. אם $x \notin L$ אז לכל y באורך סביר (כלומר $|y| \le |x|^k$) $A(x,y) = 0$.
הערה: בסעיף א הכוונה היא שזמן הריצה
$O((|x|+|y|)^k)$

בצורה פחות פורמלית, A רוצה לבדוק אם x שייכת לשפה ולכן מייצר רמז y המאפשר לבדוק זאת.

<u>מחלקת הסיבוכיות NP</u> - אוסף כל בעיות ההכרעה שקיים עבורן אלג' אימות פולינומיאלי.

<u>רשימת בעיות ב-NP</u>

1. CIRCUIT-SAT. נתון מעגל לוגי בוליאני (עם שערי NOT, AND, OR) וכניסות למשתנים. האם קיימת הצבה שעבורה המעגל מחזיר ערך 1.



דוגמה:
נציב
$x_1=0,\ x_2=0,\ x_3=1,\ x_4=1$
ונקבל ערך מוחזר 1
(ההצבה
$x_1=0 \quad x_2=0 \quad x_3=0 \quad x_4=0$
תחזיר ערך אפס).

2. SAT נתונה נוסחה בוליאנית בצורת "מכפלת סכומים". האם יש הצבה למשתנים הנותנת לה ערך 1?
לדוגמה: $(x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_4) \wedge (x_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee x_4)$

3. 3-SAT כמו 2, אך בכל הסגר יש בדיוק שלושה משתנים או שלילותיהם

4. CLIQUE (קליק). נתון גרף לא מכוון ומספר k. האם קיימת בגרף תת קבוצה של צמתים U, כך ש- $|U| \geq k$, וכל זוג מצמתי U מחובר בקשת.

5. HAM-CYCLE מעגל המילטון. נתון גרף G, האם קיים בו מעגל פשוט המכיל את כל הצמתים

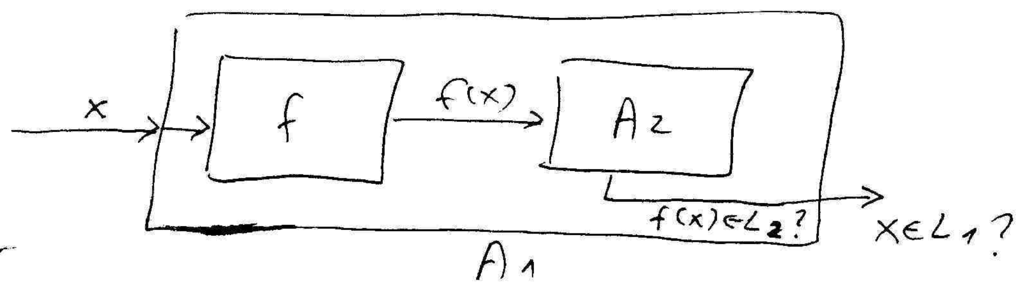
6. HAM-PATH מסלול המילטון. כמו 5 אבל עם מסלול

7. SUBSET-SUM בעיית הסכום החלקי.
נתונה קבוצת מספרים טבעיים $\{a_1,\dots,a_m\}$
ומספר טבעי Z. האם קיימת תת קבוצה שסכומה
בדיוק Z ?

המחלקה NPC (NP-complete).
בעיה L תקרא NP-שלמה אם מתקיימים שני
התנאים:
א. $L \in NP$
ב. לכל $L_1 \in NP$ מתקיים $L_1 \le_p L$

הגדרת היחס $L_1 \le_p L_2$ יחס זה משמעו שקיימת
רדוקציה פולינומיאלית מ-L_1 ל-L_2.
הגדרת רדוקציה פולינומיאלית:
תהי פונקציה f $\quad f: \{0,1\}^* \rightarrow \{0,1\}^*$
($\{0,1\}^*$ היא שפת כל המילים הבינריות הסופיות).
שמקיימת את התנאים
א. f ניתנת לחישוב ע"י אלג' פולינומיאלי
ב. לכל x מתקיים $x \in L_1 \Leftrightarrow f(x) \in L_2$

הרעיון - f היא פונקציה (לאו דווקא חח"ע ולאו דווקא על)
המאפשרת לפתור את בעיה L_1 בשימוש בפתרון ל-L_2
אם קיים פתרון כזה.
נסמן ב-A_2 אלגוריתם הפותר את L_2, אז נוכל
לבנות ממנו אלגוריתם A_1 הפותר את L_1.
A_1 ממיר את הקלט שלו x ל-$f(x)$, משתמש ב-A_2
כדי לענות על $f(x)$ (כן או לא) ומחזיר את אותה
תשובה עבור x. לפי הגדרת הרדוקציה זוהי אותה תשובה.



יש לשים לב שלפי הגדרת הרדוקציה יתכן ש-f מעבירה מילים בינריות לקבוצה קטנה מאוד של מילים בינריות (למשל שתיים - אחת ששייכת ל-L_2 ואחת שלא).

NPC נחשבת למחלקת הבעיות שהן הכי קשות מתוך NP.

<u>משפט</u> (ישמש אותנו רבות)

אם $L_1 \in NPC$, $L_2 \in NP$ וכן $L_1 \le_p L_2$ אז גם $L_2 \in NPC$

<u>הוכחה</u> מטרנזיטיביות הרדוקציה הפולינומיאלית

כיון שזהו מבוא קצר בלבד, נתמקד בהוכחת שייכות ל-NPC של בעיות. נניח ששבע הבעיות בעמ' 4-5 ידועות כ-NPC ונשתמש בכך כדי להוכיח לבעיות נוספות.

השיטה: נתונה בעיה P_2 שיש להוכיח שהיא NPC

<u>שלב 1</u> אלגוריתם אימות (זהו השלב הקל אך אין לפסוח עליו).

<u>שלב 2</u> נבחר בעיה שאני יודע שהיא NPC, נסמנה P_1 ונעשה <u>ממנה</u> רדוקציה לבעיה P_2 שלנו. (רדוקציה פולינומיאלית).