20.12.16 בס"ד

אלגוריתמים - תרגול 7

אלגוריתמים חמדניים:

בעיית בחירת הפעילויות.

נתונה קבוצת פעילויות $S=\{a_1,a_2,\dots,a_n\}$ של n פעילויות שאינן יכולות להתבצע במקביל.

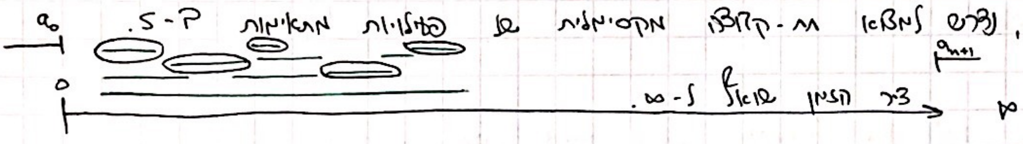
זמן ההתחלה של הפעילות a_i הוא s_i, וזמן הסיום שלה הוא f_i

כאשר $0\le s_i < f_i < \infty$. אם פעילות a_i נבחרת, היא מתבצעת במקטע החצי פתוח $[s_i,f_i)$

נאמר שבזוג פעילויות a_i ו- a_j הן <u>מתאימות</u> אם זמני הביצוע שלהם לא חופפים.

כלומר, מתקיים $f_i\le s_j$ או $f_j\le s_i$

בבעיית בחירת הפעילויות, נדרש למצוא תת-קבוצה מקסימלית של פעילויות מתאימות ב- S.



בדוגמא הזו המקסימום הוא 5 פעילויות.

פתרון -

נגדיר קבוצות S_{ij} באופן הבא: $S_{ij}=\{a_k\in S \mid f_i\le s_k<f_k\le s_j\}$

כלומר, S_{ij} הוא קבוצה של הפעילויות שמתחילות לאחר שהפעילות a_i הסתיימה ולפני שהפעילות a_j התחילה.

כדי לייצג את הבעיה המקורית, נוסיף זוג פעילויות פיקטיביות a_0 ו- a_{n+1} כאשר $f_0=0$ ו- $s_{n+1}=\infty$.

נגדיר את הקבוצה A_{ij} להיות תת הקבוצה המקסימלית של פעילויות מתאימות ב- S_{ij}

נרצה למצוא את $A_{0\,n+1}$ שהיא תת הקבוצה המקסימלית של פעילויות מתאימות ב- $S_{0\,n+1}$

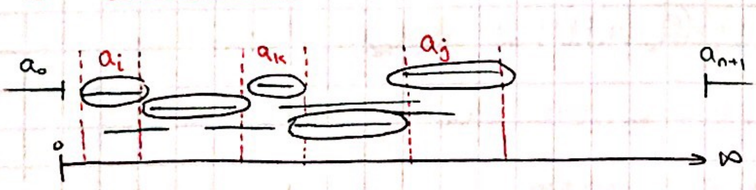
תהי a_k פעילות שנמצאת בפתרון אופטימלי עבור S_{ij}

אז: (נוכל בעצם לחלק את העולם לשניים)

$$\boxed{\begin{aligned} A_{ij} &= A_{ik}\cup\{a_k\}\cup A_{kj} \\ A_{ik} &= S_{ik}\cap A_{ij} \\ A_{kj} &= S_{kj}\cap A_{ij} \end{aligned}}$$

כאשר,

נסתכל על הדוגמה הבאה:



בס"ד

<u>נימוק:</u> הפתרון האופטימלי A_{ij} עבור S_{ij} חייב להכיל את A_{ik} ו- A_{kj} עבור S_{ik} ו- S_{kj} בהתאמה. אם נניח לשלילה שקיים פתרון אופטימלי אחר A'_{ik} עבור S_{ik} שמכיל כמות פעולות גדולה יותר מ- A_{ik} אזי ניתן היה לצרף אליו את a_k ואת הפעולות ב- A_{kj} ולקבל קבוצת פעולות מתאימות ב- S_{ij} שגדולה מ- A_{ij} בסתירה לאופטימליות.

לאור זאת, נגדיר פונק' $C[i,j]$ להיות המס' המקסימלי של פעולות מתאימות ב- S_{ij} מתקיים -

$$C[i,j]=\begin{cases} 0 & , S_{ij}=\emptyset \\ \max\limits_{k\in S_{ij}}\left(C[i,k]+C[k,j]+1\right) & , S_{ij}\neq\emptyset \end{cases}$$

פתרון באמצעות תכנון דינמי דורש $O(n^3)$ זמן (יש n^2 תאים בטבלה ולכל אחד מהם לוקח $O(|S_{ij}|)$ זמן כאשר $|S_{ij}|$ חסום ב- n)

סיבוכיות מקום $O(n^2)$.

האם ניתן לפתור באופן יותר יעיל? כן.

הרעיון: נבצע בחירה חמדנית של a_k כאשר בכל שלב נבחר את הפעולות שמסתיימת ראשונה בקבוצה.

הוכחת הבחירה החמדנית - נגדיר S_k כך: $S_k=\{a_j\in S \mid f_k\leq s_j\}$

כלומר כל הפעולות שמתחילות אחרי ש- a_k מסתיים.

<u>משפט:</u> תהי קבוצה S_k לא ריקה של קבוצות, ותהי a_m הפעולות שמסתיימת ראשונה ב- S_k, הפעולות a_m מוכלת בתת הקבוצה המקסימלית של פעולות מתאימות ב- S_k.

הוכחה -

תהי A_k תת קבוצה מקסימלית של פעולות מתאימות ב- S_k ותהי a_j הפעולות שמסתיימת ראשונה ב- A_k.

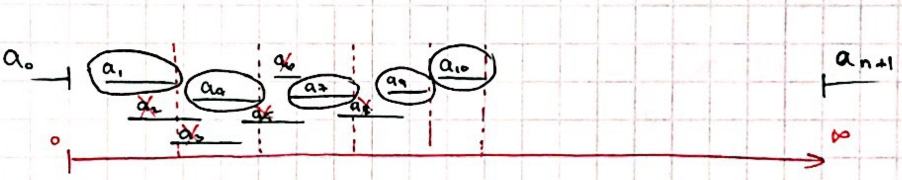
אם $a_j=a_m$ סיימנו.

אחרת, נגדיר $A'_k=(A_k\setminus\{a_j\})\cup\{a_m\}$, כלומר A'_k הינו A_k לאחר שהורדנו את a_j והוספנו את a_m. ברור ש: $|A'_k|=|A_k|$ (כי החלפנו איבר באיבר)

כמו כן, כל הפעולות ב- A'_k מתאימות כי הפעולות a_j היא הפעולות שמסתיימת ראשונה ב- A_k והחלפנו אותה בפעולות שמסתיימת לכל המאוחר באותו הזמן, לכן יתר הפעולות ב- A_k שאינן a_j a_m בהכרח מתאימה להן. לפיכך קיים A'_k אופטימלי שמכיל את a_m. $\square$

אלגוריתם חמדני רקורסיבי: (תחת ההנחה שהפעילויות ממויינות לפי זמני סיום בסדר עולה)

```
Recursive_Activity_Selection (s, f, k, n)
    m <- k + 1
    while (m <= n  and  S[m] < f[k])   -> אחראי לבטל את כל הפעילויות שחופפות לפעילות שבחרנו.
        m <- m + 1
    if (m <= n)
        return {a_m} U Recursive_Activity_Selection (s, f, m, n)   -> אם עדיין לא סיימנו נקרא לפונק בצורה רקורסיבית.
    else
        return Ø   -> תנאי עצירה של הרקורסיה.
```



זמן ריצה: $O(n)$ אם הפעילויות לא ממויינות לפי זמני סיום, צריך למיין ב-$O(n\log n)$.

בעיית התרמיל השברים:

כמו הבעייה משבוע שעבר רק שכאן ניתן לקחת שבר של פריט.

אלגוריתם חמדני:

(*) נחשב את הערך α של כל פריט a_i כאשר $\alpha_i = \frac{v_i}{w_i}$ (כלומר הערך שלו ליחידת משקל)

(*) ניקח כמות גדולה ככל הניתן מהפריט בעל ערך α המקסימלי.

(*) אם נגמרה לנו הכמות מפריט זה ונותר מקום בתרמיל נמשיך רקורסיבית לקחת מהפריט בעל ערך α המקסימלי מבין אלה שנותרו וחוזר חלילה עד למילוי התרמיל.

← תבנית הוכחת נכונות תהיה:

נסתכל על פתרון אופטימלי, יש 2 אפשרויות, או שהאיבר נמצא בו או שלא, אם הוא נמצא סיימנו אך אם לא, נוכל כנראה להחליף איפשהו איבר בתוך הפתרון באיבר שאנו טוענים שבהכרח נמצא בפתרון וכן להראות גם שהפתרון עדיין אופטימלי.

הוכחה נוספת צריכה להיות הוכחה של תת מבנה אופטימלי (בעצם על נוסחת הרקורסיה) ← שאם נחבר את שני תת המבנים האופטימליים נקבל עדיין מבנה אופטימלי.

וברגע שהוכחנו את שני הדברים הנ"ל הוכחנו נכונות של האלגוריתם.