תזכורת: בעיית הפעילויות הממושקלות

- מופע: קטעים $\{(s_i, f_i)\}_{i=1}^{n}$ עם משקלים $\{w_i\}_{i=1}^{n}$.
- יש למצוא: קבוצת קטעים זרים $I \subseteq \{1,\dots,n\}$ עם משקל $w(I)=\sum_{i\in I} w_i$ מקסימלי.

שלב 1: פירוק לתת-בעיות

[אנו מניחים שהקטעים ממוינים לפי זמני הסיום שלהם, כלומר $f_1 \le \cdots \le f_n$.]

הגדרנו:

- $T(k)=\{1,\dots,k\} \qquad \forall k\in\{1,\dots,n\}$
- $OPT(k)=$ ערך פתרון טוב ביותר שמוכל ב-$T(k)$
- $OPT(0)=0$
- $p(k)=$ מס' הקטע האחרון שמסתיים לכל המאוחר ב-s_k

שלב 2: בניית נוסחה

- טענה 1: פתרון טוב ביותר שמוכל ב-$T(k)$ **ולא** כולל את k – ערכו $OPT(k-1)$.
- טענה 2: פתרון טוב ביותר שמוכל ב-$T(k)$ **וכולל** את k – ערכו $w_k+OPT(p(k))$.

משתי הטענות הללו הסקנו כי:

$$OPT(k)=\begin{cases} 0, & k=0 \\ \max\{OPT(k-1), w_k+OPT(p(k))\}, & k>0 \end{cases}$$

מימושים אפשריים

1. רקורסיה נאיבית:

נקרא ל-$\text{Recurse}(n)$, כאשר $\text{Recurse}(k)$ מוגדרת כך:

- אם $k=0$, נחזיר 0
- אחרת, נחשב ונחזיר:

$$\max\{\text{Recurse}(k-1), w_k+\text{Recurse}(p(k))\}$$

מה זמן הריצה של אלגוריתם זה?

[עץ קריאות]

במקרה הגרוע ביותר [[אם הקטעים לא נחתכים כלל]] נקבל כ-2^n קריאות רקורסיביות.

2. אלגוריתם Bottom-Up / איטרטיבי:

- אתחול:
  - מיון
  - חישוב $p(k)$ לכל $k=1,\ldots,n$
  - נגדיר מערך M עם כניסות $0,\ldots,n$
- לולאה מרכזית:

  עבור $k=0$ ועד n:
  - אם $k=0$, נבצע: $M[k]\leftarrow 0$
  - אם $k>0$, נבצע:

$$M[k]\leftarrow \max\{M[k-1], w_k+M[p(k)]\}$$

- סיום:
  - נחזיר את $M[n]$

[נטען שאלגוריתם הזה מחזיר את $OPT(n)$ – ולמעשה שהוא מחשב את $OPT(k)$ לכל $k\in\{0,\ldots,n\}$. בעזרת השלב הבא נסביר למה.]

שלב 3: בחירת סדר על התת-בעיות

- נבחר את הסדר הטבעי, $0,1,\ldots,n$.
- לפי סדר זה, הפרמטרים שמופיעים בצד ימין ($p(k)$ ו-$k-1$) באים לפני הפרמטרים שמגדירים את התת-בעיה k (כלומר: $k-1, p(k)<k$).

נכונות האלגוריתם

נראה באינדוקציה ש-$M[k]=OPT(k)$ אחרי ההשמה ה-k, לכל $k=0,\ldots,n$.

בשלב ה-k, לפי הנחת האינדוקציה:

$$M[k-1]=OPT(k-1)$$
$$M[p(k)]=OPT(p(k))$$

לכן:

$$\max\{M[k-1], w_k+M[p(k)]\}=\max\{OPT(k-1), w_k+OPT(p(k))\}=OPT(k)$$

לפי הנוסחה

שחזור פתרון אופטימלי בהינתן [ ]M

מהאלגוריתם קיבלנו תשובה מספרית (משקל פתרון אופטימלי).
כעת נרצה לשחזר מתוך המידע שהאלגוריתם מייצר את הקטעים עצמם.

[[הסבר מקדים: בשלב k בריצת האלגוריתם, ניתן לדעת אם אפשר לבחור את k כחלק מפתרון אופטימלי לפי האם תוצאת ה-max שמבצעים שווה ל-$w_k + M\left[p(k)\right]$ (אם כן, ניתן לקחת אותו; אם לא, לא ניתן לקחת אותו). לכן תוך ריצת האלגוריתם ניתן גם לבנות את קבוצת הקטעים בכך שבכל פעם שניתן לבחור קטע, נעשה זאת. למעשה ניתן גם לעשות זאת כלאחר מעשה, וזה מה שנעשה – זה אולי ייקח יותר זמן, אבל יותר קל לנו לכתוב זאת בנפרד.]]

<u>אבחנה</u>: קיים פתרון אופטימלי ב-$T(k)$ שכולל את $k \Leftrightarrow OPT(k) = w_k + OPT(p(k))$.
זאת לפי טענה 2 ולפי הוכחת הנכונות המראה שהנ"ל מתקיים אם"ם $M[k] = w_k + M\left[p(k)\right]$.

<u>אלגוריתם לבניית קבוצת הקטעים</u>:

- $I \leftarrow \varnothing, k \leftarrow n$
- כל עוד $k \neq \varnothing$:
  - אם $M[k] = w_k + M\left[p(k)\right]$ אז:
    - $I \leftarrow I \cup \{k\}$
    - $k \leftarrow p(k)$
  - אחרת:
    - $k \leftarrow k-1$

חישוב זמן ריצה

- חישוב $OPT(n)$:
  - לולאה מרכזית: $n+1$ צעדים שלוקחים $O(1)$ – סה"כ $O(n)$
  - מיון: $O(n\log n)$
- חישוב הערכים $p(k)$: ניתן לעשות זאת ע"י חיפוש בינארי של s_k בתוך טבלת ה-$(f_1,\dots,f_n)$ [[כי רוצים למצוא את הקטע j האחרון שמקיים $f_j \le s_k$]]
  - $\Leftarrow O(n\log n)$
- שחזור הפתרון: $O(n)$ [[קצת יותר יעיל להכניס את זה לאלגוריתם הראשי, אבל אין הבדל בסדרי גודל]]

בעיית הפירוק לגורמים

- <u>מופע</u>: מספר שלם n כך ש-$n = p \cdot q$ עבור מספרים ראשוניים p,q (המספרים p,q לא נתונים!)
- <u>יש למצוא</u>: p,q

אלגוריתם פשוט: לעבור על כל המספרים $k=1,\ldots,\sqrt{n}$ ולבדוק האם $n \mid k$.

זמן ריצה: $O\left(\sqrt{n}\right)$

[להוסיף הסבר]

למעשה אורך הקלט הוא $b \approx \log_2 n$.

לכן אם מספר הביטים הוא b, אז $2^{b-1} \le n < 2^b$, ואז הסיבוכיות היא למעשה:

$$O\left(\sqrt{2^b}\right)=O\left(2^{\frac{b}{2}}\right)$$

ועכשיו, למדבר!

בעיית התרמיל (Knapsack Problem)

[לרוב נקראת בארץ "בעיית הגנב".]

דוגמת הקדמה

תכנית מגירה נוספת של המרצה הרב-תחומי שלנו: מסע במדבר בתקווה לקבלת חוויה טרנסנדנטלית (transcendental).
הוא רוצה להכין צידה לדרך ע"מ לשרוד במסע.

להלן טבלת המוצרים שיש לו בבית:

מוצר	משקל בק"ג	קלוריות
חבילת סוכר	1	4000
שמן זית	0.4	3500
ארבעה תפוחי אדמה	0.2	150
	0.2	150
	0.2	150
	0.2	150
מים	1.5	0

יש לו מגבלה של 1.1 ק"ג לכל היותר, והוא רוצה לקבל כמות מקסימלית של קלוריות מבלי לעבור על המגבלה.

נציע אלגוריתם חמדן: נמיין לפי (קלוריות / ק"ג).

עבור הדוגמה הנ"ל נקבל בעזרת האלגוריתם את הבחירה הבאה:

1. שמן זית
2. תפוח אדמה
3. תפוח אדמה
4. תפוח אדמה

סה"כ: 3950 קלוריות.

אנו רואים שיכולנו לקחת את הסוכר לבד ולקבל תוצאה טובה יותר – לכן נראה שאלגוריתם זה לא עובד.

[ראינו כאן שהאלגוריתם החמדן הנ"ל לא עובד, אבל אם תמצאו אחד, יש על זה פרס של מיליון דולר (זה למעשה יענה את אחת השאלות הגדולות ביותר במדעי המחשב כיום, שהיא האם $NP = P$; נגיע אליה בהמשך הקורס).]

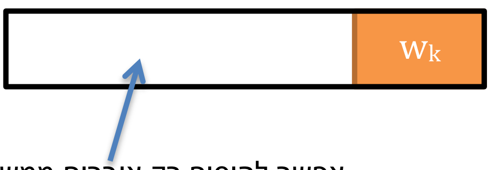
הגדרת הבעיה

- מופע:
  - n איברים $\{1,\dots,n\}$ עם משקלים $w_1,\dots,w_n$ וערכים $v_1,\dots,v_n$ שכולם שלמים ואי-שליליים
  - קיבולת / חסם W
- פתרון חוקי: קבוצת איברים $I \subseteq \{1,\dots,n\}$ עם משקל $w(I) = \sum_{i\in I} w_i \le W$.
- יש למצוא: קבוצה חוקית I עם ערך $v(I) = \sum_{i\in I} v_i$ מקסימלי.

רעיון שגוי

נמיין את האיברים $w_1 \le \cdots \le w_n$ ונגדיר $OPT(k) =$ ערך מקסימלי של פתרון חוקי $\supseteq \{1,\dots,k\}$.

- מקרה א': פתרון $\supseteq \{1,\dots,k\}$ שכולל את k.



אפשר להוסיף רק איברים ממשקל $\geq W - w_k$

טענה לא נכונה: במקרה זה, פתרון אופטימלי יהיה בעל ערך $OPT(t_k)$, כאשר:

$$t_k = \max\{t \mid w_t \le W - w_k\}$$

רעיון זה שגוי כיוון שהמשקל הכולל $OPT(t_k)$ יכול להגיע (מהגדרת OPT) עד ל-W, ואז יחד עם w_k יכול לחרוג מהחסם.

רעיון לא שגוי

שלב 1: פירוק לתת-בעיות

נגדיר:

- $\mathcal{T}(k,U) =$ פתרונות חוקיים מתוך $1,\dots,k$ ממשקל $U \geq$ $(U,k\geq 0)$, כלומר:

$$\mathcal{T}(k,u)=\left\{I\subseteq\{1,\dots,k\}\,\middle|\,w(I)\leq U\right\}$$

- $\mathcal{T}(0,U)=\varnothing$
- $OPT(k,U) =:$ ערך אופטימלי עבור $\mathcal{T}(k,U)$, כלומר:

$$OPT(k,U)=\max_{I\in\mathcal{T}(k,U)}\left(v(I)\right)$$

- $OPT(0,U)=0$

זיהוי הבעיה המקורית ומס' התת-בעיות

הבעיה המקורית = $OPT(n,W)$.

<u>מס' התת-בעיות:</u>

$$\left.\begin{aligned}k&\in\{0,\dots,n\}\\U&\in\{0,\dots,W\}\end{aligned}\right\}\Rightarrow(n+1)(W+1)=O(n\cdot W)$$

שלב 2: מציאת נוסחת נסיגה

טבלה!

k בפתרון?	נוסחה
לא	
כן	

- <u>מקרה א'</u>: פתרון $\mathcal{T}(k,U)$ **שלא** כולל את k.
    - <u>טענה 1</u>: פתרון אופטימלי מסוג זה, ערכו $OPT(k-1,U)$.

    <u>הוכחה</u>: מיידי מההגדרה של $OPT(\ )$, מכיוון שאי-בחירת k לא מגבילה את האפשרות לבחור פתרון חוקי מתוך $1,\dots,k-1$.

- <u>מקרה ב'</u>: פתרון $\mathcal{T}(k,U)$ **שכן** כולל את k.
    - <u>טענה 2</u>: פתרון אופטימלי כזה, ערכו הוא $v_k+OPT(k-1,U-w_k)$.

    <u>הוכחה</u>: בשיעור הבא.

לפי טענות 1+2, מקבלים:

$$OPT(k,U)=\max\left\{OPT(k-1,U),v_k+OPT(k-1,U-w_k)\right\}$$

<u>מקרי קצה</u>: $k=0$ וְ- $w_k > U$ [שכן צריך שיתקיים $k-1 \geq 0$ וְ- $U-w_k \geq 0$ כי $OPT(\ )$ מוגדרת רק עבור ערכים אי-שליליים].

נרשום טבלה מעודכנת:

$w_k \leq U$	k בפתרון?	נוסחה
כן	כן	$v_k + OPT(k-1, U-w_k)$
כן	לא	$OPT(k-1,U)$
לא	כן	אין פתרון כזה
לא	לא	$OPT(k-1,U)$

לכן:

$$OPT(k,U)=\begin{cases} 0, & k=0 \\ \max\{OPT(k-1,U), v_k + OPT(k-1,U-w_k)\}, & k>0 \wedge w_k \leq U \\ OPT(k-1,U), & k>0 \wedge w_k > U \end{cases}$$