מבני נתונים ואלגוריתמים – תרגול 11

תכנון דינמי

תכנון דינמי –שיטה לפיתרון בעיות רקורסיביות בהן משתמשים כמה פעמים בפיתרון של תתי בעיות.
במקום לפתור את תתי הבעיות שוב ושוב, נחשב רק פעם אחת ונשמור את הפתרונות בטבלה.

בעיה לדוגמא –

<u>בעיית תת הסדרה המשותפת הארוכה ביותר – Largest Common Subsequence (LCS)</u>

הגדרת הבעיה:

<u>קלט:</u> 2 מחרוזות (באורכים שונים):

$$X = x_1 x_2 \dots x_n$$

$$Y = y_1 y_2 \dots y_m$$

<u>פלט:</u> תת סדרה משותפת מקסימלית:

$$Z = z_1 \dots z_k$$

כך שקיימים:

$$i_1 < i_2 < \dots < i_k$$

$$j_1 < j_2 < \dots < j_k$$

כך ש-

$$Z = x_{i_1} x_{i_2} \dots x_{i_k} = y_{j_1} y_{j_2} \dots y_{j_k}$$

(כלומר התווים לא חייבים להיות רציפים).

דוגמא – X = ABCBDAB, Y = BDCABA. אורך תת הסדרה המקסימלית היא 4: BCBA,BDAB.

<u>פיתרון נאיבי:</u>

- נעבור על כל תתי הסדרות של X
- מחפש האם הן קיימות ב-Y.

← <u>סיבוכיות</u> – אקספוננציאלי

<u>פיתרון רקורסיבי:</u>

אם נסתכל על התו האחרון בכל מחרוזת:

$$X = x_1 x_2 \dots x_{n-1} | x_n$$

$$Y = y_1 y_2 \dots y_{m-1} | y_m$$

אם $\boldsymbol{y_m = x_n}$: אז זה יכול להיות חלק מתת הסדרה הארוכה ביותר. נחתוך את התווים האלו ונסתכל על:

$$X' = x_1 x_2 \dots x_{n-1}$$

$$Y' = y_1 y_2 \dots y_{m-1}$$

אם נמצא ב-'X ו-'Y תת סדרה מקסימלית, נוכל להוסיף את x_n ו-y_m.

אם $\boldsymbol{y_m \neq x_n}$: אפשר להגיד בודאות ש- x_n לא שייך לתת הסדרה המקסימלית **או** y_m.

לכן נחפש את התת הסדרה המקסימלית ב-2 המקרים הבאים:

מקרה 1:

$$X' = x_1 x_2 \dots x_{n-1}$$

$$Y = y_1 y_2 \dots y_m$$

מקרה 2:

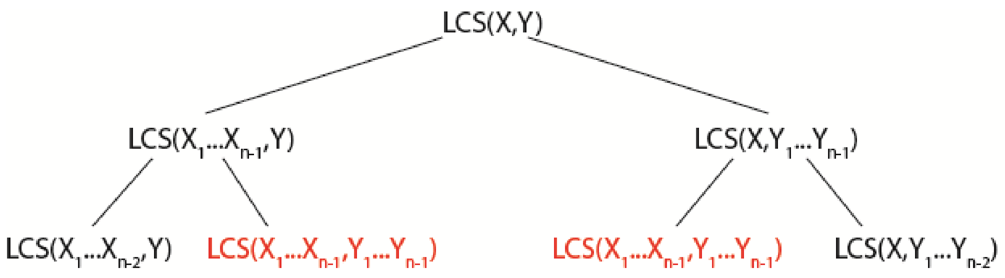
$$X = x_1 x_2 \dots x_n$$

$$Y' = y_1 y_2 \dots y_{m-1}$$

ננסח בצורה מסודרת את האלגוריתם הרקורסיבי:

```
LCS( X,n, Y,m ):
if n=0 or m=0
        return 0
else if X_n = Y_m
        return LCS(X,n-1,Y,m-1) +1
else // X_n ≠ Y_m
        return max{LCS(X,n,Y,m-1), LCS(X,n-1,Y,m)}
```

←<u>סיבוכיות:</u> גם אקספוננציאלי – במקרה הגרוע בכל קריאה לפונקציה, נעשה שוב 2 קריאות וכו':



← יש הרבה חישובים שחוזרים על עצמם!

<u>פיתרון באמצעות תכנון דינמי:</u>

- נשתמש בטבלה D שגודלה (n+1)(m+1) (1 עבור המחרוזת הריקה)
- בונים את הטבלה תא אחרי תא –תא (LCS(i,j מתאר את אורך תת הסדרה המקסימלית עבור המחרוזות:

$$X = x_1 x_2 \dots x_i$$
$$Y = y_1 y_2 \dots y_j$$

- מסתכלים על 3 תאים – למעלה, אלכסון, שמאלה.
- אפשר לשמור את המסלול וכך לשחזר את הפיתרון האופטימלי.

```
LCS(X, Y):
for i=0 to n
        D(i,0) = 0
for j=0 to m
        D(0,j) = 0
for i=1 to n
        for j=1 to m
                if X_i = Y_j
                        D(i,j) = D(i-1, j-1) + 1
                        P(i,j) = אלכסון
                else
                        D(i,j) = max{D(i,j-1), D(i-1,j)}
                        P(i,j) = שמאלה/למעלה
```

P- טבלה בגודל זהה ל-D שבה שומרים את המסלול.

←<u>סיבוכיות</u>: זמן ומקום O(nm).

- אם לא רוצים לשחזר את הפיתרון, אפשר לצמצם את סיבוכיות המקום ל-2 שורות (השורה הנוכחית והשורה הקודמת) ואז O(max(n,m)).

דוגמא:

	j	0	1	2	3	4	5
i		**-**	**B**	**D**	**C**	**A**	**B**
0	-	**0**	**0**	**0**	**0**	**0**	**0**
1	**A**	**0**	0	0	0	1	1
2	**B**	**0**	1	1	1	1	2
3	**C**	**0**	1	1	2	2	2
4	**B**	**0**	1	1	2	2	**3**

← הפיתרון האופטימלי במקרה זה נמצא ב-3 = D(n+1,m+1)

<u>שחזור הפיתרון:</u>

- מתחילים מ-P(n+1,m+1) והולכים בכיוון החצי.
- כאשר עולים למעלה באלכסון - מדפיסים את התו המתאים.

הפלט: B C B.

לסיכום – שלבי התכנון הדינמי:

- הגדרה רקורסיבית של הבעיה
- בניית מרחב כל תתי הפתרונות
- מציאת הפיתרון האופטימלי
- שחזור הפיתרון האופטימלי

תרגיל 1 – תכנון דינאמי לטובת תורת המשחקים

תרגיל

אליס ובוב משחקים את המשחק הבא – נתונה סדרה של n מספרים המסודרים בשורה משמאל לימין - $a_1, a_2, \ldots, a_n$. אליס ובוב משחקים לסירוגין כשכל אחד בתורו לוקח את המספר הימני ביותר או את השמאלי ביותר. אליס משחקת ראשונה. בסיום המשחק הרווח של אליס הוא סכום המספרים שהיא לקחה פחות סכום המספרים שבוב לקח. הרווח של בוב הוא סכום המספרים שהוא לקח פחות סכום המספרים שאליס לקחה. מטרת כל שחקן היא להגיע לרווח מירבי. למשל אם הסדרה ההתחלתית היא 3,8,1 אז הרווח של אליס הוא -4 ושל בוב הוא 4 (לא משנה איזה מבין המספרים 1,3 אליס תבחר בצעד הראשון, בוב יבחר את המספר 8 בכדי להביא למקסימום את הרווח שלו, מה שיותיר לאליס את אותו מספר מבין 1,3 שלא בחרה בצעד הראשון).

הצע אלגוריתם הרץ בזמן $O\left(n^2\right)$ המחשב את הרווח המקסימאלי שאליס יכולה להבטיח לעצמה.

האם הגישה החמדנית תעבוד?

ראשית נציין כי הפתרון החמדני הטבעי – בחירת המספר הגדול יותר מבין שני המספרים בקצוות לא פועל. אכן, ייתכן והמספר הגדול שנסיר "מסתיר" מספר גדול הרבה יותר. כך למשל, אם סדרת המספרים היא $1,1,100,90$ אז אם אליס תבחר חמדנית את 90, אליס תחשוף בפני בוב את 100 ולבוב יובטח ניצחון. האסטרטגיה העדיפה על אליס היא לבחור את ה-1 . כך לבוב לא תהיה ברירה אלא לחשוף עבור אליס את ה-100 , מה שיבטיח לה ניצחון.

פתרון

עבור $i \le j$, נסמן ב-$T(i,j)$ את הרווח של אליס עבור משחק על הסדרה $a_i, a_{i+1}, \ldots, a_j$. היחס הרקורסיבי הוא היחס הבא

$$T(i,j) = \begin{cases} a_i & i = j \\ \max\left\{a_i - T(i+1,j), a_j - T(i,j-1)\right\} & i \neq j \end{cases}$$

<u>הסבר</u>: אם $i = j$, כלומר ישנו רק איבר אחד בסדרה, אין לאליס ברירה אלא לקחת אותו ולהרוויח a_i. אחרת, לאליס יש בחירה בין האם לקחת את a_i לבין לקחת את a_j. אם אליס בוחרת ב-a_i אז היא מרוויחה a_i לסכום המצטבר שלה, ותורו של בוב לשחק. כללי המשחק של בוב סימטרים לחלוטין לכללי המשחק של אליס, וכל מה שבוב לוקח יורד מהסכום המצטבר של אליס, ולכן המינוס.

קל לראות כי הסיבוכיות ריבועית.

תרגיל 2

יהא d מספר טבעי, ויהיו k ו- איברים $x_1, \ldots, x_n$ ב-Z_d. נרצה למצוא אלגוריתם המחשב האם ניתן להכניס סימנים של חיבור וחיסור בין ה-X -ים, כך שתוצאת הביטוי המתקבל תהיה $k \in Z_d$ (כל החישובים מתבצעים מודולו d).

כך למשל, אם ה-X -ים הם 7,6,9 ו- $k = 2$ אז ב-Z_{10} האלגוריתם יחזיר כן, כי

$$\begin{aligned} 7+6+9 &\equiv 2 \pmod{10} \\ 7-6-9 &\equiv 2 \pmod{10} \\ 7+6-9 &\equiv 4 \pmod{10} \\ 7-6+9 &\equiv 0 \pmod{10} \end{aligned}$$

הצע אלגוריתם לפתרון הבעיה, הרץ ב-$O(nd)$ זמן ו-$O(d)$ מקום.

פתרון

נגדיר את $T(i,j)$ להיות משתנה בוליאני, כך ש- $T(i,j) = true$ אם ניתן להכניס סימנים של חיבור וחיסור בין ה-X -ים $x_1, \dots, x_i$, כך שתוצאת הביטוי המתקבל תהיה $j \ (mod\ d)$.

היחס הרקורסיבי הוא היחס הבא

$$T(i,j) = T(i-1, (j - x_i) \bmod d) \quad OR \quad T(i-1, (j + x_i) \bmod d)$$

בסיס:

$$T(1,j) = \begin{cases} true & \text{if } j(mod\ d) = x_1 \\ false & \text{otherwise} \end{cases}$$

הסבר: אם $i = 1$ אז תוצאת הביטוי המתקבל מאיבר אחד הוא x_1, ואם $j(mod\ d) = x_1$, $T(1,j) = true$. אם $i = 1$ אבל $j(mod\ d) \neq x_1$, אזי לא ניתן לקבל את j(mod d) מהאיבר x_1 , ולכן $T(1,j) = false$.

עבור $i > 1$, נבדוק האם ניתן לקבל את $j\ (mod\ d)$ על ידי הוספת x_i או החסרת x_i לתוצאה שהתקבלה על ידי ה-X-ים $x_1, \dots, x_{i-1}$.

תיאור האלגוריתם: ניצור טבלה בגודל nd, ונמלא את הטבלה שורה אחר שורה לפי הנוסחת רקורסיה. נחזיר $T(n,k)$.

קל לראות כי סיבוכיות הזמן שווה ל- $O(nd)$ (ממלאים טבלה בגודל nd, כל תא בעלות $O(1)$). כמו כן, כיוון וחישוב כל תא מסתמך על תאים בשורה שמעליו בלבד, ניתן לשמור את השורה הנוכחית ואת השורה שמעליה בכל שלב. לכן, סיבוכיות המקום שווה ל- $O(d)$.