תרגול 13 – המשך מבוא לסיבוכיות

: SUSU – Partition

תזכורת:

בהינתן בעיה A, מהם השלבים להוכחת היותה NP-שלמה? (תבנית להוכחת NP-completeness)

1. הוכיחו ש-A ב-NP: מצאו אלגוריתם אימות פולינומיאלי והוכיחו נכונותו.
2. הוכיחו ש-A NP-קשה: (כלומר היא לא קלה משום בעיה ב-NP)
   - בחרו בעיה $B\in NPC$.
   - הוכיחו ש-$B\leq_p A$:
     - א. מצאו רדוקציה פולינומיאלית f מ-A ל-B: הציגו את המעבר מקלט לבעיה B לקלט לבעיה A.
     - ב. הוכיחו כי הרדוקציה ניתנת לחישוב בזמן פולינומי בגודל הקלט.
     - ג. הוכיחו $x \in B \Leftrightarrow f(x) \in A$. (שימו לב: הוכחה לשני הכיוונים!)

   כיוון ש-B היא NP קשה אזי כל שפה ב-NP ניתנת לרדוקציה פולינומיאלית ל-B.
   מטרנזיטיביות היחס $\leq_p$ עולה כי כל שפה ב-NP ניתנת לרדוקציה פולינומיאלית ל-A,
   ולכן גם A היא NP קשה.

בעיה 1- Subset Sum (SUSU):

מופע: $C=(c_1,\ldots,c_n)$, T כאשר C סדרת מספרים טבעיים ו T מספר טבעי.

פלט: יש להכריע האם קיימת תת-סדרה של C כך שסכום האיברים בה הוא T.

כלומר, האם קיימת קבוצת אינדקסים $D = \{i_1, \ldots, i_k\}$ כך ש $\sum_{j\in D} c_j = T$.

בעיית SUSU היא NP שלמה, בתירגול לא נוכיח זאת – ניתן לראות את ההוכחה בספר (פרק 37.4)

בעיה 2- Partition בעיית החלוקה:

מופע: $A=(a_1,\ldots,a_n)$, סדרת מספרים טבעיים.

פלט: יש להכריע האם ניתן לחלק את A לשתי תתי-סדרות שסכום איברהן שווה.

כלומר, האם קיימת חלוקה של $\{1,\ldots,n\}$ לשתי קבוצות זרות: D_1 ו D_2 כך ש:

$$\sum_{i\in D_1} a_i = \sum_{i\in D_2} a_i = \frac{1}{2}\sum_{i=1}^{n} a\_i$$

טענה: בעיית החלוקה היא NP שלמה.

הוכחה:

1) $Partition\in NP$:

עד יהיה תת-קבוצת איברים מ-A (ולכן גודלו פולינומי בקלט).

אלגוריתם האימות מקבל קלט סדרה A , עד תת סדרה Y:

נסכום את איברי A (נסמן סכום זה ב-SA)

נסכום את איברים העד Y (נסמן סכום זה ב-SY) ונוודא כי Y אכן תת סדרה של A

נבדוק אם SA = 2SY ונענה בהתאם.

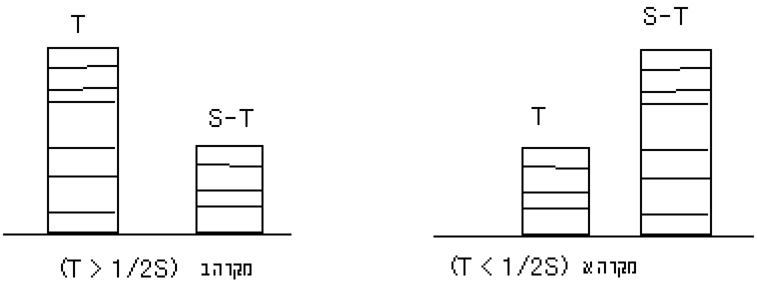
(יש להוכיח נכונות ופולינומיאליות אלגוריתם האימות)

2) בעיית החלוקה היא NP קשה:

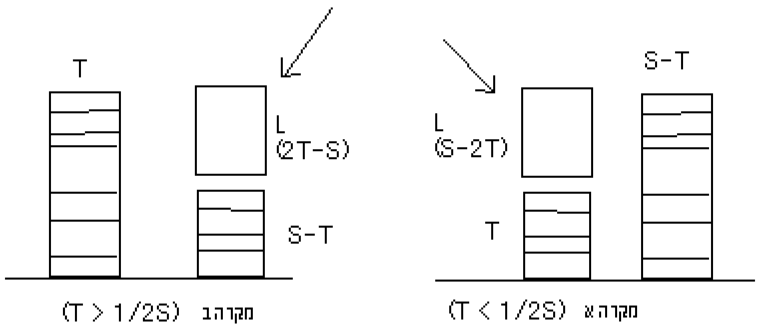
- נבחר את SUSU כבעיה NP קשה.
- נראה כי $SUSU \leq_p Partition$

א) **תיאור הרדוקציה f:** (הערה: אם $T = \frac{1}{2}S$ אזי שתי הבעיות זהות.)

אינטואיציה: נחשוב על סדרת המספרים כעל סדרת אבנים עם גבהים שלמים (מספר a_i הוא אבן עם גובה a_i). כעת אנו רוצים לחלק את האבנים לשתי ערימות בגודל T ו- S-T.



הפועלים שלנו יודעים רק לחלק את הערימה רק לשתי ערמות שוות גובה. איזה אבן L נרצה להוסיף בכל פעם לפתור את הבעיה ?
במקרה הראשון נרצה להגדיל את הערימה הנמוכה מגובה T לגובה S-T ולקבל שתי ערימות בגובה S-T (ולמציאת ערימה בגובה T ניקח את הערימה בה יש את L ונסתכל על שאר האיברים (מלבד L)).
במקרה השני נרצה להגדיל את הערימה הנמוכה מגובה S-T לגובה T ולקבל שתי ערימות בגובה T (ולמציאת ערימה בגובה T ניקח את הערימה בה אין את L).



באופן פורמאלי:
בהינתן קלט: $C = (c_1 \dots , c_n), T$ לבעיית SUSU:
נסכום $\sum_{c_i \in C} c_i$ ונסמן ב-S

אם $T \le \frac{1}{2} S$ (מקרה ראשון)
נקח $L = S - 2T$ (מספר חיובי ושלם) ונחזיר: $A = (c_1, \dots , c_n, L)$.
אם $T > \frac{1}{2} S$ (מקרה שני)
נקח $L = 2T - S$ (מספר חיובי ושלם) ונחזיר $A = (c_1 \dots , c_n, L)$.

ב) חישוב S ו L ויצירת הסדרות הם פולינומיאלים בגודל הקלט.
לכן, הרדוקציה פולינומיאלית בגודל הקלט.

ג) נוכיח $x \in SuSu \Leftrightarrow f(x) \in Partition$,
כלומר יש קבוצת אינדקסים D כך ש- $\sum_{j \in D} c_j = T$ אמ"מ יש חלוקה של A לשתי תתי סדרות שסכומן זהה.

הוכחת הכיוון $\Rightarrow$

מקרה ראשון: $T \leq \frac{1}{2}S$

אם x∈ SUSU , אזי קיימת קבוצת אינדקסים D = { i_1,..i_k} המקיימת:

$$\sum_{j\in D} c_j = T \leq \frac{1}{2}\sum_{c_j\in C} c_j$$

(נסמן שירשור של סדרות ע"י $\cup$ וסכום סדרה כלשהי C ע"י sum(C))

נבחר $D_1 = D \cup \{n+1\}$, (כאשר $L = a_{n+1}$), ובהתאם $D_2 = \{1, \dots, n+1\} \setminus D_1$.

נבדוק כי החלוקה עונה על תנאי בעיית Partition :

- מהגדרת D_2, D_1 ו D_2 חלוקה של $\{1, \dots, n+1\}$.

- נראה כי אכן:

$$\sum_{i\in D_1} a_i = \sum_{i\in D_2} a_i = \frac{1}{2}\sum_{i=1}^{n+1} a\_i$$

ע"פ ההנחה (x∈ SUSU) מתקיים $\sum_{j\in D} c_j = T$ וגם $\sum_{j\notin D} c_j = S - T$

$$sum(D1) = \sum_{j\in D} c_j + L = T + S - 2T = S - T$$

$$sum(D2) = \sum_{j\notin D} c_j = S - T$$

$$sum(A) = \sum_{c_j\in C} c_j + L = S + L = S + S - 2T = 2(S - T)$$

מקרה שני: $T > \frac{1}{2}S$

אם x∈ SUSU , אזי קיימת קבוצת אינדקסים D = { i_1,..i_k} המקיימת:

$$\sum_{j\in D} c_j = T > \frac{1}{2}\sum_{c_j\in C} c_j$$

נבחר $\mathrm{D}_1 = \mathrm{D}$, $D_2 = \{1, \dots, n+1\} \setminus D_1$.

נבדוק כי שני התנאים לבעיית Partition מתקיימים:

- מהגדרת D_2, D_1 ו D_2 חלוקה של $\{1, \dots, n+1\}$.
-
- נראה כי אכן:

$$\sum_{i\in D_1} a_i = \sum_{i\in D_2} a_i = \frac{1}{2}\sum_{i=1}^{n+1} a\_i$$

ע"פ ההנחה (x∈ SUSU) מתקיים $\sum_{j\in D} c_j = T$ וגם $\sum_{j\notin D} c_j = S - T$

$$sum(D1) = \sum_{j\in D} c_j = T$$

$$sum(D2) = \sum_{j\notin D} c_j + L = S - T + L = S - T + 2T - S = T$$

$$sum(A) = \sum_{c_j\in C} c_j + L = S + 2T - S = 2T$$

הוכחת הכיוון $\Leftarrow$
מקרה ראשון: $T \le \frac{1}{2}S$

אם f(x)∈Partiton אז קיימת חלוקה של $\{1, \dots, n+1\}$ לשתי קבוצות D_1 ו- D_2 כך ש:

$$\sum_{i \in D_1} c_i = \sum_{i \in D_2} c_i = \frac{1}{2}\sum_{i=1}^{n+1} a\_i$$

בה"כ נניח כי $n+1 \in D_1$ (זכרו כי האיבר שהוספנו L הוא האיבר ה $n+1$) ונבחר $D = D_1 \setminus \{n+1\}$

(נשים לב כי אברי D1 הם האינדקס של L ואינדקסים של אברים נוספים מתוך C)

כעת נראה כי אכן $x \in$ SuSu. נשים לב כי

$$\sum_{a_j \in A} a_j = \sum_{c_j \in C} c_j + L = S + S - 2T = 2(S - T)$$

ולכן $\sum_{i \in D_1} c_i = S - T$. מתוך בחירת D מתקיים:

$\sum_{i \in D} c_i = \sum_{i \in D_1} c_i - L = S - T - (S - 2T) = T$ כנדרש.

מקרה שני: $T > \frac{1}{2}S$

אם f(x)∈Partiton אז קיימת חלוקה של $\{1, \dots, n+1\}$ לשתי קבוצות D_1 ו- D_2 כך ש:

$$\sum_{i \in D_1} c_i = \sum_{i \in D_2} c_i = \frac{1}{2}\sum_{i=1}^{n+1} c_i$$

בה"כ נניח כי $n+1 \in D_2$ (האינדקס של L) ונבחר $D = D_1$.

(נשים לב כי אברי D1 הם אברים מתוך C. זאת על פי בניית A והעובדה כי L∈D2).

כעת נראה כי אכן $x \in$ SuSu. נשים לב כי

נשים לב כי: $\sum_{a_j \in A} a_j = \sum_{c_j \in C} c_j + L = S + 2T - S = 2T$

ולכן $\sum_{i \in D_1} c_i = T$.

מכיוון ש $D = D_1$ מתקיים: $\sum_{i \in D_1} c_i = T$ כנדרש.

<u>**הראנו כי בעיית החלוקה היא ב-NP וכי היא NP קשה ולכן היא NP שלמה** . מש"ל</u>

: KnapSack – SUSU

תזכורת לבעיית הגנב בשלמים:
נתונה קבוצת חפצים $A=(a_1,\ldots,a_n)$ ולכל חפץ יש ערך a_i ומשקל w_i . כמו כן נתון משקל מקסימאלי שהגנב יכול לסחוב –W, המטרה היא למצא אוסף של פריטים עם ערך מקסימאלי שניתן לקחת.

נתבונן **בגירסת ההכרעה** של הבעיה:
נתון: $A=(a_1,\ldots,a_n)$, P, W.
עבור כל a_i נתונים $a_i.w$, $a_i.p$.
צ"ל: האם קיימת קבוצת אינדקסים $B = \{i_1, \ldots, i_k\}$ כך ש:

- $\sum_{j\in B} a_j.w \le W$
- $\sum_{j\in B} a_j.p \ge P$

טענה:
בעיית הגנב הבדיד היא NP שלמה.

הוכחה:
1. $\text{Knap Sack} \in NP$:
אלגוריתם אימות מקבל את הקבוצה A ואת B כעד ובודק קיום של שלושת התנאים.
הוכחה + יעילות.

2. Knap Sack היא NP קשה:
- הבעיה ה-NP קשה שנבחר לרדוקציה: Subset Sum.
- נראה כי $SUSU \le_p KnapSack$

א) תיאור הרדוקציה f:
בהינתן מופע לבעיית SUSU $(C=(c_1,..,c_n), T)$ נבנה את המופע לבעיית KnapSack כך: $A=(a_1,\ldots,a_n)$ כאשר $(a_j.w = a_j.p = c_j)$, $P=W=T$.
(כלומר לכל מספר c_j נתאים פריט a_j שמשקלו ומחירו שווים ל c_j. את P ואת W נקבע להיות T).

ב) יעילות:
בניה פשוטה – טריוויאלי.

ג) נאותות - הוכחת נכונות הרדוקציה:

$f(x) \in KnapSack \Leftarrow x \in SUSU$

אם $x\in SUSU$, אזי קיימת קבוצת אידקסים $D = \{ i_1,..i_k\}$ המקיימת $\sum_{j\in D} c_j = T$.

תהי $B = D$.
נבדוק כי התנאים מתקיימים:

- $\sum_{j\in B} a_j.w \le W$: $\sum_{j\in B} a_j.w = \sum_{j\in D} c_j = T \le T$
- $\sum_{j\in B} a_j.p \ge P$: $\sum_{j\in B} a_j.p = \sum_{j\in D} c_j = T \ge T$

$f(x) \in KnapSack \Rightarrow x\in SUSU$

אם $f(x) \in KnapSack$ אזי קיימת קבוצת אינדקסים $B = \{i_1, \ldots, i_k\}$ כך ש:

- $\sum_{j\in B} a_j.w \le W$
- $\sum_{j\in B} a_j.p \ge P$

תהי D = B.

נבדוק כי שני התנאי מתקיים:

$\sum_{j\in D} c_j = T$: מצד אחד $\sum_{j\in D} c_j = \sum_{j\in B} a_j.w \le W = T$

ומצד שני: $\sum_{j\in D} c_j = \sum_{j\in B} a_j.p \ge P = T$

ולכן $\sum_{j\in D} c_j = T$

מש"ל (KnapSack NP קשה)

הראנו כי $\in$NP KnapSack וכי KnapSack היא NP קשה ולכן KnapSack היא NP שלמה.מש"ל

טוב, קרה פה משהו מוזר... כרגע הוכחנו שבעיית הגנב הבדידה הינה בעיה NP שלמה, כלומר שזו בעיה קשה (שלא ידוע פתרון יעיל עבורה..)
אבל בתחילת/באמצע הסמסטר ראינו שיש פתרון של תכנון דינאמי לבעיית הגנב..
אז מה בעצם קורה פה?

<u>תזכורת – פתרון בעית הגנב ע"י תכנון דינמי:</u>
בעזרת מטריצה n*W.
<u>זמן הריצה:</u> (זמן פעולת חיבור והשוואה)*n*W.
כאשר זמן פעולת חיבור והשוואה הוא {max{logP,logW.
<u>תאור הקלט:</u> n זוגות של מספרים (p_i, w_i), כך ש p_i מייצג את ערכו ו w_i מייצג את משקלו של הפריט ה i. בנוסף מספר W המייצג את סך המשקל שניתן לשאת בשק.
<u>גודל תאור הקלט:</u> נניח שכל p_i מקיים $P>p_i$, וכל w_i מקיים $W>w_i$.
אזי על מנת לייצג זוג מספרים (p_i, w_i), דרושים לנו לכל היותר log(W)+log(P) ביטים.
הקלט כולו יתואר עם כן ע"י לכל היותר n(log(W)+log(P))+log(W) ביטים, כלומר:
O(n*log(W)+n*log(P)) או O(n*max{logP,logW}).

אם נבחן את זמן הריצה של אלגוריתם התכנון הדינמי כפונקציה של גודל קלט הבעיה, נגלה כי אכן עבור W גדול מספיק (אקספוננציאלי ב-n) מתקבל זמן ריצה אקספוננציאלי. זאת משום ש $W = 2^{\log(W)}$!!