

הרצאה 9: תכנון דנאמי

Textbook: Cormen, Leiserson, Rivest,
Stein. Introduction to Algorithms.

מרצה: אורי שטמר

בעית התרמיל – המשך

בסוף השיעור הקודם כתבנו את נוסחת המבנה הבאה:

$$OPT(j, T') = \begin{cases} 0 & , j = 0 \\ OPT(j - 1, T') & , j > 0 \text{ \& } w_j > T' \\ \max\{ OPT(j - 1, T'), p_j + OPT(j - 1, T' - w_j) \} & , j > 0 \text{ \& } w_j \leq T' \end{cases}$$

הוכחת נוסחת הנבנה:

לכל אחד משלושת המקרים בנוסחת המבנה, ננתח את $OPT(j, T')$ ונראה שהוא שווה לאגף ימין של הנוסחה (לפי המקרה המתאים).

מקרה א: $j = 0$

במקרה זה, לכל T' מתקיים $Sol(0, T') = \{\emptyset\}$ ומחיר הפתרון הריק הוא אפס. לכן $OPT(0, T') = 0$.

מקרה ב: $j > 0 \text{ \& } w_j > T'$

אם $w_j > T'$ אז המוצר ה- j לא יכול להיות בפתרון ולכן כל פתרון חוקי מוכל ב- $j - 1$ המוצרים הראשונים. כלומר $Sol(j, T') \subseteq Sol(j - 1, T')$. בנוסף, תמיד מתקיים ש- $Sol(j - 1, T') \subseteq Sol(j, T')$ ולכן במקרה שלנו מתקיים ש- $Sol(j, T') = Sol(j - 1, T')$.
לכן

$$OPT(j, T') = \max_{I \in Sol(j, T')} p(I) = \max_{I \in Sol(j-1, T')} p(I) = OPT(j - 1, T')$$

מקרה ג: $j > 0 \text{ \& } w_j \leq T'$

אבחנה 0: כל פתרון חוקי לתת הבעיה (j, T') או מכיל את המוצר ה- j או לא.

אבחנה 1: קבוצת הפתרונות ב- $Sol(j, T')$ שלא כוללים את המוצר ה- j היא בדיוק $Sol(j - 1, T')$. כלומר,
 $\{I \in Sol(j, T') : j \notin I\} = Sol(j - 1, T')$

הסבר: נסתכל על כל הפתרונות החוקיים שמוכלים ב- $\{1, 2, \dots, j\}$ ולא מכילים את j . כלומר אלו פתרונות שמכילים מוצרים מ- $\{1, 2, \dots, j - 1\}$ שסך משקלם לכל היותר T' . כלומר אלו בדיוק כל הפתרונות ב- $Sol(j - 1, T')$.

טענה 2: הפתרונות ב- $Sol(j, T')$ שכן כוללים את המוצר ה- j הם בדיוק כל הפתרונות מהסוג $K \cup \{j\}$ עבור $K \in Sol(j - 1, T' - w_j)$. כלומר,

$$\{I \in Sol(j, T') : j \in I\} = \{K \cup \{j\} : K \in Sol(j - 1, T' - w_j)\}$$

זאת טענה אינטואיטיבית, אבל היא דורשת הוכחה.

הוכחת טענה 2: נראה הכלה בשני הכיוונים.

כיוון ראשון: יהי $I \in \{K \cup \{j\} : K \in \text{Sol}(j-1, T' - w_j)\}$ אזי $I = K \cup \{j\}$ עבור $K \in \text{Sol}(j-1, T' - w_j)$. בפרט, $K \subseteq \{1, 2, \dots, j-1\}$ ולכן $I \subseteq \{1, 2, \dots, j\}$. כמו כן, $w(K) \leq T' - w_j$ ולכן $w(I) = w(K \cup \{j\}) = w(K) + w_j \leq T' - w_j + w_j = T'$. כלומר הראינו כי I הוא פתרון חוקי עבור תת הבעיה (j, T') . כלומר $I \in \text{Sol}(j, T')$. בנוסף, $j \in I$ ולכן $I \in \{I \in \text{Sol}(j, T') : j \in I\}$.

כיוון שני: יהי $I \in \{I \in \text{Sol}(j, T') : j \in I\}$. בפרט $I \subseteq \{1, 2, \dots, j\}$. נגדיר $K = I \setminus \{j\}$. אזי $K \subseteq \{1, 2, \dots, j-1\}$. בנוסף, מכיוון ש- $j \in I$ מתקיים ש- $I = K \cup \{j\}$ וגם $w(K) = w(I \setminus \{j\}) = w(I) - w_j \leq T' - w_j$. כלומר $K \in \text{Sol}(j-1, T' - w_j)$. לכן $I \in \{K \cup \{j\} : K \in \text{Sol}(j-1, T' - w_j)\}$. מ.ש.ל. (טענה 2)

עכשיו נוכל להשלים את הוכחת (מקרה ג של) נוסחת המבנה:

$$\begin{aligned} OPT(j, T') &\stackrel{\text{הגדרת } OPT}{=} \max_{I \in \text{Sol}(j, T')} \{p(I)\} \stackrel{\text{אבחנה 0}}{=} \max \left\{ \max_{\substack{I \in \text{Sol}(j, T') \\ j \notin I}} \{p(I)\}, \max_{j \in I} \{p(I)\} \right\} \\ &\stackrel{\text{טענה 2 ואבחנה 1}}{=} \max \left\{ \max_{I \in \text{Sol}(j-1, T')} \{p(I)\}, \max_{K \in \text{Sol}(j-1, T' - w_j)} \{p(K \cup \{j\})\} \right\} \\ &\stackrel{\text{הגדרת } p(\cdot)}{=} \max \left\{ \max_{I \in \text{Sol}(j-1, T')} \{p(I)\}, p_j + \max_{K \in \text{Sol}(j-1, T' - w_j)} \{w(K)\} \right\} \\ &\stackrel{\text{הגדרת } OPT}{=} \max \{ OPT(j-1, T'), w_j + OPT(j-1, T' - w_j) \} \end{aligned}$$

מ.ש.ל. (נוסחת המבנה)

אז הוכחנו את נוסחת המבנה. עכשיו אנחנו רוצים להשתמש בה כדי לתכנן אלגוריתם שמחשב את $OPT(n, T)$, כלומר את ערך פתרון אופטימלי עבור n המוצרים הראשונים במשקל לכל היותר T .

אלגוריתם איטרטיבי לבעיה:

נגדיר מערך $M[j, T']$ עבור $j \in \{0, 1, 2, \dots, n\}$ ועבור $T' \in \{0, 1, 2, \dots, T\}$.

אתחול: לכל $0 \leq T' \leq T$ נקבע $M[0, T'] = 0$

צעד:

עבור $j = 1$ עד n בצע:

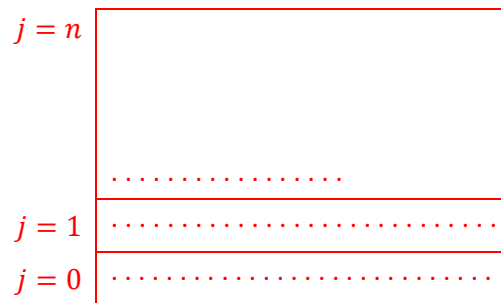
עבור $T' = 0$ עד T בצע:

אם $w_j \leq T'$ אזי נקבע $M[j, T'] = \max\{M[j - 1, T'], p_j + M[j - 1, T' - w_j]\}$

אחרת נקבע $M[j, T'] = M[j - 1, T']$

החזר: $M[n, T]$

בציור:



הוכחת נכונות של האלגוריתם האיטרטיבי:

משפט: לכל $0 \leq j \leq n$

לאחר סיום האיטרציה של הלולאה עבור j , לכל $0 \leq T' \leq T$ מתקיים $M[j, T'] = OPT(j, T')$

הוכחה: באינדוקציה על j

בסיס: עבור $j = 0$ הטענה נובעת מהאתחול

צעד האינדוקציה: נניח כי הטענה נכונה לכל $\ell < j$ ונוכיח עבור j .

יהי $0 \leq T' \leq T$ כלשהו, ונסתכל במהלך האיטרציה עבור T' (בתוך האיטרציה עבור j).

אם $w_j > T'$ אזי

$$M[j, T'] \underbrace{=}_{\text{האלגוריתם}} M[j - 1, T'] \underbrace{=}_{\text{ה.א.}} OPT(j - 1, T') \underbrace{=}_{\substack{\text{נוסחת המבנה} \\ \text{כאשר } T' < w_j}} OPT(j, T')$$

אם $w_j \leq T'$ אזי

$$M[j, T'] \underbrace{=}_{\text{האלגוריתם}} \max\{M[j - 1, T'], p_j + M[j - 1, T' - w_j]\}$$

$$\underbrace{=}_{\text{ה.א.}} \max\{OPT(j - 1, T'), p_j + OPT(j - 1, T' - w_j)\} \underbrace{=}_{\text{נוסחת המבנה}} OPT(j, T')$$

נשים לב כי תאים אלו לא מעודכנים עוד המבלך הריצה, ולכן שיויונות אלו נשארים עד לסוף הריצה.

ניתוח זמן ריצה של האלגוריתם האיטרטיבי:

אתחול: $O(T)$

צעד: מספר הצעדים הוא $O(n \cdot T)$ ועלות כל צעד היא $O(1)$.
סה"כ זמן ריצה $O(n \cdot T)$.

שאלה: האם זהו אלגוריתם פולינומי?

תשובה: לא. אלגוריתם הוא פולינומי אם זמן הריצה שלו פולינומי בגודל ייצוג הקלט.
כדי לייצג את המספר T דרושים $\log T$ ביטים.
לכן האלגוריתם לא פולינומי בגודל הקלט.

דוגמה: נניח T מיוצג על ידי 128 ביטים (אזי $T \leq 2^{128} - 1$) ונניח $n = 100$.
זמן הריצה של האלגוריתם יהיה $100 \cdot 2^{128}$. זה זמן ריצה שלא יסתיים בחיים... אפילו לא ברשת מחשבים...

שאלה: אז מתי האלגוריתם כן יעיל?

תשובה: כאשר המשקלים קטנים אז האלגוריתם יעיל, אבל עבור משקלים גדולים הוא לא יעיל. למשל, אם $T \leq n$
אזי זמן הריצה הוא $O(n^2)$.

שאלה: האם קיים אלגוריתם יעיל גם עבור משקלים גדולים?

תשובה: לא ידוע אלגוריתם יעיל (ואפילו מאמינים שאין כזה).

אלגוריתם לשחזור פתרון:

אלגוריתם השחזור שנראה מבוסס על 2 האבחנות הבאות:

אבחנה 3: אם $OPT(j, T') = OPT(j - 1, T')$ אזי קיים פתרון אופטימלי לתת הבעיה (j, T') שלא מכיל את j
לכן, במקרה זה, אם מבקשים ממני לשחזר פתרון אופטימלי עבור תת הבעיה (j, T') , אז אני יכול (ברקורסיה)
להחזיר פתרון אופטימלי לתת הבעיה $(j - 1, T')$

אבחנה 4: אם $OPT(j, T') > OPT(j - 1, T')$ אזי כל פתרון אופטימלי לתת הבעיה (j, T') מכיל את j

מנוסחת המבנה אנחנו יודעים כי אם $OPT(j, T') \neq OPT(j - 1, T')$ אזי
 $OPT(j, T') = p_j + OPT(j - 1, T' - w_j)$

לכן, במקרה זה, אם מבקשים ממני לשחזר פתרון אופטימלי עבור תת הבעיה (j, T') , אז אני יכול (ברקורסיה)
להחזיר פתרון אופטימלי לתת הבעיה $(j - 1, T' - w_j)$ ולהוסיף לפתרון זה את המוצר j (כמו שראינו
בהוכחה של נוסחת המבנה, הפתרון המתקבל חוקי).

Reconstruct(j, T')

- If $j = 0$ then return \emptyset
- If $M[j, T'] \neq M[j - 1, T']$ then return $\{j\} \cup \text{Reconstruct}(j - 1, T' - w_j)$
- Else return $\text{Reconstruct}(j - 1, T')$

משפט: נניח כי לכל $0 \leq j \leq n$ ולכל $0 \leq T' \leq T$ מתקיים $M[j, T'] = OPT(j, T')$. אזי $Recunstruct(n, T)$ מחזיר פתרון אופטימלי לבעיית התרמיל, כלומר מחזיר פתרון $I \in Sol(n, T)$ במחיר $p(I) = OPT(n, T)$.

הוכחה:

נוכיח באינדוקציה על j את הטענה הבאה:
לכל $0 \leq T' \leq T$, קריאה ל- $Recunstruct(j, T')$ מחזירה פתרון $I \in Sol(j, T')$ במחיר $p(I) = OPT(j, T')$.

בסיס: $j = 0$. לכל $0 \leq T' \leq T$ מתקיים כי הפתרון היחיד ב- $Sol(0, T')$ הוא הפתרון הריק. מהגדרת האלגוריתם מוחזרת קבוצה ריקה.

הנחת האינדוקציה:

נניח כי לכל $\ell < j$ מתקיים:
לכל $0 \leq T' \leq T$, קריאה ל- $Recunstruct(\ell, T')$ מחזירה פתרון $I \in Sol(\ell, T')$ במחיר $p(I) = OPT(\ell, T')$.

צעד האינדוקציה: נראה את הטענה עבור j .

נקבע $0 \leq T' \leq T$ כלשהו. עלינו להראות שקריאה עבור $Recunstruct(j, T')$ מחזירה פתרון אופטימלי עבור תת הבעיה (j, T') .

מקרה א: $M[j, T'] \neq M[j - 1, T']$. אזי, כפי שכתבנו בטקסט האדום מתחת לאבחנה 4, מתקיים

$$M[j, T'] = OPT(j, T') = p_j + OPT(j - 1, T' - w_j) = p_j + M[j - 1, T' - w_j] \quad ((1))$$

במקרה זה $Recunstruct(j, T')$ מחזיר $I = \{j\} \cup Recunstruct(j - 1, T' - w_j)$

נסמן $K = Recunstruct(j - 1, T' - w_j)$. לפי הנחת האינדוקציה, K הוא פתרון אופטימלי עבור תת הבעיה $(j - 1, T' - w_j)$, כלומר $K \in Sol(j - 1, T' - w_j)$ ו- $p(K) = OPT(j - 1, T' - w_j)$.

האלגוריתם מחזיר את $I = K \cup \{j\}$. מטענה 2 אנחנו יודעים כי $I \in Sol(j, T')$ ולכן I פתרון חוקי. מחירו של I הוא:

$$p(I) = p(K \cup \{j\}) = p(K) + p_j = OPT(j - 1, T' - w_j) + p_j = OPT(j, T')$$

כאשר השיוויון האחרון נובע מ- $((1))$

מקרה ב: $M[j, T'] = M[j - 1, T']$. במקרה זה $Recunstruct(j, T')$ מחזיר $I = Recunstruct(j - 1, T')$ לפי הנחת האינדוקציה

$$I \in Sol(j - 1, T') \subseteq Sol(j, T')$$

וגם

$$p(I) = OPT(j - 1, T') = M[j - 1, T'] = M[j, T'] = OPT(j, T')$$

מ.ש.ל.

ניתוח זמן ריצה:

הפרוצדורה $Recunstruct(j)$ מבצעת קריאה רקורסיבית אחת עם ערך ℓ , עבור $\ell < j$. אז סה"כ יש לכל היותר n קריאות רקורסיביות. כל קריאה (ללא עלות הקריאות הרקורסיביות במהלכה) היא בזמן $O(1)$. לכן סה"כ זמן ריצה (לא כולל חישוב המטריצה M) הוא $O(n)$.