פתרון ממ"ן 14 – נכתב על-ידי שי גרשטיין

<u>שאלה 1</u>

האלגוריתם

נבחר צומת כלשהו R, ונסתכל על העץ כמושרש בצומת R. נסמנו T.

לכל צומת X ב-T נסמן את גודל הזיווג המקסימלי בתת-העץ המושרש בו ב- $M(x)$, ואת גודל הזיווג המקסימלי בתת-העץ המושרש בו, כך ש-X אינו מכוסה בזיווג, ב- $N(x)$.

נעבור על כל צמתי T בסדר post-order, ועבור כל צומת X נפעל כך:

אם הוא עלה אז $M(x)=0$ ו- $N(x)=0$. אחרת, נסמן את קבוצת ילדיו ב-Y ונציב
$N(x)=\sum_{y\in Y} M(Y)$

ו- $M(x)=MAX\{N(x),\ 1+P(x)\}$

כאשר $P(x)$ מוגדר כך:

אם לפחות עבור אחד מילדי X מתקיים $M(y)=N(y)$ אז $P(x)=\sum_{y\in Y} M(y)$

אחרת, $P(x)=\sum_{y\in Y}(M(y))-1$

[במילים: הזיווג המקסימלי בעץ שמושרש ב-X הוא המקסימום בין $N(x)$ לבין 1+P(x), כאשר P(x) מוגדר להיות סכום ערכי ה- M של כל הבנים או הסכום הזה פחות 1, בהתאם לתנאי שפורט לעיל].

לבסוף נחזיר את $M(R)$.

זמן ריצה:

סריקת post-order בעץ היא $O(n)$. בנוסף, אנחנו עוברים על כל צומת פעמיים: פעם אחת בתור אבא ופעם אחת בתור בן. בכל מעבר אנו או בודקים את ערכו, או רושמים ערך חדש, כלומר פעולה בזמן קבוע. לכן זמן הריצה הוא $O(n)$.

הוכחת נכונות:

<u>טענה 1</u>: לכל צומת X ב- T מתקיים $0\le M(x)-N(x)\le 1$

הוכחת טענה 1: לפי הגדרה נקבל $N(x) \le M(x)$. אם הזיווג המקסימלי בתת-העץ של X לא מכסה את X, אז לפי הגדרה $N(x) = M(x)$. אחרת, נסיר מהזיווג המקסימלי את הקשת שמכסה את X, ונקבל זיווג שלא מכסה את X שגודלו $M(x)-1$, ולפי הגדרה נקבל $M(x)-1 \le N(x)$. ובכך כיסינו את כל המקרים האפשריים.

טענה 2: האלגוריתם מחשב נכון את הערכים N ו-M עבור כל צומת X ב-T.

הוכחת טענה 2: נוכיח באינדוקציה על מספר הצמתים בתת-העץ של X.

עבור צומת אחד, ברור שמתקיים $N(x) = M(x) = 0$.

נניח שהטענה נכונה עבור כל תת-עץ עם פחות מ-n צמתים.

נסתכל על עץ בן n צמתים המושרש ב-X. אם X אינו נכלל בזיווג המקסימלי אז אין כל הגבלה על הקשתות שיכולות להיבחר בתת-העצים של בניו, ולכן הזיווג המקסימלי הוא סכום הזיווגים המקסימליים של תת-העצים של בניו.

אחרת, אם X נכלל, הערך תלוי בבן שאליו מחוברת שהקשת שמכסה את X, אותו נסמן ב-y. בתת-העצים של שאר הבנים, ניתן לבחור את הזיווג המקסימלי, אבל עבור תת-העץ של y לא ניתן לבחור קשת שמכסה אותו ולכן, יש לבחור בזיווג שנותן את הערך $N(y)$. כלומר $M(x)$ יהיה מורכב מסכום ערכי M של כל הבנים, חוץ מאחד מהם שיתן את ערך ה- N שלו. לפי טענה 1, ערך זה יכול להיות קטן לכל היותר ב-1 מ-$\sum_{y \in Y} M(y)$. אם קיים בן עבורו ערך ה- M שווה לערך ה-N, אז ניתן לבחור זיווג עם קשת שמחברת את X לבן הזה, וסכום הזיווגים של העצים של בניו יהיה $\sum_{y \in Y} M(y)$. אחרת, כאמור, הסכום יהיה קטן מערך זה בדיוק ב-1. ולכל זה מוסיפים כמובן 1 בגלל הקשת שמכסה את X. כל הערכים שקשורים לתת-העצים של בניו של X, חושבו נכונה בגלל הנחת האינדוקציה.

האלגוריתם בוחר במקרה שנותן את הערך המקסימלי מבין השניים. ומכיוון שאלה 2 המקרים היחידים שאפשריים, האלגוריתם נותן את האפשרות שמתאימה לזיווג המקסימלי בתת-העץ של X.

לבסוף, האלגוריתם מחזיר את גודל הזיווג המקסימלי בעץ המושרש ב-R, אבל עץ זה הוא העץ כולו שניתן בקלט, ולכן מדובר בפלט הרצוי.

[הערה: אם מדובר ביער שמורכב ממספר עצים, אז ניתן להפעיל את האלגוריתם על כל אחד מהעצים ולסכום את כל הערכים שיתקבלו. אין בכך השפעה עקרונית שום דבר באלגוריתם וניתוחו]

שאלה 2

האלגוריתם:

נריץ סריקת BFS, כדי לחלק את הגרף לרכיבי קשירות, ולקבל את גודלם. נסמן את מספרם ב-K.

נעביר את גדלי הרכיבים כקלט לאלגוריתם Subset-Sum(k, n/2) שמוצג בספר. נחזיר "כן" אם"ם האלגוריתם מחזיר $n/2$, אחרת נחזיר לא.

זמן ריצה:

נסמן את מספר הקשתות ב-m. הזמן לסריקה הוא $O(m+n)=O(n^2)$.

זמן הריצה של אלגוריתם ת"ד לפתרון בעיית סכומי התת-קבוצות הוא $O(k\frac{n}{2})=O(kn)=O(n^2)$.

ולכן סך זמן הריצה הוא $O(n^2)$.

הוכחת נכונות

<u>טענה 1</u>: לא יתכן ש-2 צמתים באותו רכיב קשירות שייכים לקבוצות שונות.

<u>הוכחת טענה 1</u>: נניח בשלילה שצומת X שייך ל-A, וצומת Y שייך ל-B, ושניהם באותו רכיב קשירות. כיוון שהם באותו רכיב, קיים ביניהם מסלול P. כיוון שקצוות המסלול שייכים לקבוצות שונות, בהכרח קיימת לפחות קשת אחת ב- P כך שקצה אחד שלה ב- A והשני ב-B. אבל זה בלתי אפשרי לפי התנאי השני בשאלה. קיבלנו סתירה, ולכן ההנחה הייתה שגויה.

<u>מסקנה 1</u>: כל אחת מהקבוצות מוכרחה להיות מורכבת מאיחוד של רכיבים, כך שכל רכיב קשירות בגרף נמצא בקבוצה אחת בדיוק.

<u>טענה 2</u>: אם קיימת קבוצה של רכיבים שסכום צמתיה הוא חצי מסך הצמתים, אז גודלה של הקבוצה השניה הוא זהה, וחלוקה זו היא פתרון מתאים. ולהפך.

<u>הוכחה</u>: נובעת ישירות ממסקנה 1.

<u>טענה 3</u>: קיימת קבוצה של רכיבים שסכום צמתיה הוא חצי מסך הצמתים אם"ם האלגוריתם Subset-Sum(k, n/2) מחזיר $n/2$.

<u>הוכחה</u>: נובע ישירות מנכונות האלגוריתם בספר.

מטענה 3 נובעת נכונות האלגוריתם.

<u>שאלה 3</u>

<u>האלגוריתם</u>:

נמיין בסדר יורד את כל הפריטים לפי ערכם. נעבור על הרשימה ונחלק את הפריטים לשתי רשימות A[1...k] ו-B[1...r] (כך ש- $r+k=n$), כאשר ברשימה A יופיעו ערכי הפריטים שמשקלם הוא W_1 וברשימה האחרת הערכים של הפריטים שמשקלם W_2. הרשימות כמובן עדיין ממוינות בסדר הפוך.

נקבע $A[0]=B[0]=0$. כמו כן $Sum=0$

כעת נהפוך את המערכים למערכי סכומים:

עבור i מ-1 עד K, נבצע: $A[i]=A[i]+A[i-1]$

עבור j מ-1 עד r, נבצע: $B[i]=B[i]+B[i-1]$

כעת עבור i מ-0 עד $MIN\{k, \left\lfloor \frac{W}{w_1} \right\rfloor\}$, נבצע:

$$j=\left\lfloor \frac{W-i\cdot w_1}{w_2} \right\rfloor$$

$$Sum=Max\{Sum,\ \ A[i]+B[j]\}$$

ונשמור גם את ה-i וה-j שנתנו את ה-SUM המקסימלי.

(שיפור: במקום לעבור באיטרציה בהכרח על רשימה A, נעבור על המערך הקצר יותר)

לבסוף נחזיר את Sum בתור הפתרון, ואם מבוקשת גם הקבוצה, אז ניקח את ה-i וה-j שנתנו את ה-SUM המקסימלי, ואת העותקים ששמרנו של הרשימות הממוינות, ונחזיר את i הפריטים הראשונים ברשימה A ו-j הפריטים הראשונים ברשימה B.

<u>זמן ריצה:</u>

זמן הריצה של המיון אם נשתמש במיון-מיזוג הוא $O(n\lg n)$.

זמן הריצה של המעבר על הפריטים כדי לחלקם לשתי קבוצות הוא $O(n)$.

הזמן לבניית מערכי הסכומים, בגלל שעוברים על כל תא רק פעם אחת הוא גם $O(k+r)=O(n)$.

כל איטרציה של הלולאה מתבצעת בזמן קבוע. ומספר האיטרציות הוא $O(k)$, ולכן זמן הריצה של הלולאה הוא בהכרח $O(n)$.

ומכאן סך זמן הריצה של האלגוריתם כולו הוא $O(n\lg n)$.

<u>הוכחת נכונות:</u>

<u>טענה 1</u>: לקראת ביצוע הלולאה האחרונה, התא ה-A[i] מכיל את סכום ערכי i הפריטים היקרים ביותר ב-A. כנ"ל לגבי B.

<u>הוכחה</u>: נוכיח באינדוקציה על i. נוכיח עבור A (ההוכחה עבור B זהה).

עבור i=0: כמובן שהטענה מתקיימת כי $A[0]=0$.

נניח שהטענה נכונה עבור i<n. נוכיח עבור i=n.

האלגוריתם נבצע את הפעולה $A[n]=A[n]+A[n-1]$. לפי הנחת האינדוקציה $A[n-1]$ זה סכום n-1 הפריטים היקרים ביותר, ויחד עם הפריט ה-n היקר ביותר שהוא הערך שהיה ב- A[n], נקבל ש- A[n] מכיל את סכום ערכי n הפריטים היקרים ביותר ב-A.

בפתרון האופטימלי יכולים להיות לכל הפחות 0 פריטים מ- A ולכל היותר $MIN\{k, \left\lfloor \frac{W}{w_1} \right\rfloor\}$ פריטים מ-A. אם הפתרון האופטימלי מכיל X פריטים מ-A, אז אלה בהכרח X הפריטים היקרים ביותר בקבוצה, כי אחרת ניתן להחליף את הפריטים בפריטים היקרים ביותר, ולקבל קבוצת פריטים יקרה יותר באותו משקל. וכנ"ל לגבי B. בהינתן שנבחרו i פריטים מ-A, ברור שצריך לבחור $j=\left\lfloor \frac{W-i\cdot w_1}{w_2} \right\rfloor$ פריטים מ-B, אחרת, ניתן להוסיף פריט נוסף מ- B כך שהערך הכולל יגדל ולא תופר מגבלת המשקל.

לכן, אלגוריתם שעובר על כל מספר i אפשרי של פריטים מ-A ו-j מתאים ב- B עובר על כל המועמדים לפתרון האופטימלי, וע"י שמירת המקסימום, יחזיר בהכרח אותו. ובהתאם לטענה 1, זה בדיוק מה שהאלגוריתם עושה.

<u>שאלה 4</u>

<u>כיוון 1</u>: לכל i, $M(i,i)\geq 0$ <== אין בגרף מעגל עם משקל שלילי.

הוכחה:

נניח בשלילה שקיים בגרף לפחות מעגל שלילי אחד. נסתכל על מעגל C שהאינדקס הכי גבוה של צומת בו הוא המינימלי מבין כל המעגלים השליליים. ונסמן את האינדקס ב-X.

האלגוריתם עדיין מחשב נכון את כל הערכים עבור כל האיטרציות בהן K<X, כיוון שהנוסחה 6.4 במדריך תקפה גם כאן (בגלל אותם הנימוקים המופיעים במדריך). כלומר האלגוריתם עדיין מחשב נכון את המרחקים כאשר לא נלקחים בחשבון מסלולים שקיים בהם צומת פנימי שאינו קטן מ-X.

החל מהאיטרציה ה-X ניתן לעבור דרך הצומת X ולכן ניתן לראשונה ליצור מסלול שהוא מעגל שלילי, ואז נוסחה 6.4 לא מתאימה, כי במסלול לא מספיק לחשב רק את ערך M של הרישא עד צומת K ועוד ערך M של הסיפא מצומת K, אלא גם יש לקחת בחשבון את מסלולי הביניים מ- K לעצמו בין הרישא והסיפא, כיוון שכעת מסלולים אלה יכולים להיות שליליים.

באיטרציה ה- $k = x$ של הלולאה החיצונית באלגוריתם נסתכל על צומת i כלשהו על C.

$M(i,x,x-1)+M(x,i,x-1)$ זה סכום המשקלים של שני חלקים משלימים של C, לכן משקלם הוא משקלו של C, ומשקל זה הוא שלילי. כיוון שהאלגוריתם מצמיד ל- $M(i,i,k)$ את המינימלי מבין שני הערכים שהוא בוחן, ערכו יהיה כעת שלילי. היות וגם באיטרציות העוקבות, יבחר הערך המינימלי, ואחד המועמדים הוא הערך השלילי הקודם, הערך ישאר שלילי לאחר כל איטרציה. ובסיום ריצת האלגוריתם יתקבל $M(i,i)<0$ בסתירה לנתון. ולכן, ההנחה הייתה שגויה, ולא קיים בגרף מעגל שלילי.

כיוון 2: אין בגרף מעגל עם משקל שלילי ==> לכל i , $M(i,i)\geq 0$

הוכחה: מדובר בהרצה ה"תקנית" של פלויד-וורשל. ומנכונות האלגוריתם נובע שהערך המוחזר מייצג את המרחק של צומת לעצמו. ולא יתכן שהמרחק שלו מעצמו הוא שלילי (כי זה אומר שקיים מסלול של צומת אל עצמו שהוא שלילי. ומסלול כזה הוא מעגל. ונתון שאין מעגל עם משקל שלילי).

שאלה 5

נגדיר $M(t,i)$ בתור משקל המסלול הקל ביותר מ- V אל t המשתמש בדיוק ב-i קשתות.

נגדיר $N(t,i)$ בתור מספר המסלולים הקצרים ביותר מ- V אל t המשתמשים בדיוק ב-i קשתות.

נסמן ב- $W_{t,s}$ את משקל הקשת מ- t אל s.

האלגוריתם

נציב $M(v,0)=0$, $N(v,0)=1$, ולכל $u \neq v$ נציב $M(u,0)=\infty$, $N(u,0)=0$.

לכל $1 \le i \le n-1$ נעבור על כל הצמתים בגרף ועבור כל צומת u נבחן כל צומת p שיוצאת ממנו קשת אל u, ונציב $M(u,i)=\underset{p}{Min}\{M(p,i-1)+W_{p,u}\}$

וכעת נעבור על כל שכני u שמהם אפשר להגיע עם במסלול באורך הקצר ביותר שזה עתה מצאנו, ונציב: $N(u,i)=\sum_{p,\ M(u,i)=M(p,i-1)+W_{p,u}} N(p,i-1)$.

לבסוף נגדיר $S=\min_i\{M(w,i)\}$ ועבור כל i כך ש- $M(w,i)=S$ נצרף את ערך ה- N שלו לסכום המבוקש. כלומר, נחזיר $\sum_{i,\ M(w,i)=S} N(w,i)$

זמן ריצה:

כמקובל, מספר הצמתים מסומן ב-n, ומספר הקשתות ב-m.

החישוב של S דורש בחינה של n ערכים, והחישוב של הערך המוחזר דורש סכימה של לכל היותר n ערכים. ולכן החלק הזה של האלגוריתם רץ ב- $O(n)$.

עבור כל i אנו בוחנים כל קשת פעם אחת. עבור כל קשת אנו מבצעים פעולות בזמן קבוע. ול-i יש n ערכים שונים. ולכן זמן הריצה הוא $O(mn)$. וזהו גם זמן הריצה הכולל של האלגוריתם.

הוכחת נכונות:

כיוון שבאפס קשתות ניתן להגיע מצומת רק אל עצמו ע"י המסלול הריק, מתקבלים הערכים $M(v,0)=0$, $N(v,0)=1$, ולכל $u \neq v$ $M(u,0)=\infty$, $N(u,0)=0$.

נוסחת הנסיגה עבור M היא $M(u,i)=\underset{p}{Min}\{M(p,i-1)+W_{p,u}\}$, כיוון שעבור כל מסלול באורך i אל u יש רישא באורך i-1 אל צומת שממנו יש קשת אל u. ולכן אנו בוחנים את כל האפשרויות למסלולים כאלה, ובוחרים את המינימלי, כלומר הקצר ביותר. והמשקל מחושב כמשקל המסלול עד השכן, שכבר חישבנו באיטרציה קודמת, ועוד משקל הקשת המחברת בין השכן לצומת היעד.

נוסחת הנסיגה עבור N היא $N(u,i)=\sum_{p,\ M(u,i)=M(p,i-1)+W_{p,u}} N(p,i-1)$, כי כל שכן שדרכו ניתן ליצור מסלול קצר ביותר אל צומת היעד, תורם מספר מסלולים כמספר המסלולים הקצרים ביותר אליו (כיוון שכל מסלול כזה בצירוף עם הקשת האחרונה נותן מסלול אחר קצר ביותר אל צומת היעד). ולכן אנו עוברים על כל השכנים המתאימים וסוכמים את מספר המסלולים הקצרים ביותר אליהם.

אין צורך לבדוק יותר מ- n-1 איטרציות, כי עבור i=n זה אומר בדיקת מסלולים עם n קשתות, כך שבהכרח ביקרנו בצומת כלשהו פעמיים, כלומר המסלול כולל מעגל. וכיוון שנתון שמשקלו של כל מעגל הוא חיובי ממש, אז הסרת המעגל מהמסלול תיצור מסלול קל יותר, כלומר לא יתכן שנקבל ערך אופטימלי.

לבסוף אנו מוצאים את ערכו של המסלול המינימלי ביותר אל W, בכך שאנו עוברים על כל הערכים עבור האורכים השונים האפשריים. ועבור כל אורך שיכול להפיק מסלול קצר ביותר, כבר חישבנו את מספר המסלול הקצרים ביותר באורך הזה, אז נותר רק לסכום אותם, וסכום זה הוא בדיוק מספר המסלולים הקצרים ביותר אל W (שכמו כל המסלולים שבדקנו, מתחילים ב-V).

<יפה מאוד>