1 .a $insert(x)$

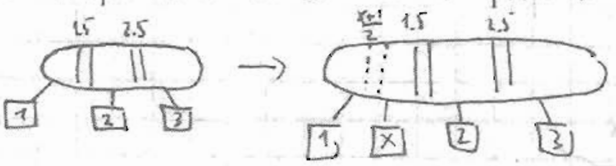
10/10 (לסעיף a)

1. נעלה במעלה ה- spline הימני והשמאלי במקביל (זיג - זג).
עד שנגיע לצומת v כך ש:
אם $v \in$ left spline, x קטן מהמפתח השמאלי של $p(v)$
אם $v \in$ right spline, x גדול מהמפתח הימני של $p(v)$

2. נרד במסלול חיפוש אחר צומת x, עד שנגיע לצומת u שכל בניו עלים.

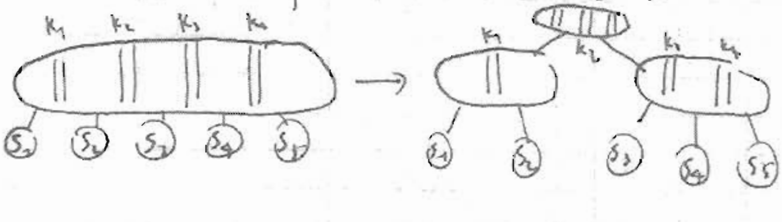
3. נוסיף את x במקום המתאים (בין עלה שגדול ממנו לעלה שקטן ממנו או כעלה השמאלי ביותר או כעלה הימני ביותר). ונוסיף מפתח ב-u בהתאם עם הערך הממוצע של x ועלה הקרוב אליו (שאין ביניהם מפ').

$X = 1.25$



אם מספר הבנים של u נשאר קטן מ-5 אז סיימנו.

4. מספר הבנים של u הוא 5, אזי נפצל את u לשני צמתים: צומת w עם שני בנים וצומת z עם 3 בנים. כאשר המפתח השמאלי ביותר ושני הבנים השמאליים ביותר ילכו ל w, שני המפתחות הימניים ביותר ושלושת הבנים הימניים ביותר ילכו ל z, והמפתח השני יעלה לאב של u ויפריד עכשיו בין w ל z.



אם האבא של u נשאר קטן מ-5 אז סיימנו אחרת נמשיך רקורסיבית את סעיף 4.

אם u הוא שורש נבצע את הפיצול, וניצור צומת חדש שיהיה השורש והמפתח שלו יהיה k_2.

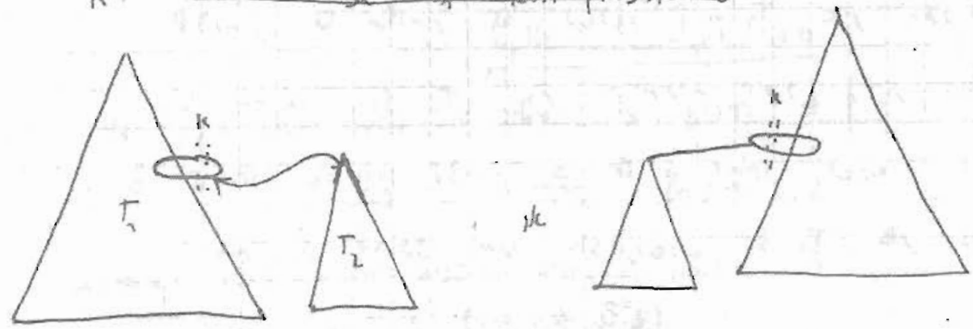
לעדכן את ה-finger בצורה הנכונה

concatenate(T_1,T_2)

1. נלך במעלה ה spine הימני של T_1 וה spine השמאלי של T_2 במקביל (צעד - צעד). עד שנגיע לשורש של הראשון (הנמוך מביניהם), ולצומת v בעץ השני, נצטרך לשמור את הנמוך לצומת p(v).

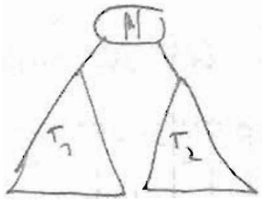
2. נוסיף את הנמוך ל p(v) (בעזרת insert) כאשר המפתח k שנוסיף ל- p(v) יהיה הממוצע בין האיבר הימני של T_1 לאיבר השמאלי של T_2.

~~אם הנמוך הוא T_1 אז - k~~



אם p(v) קטן מ-2 סיימנו, אחרת נפצל רקורסיבית כמו בהכנסה.

אם T_1 ו T_2 באותו גובה אז ניצור צומת חדש שיהיה בשורש עם אותו המפתח



:split(x)

1. נחפש את x כמו ב $insert(x)$. עד שנגיע לצומת u שכל בניו עלים
2. נאתחל את T_1 ו- T_2 להיות ריקים.
3. נשרשר* ל- T_1 את האלמנטים הקטנים מ-x ול T_2 את הגדולים שווים ל-x.
4. נעלה חזרה במסלול החיפוש עד ל-שורש. כאשר כל פעם שעולים לצומת w מצומת u נשרשר* ל- T_1 את הבנים השמאליים יותר של w מ- u ול T_2 את הימניים יותר. (זאת הנחה שהבנים הם ממוינים)
5. לאחר שהגענו לצומת z של ה spine ושרשרנו ל T_1, T_2 את בניו
אם ה spine ימני אז נשרשר ל T_1 את העץ T לאחר שמוחקים ממנו את z (זאת העץ של z).
אם ה spine הוא שמאלי אז נשרשר ל T_2 את העץ T ללא מחיקת z.

* שרשור כבר מימשנו קודם מלמעלה (כלומר נניח מימוש עד מבואות נקודת בסיס
ניתן להחזיק בשורש את גובה העץ כדי לאפשר זאת.

1.6.1

10/10

סיבוכיות - Concatenate

1. לרדת במעלה ה spine עד מציאת V והשורש - $O(\log n)$
2. הוספת החוליה והמצביע או הוספת צומת שורש במקרה שנדרש
   זה - $O(1)$
3. פיצול רקורסיבי עד לשורש במקרה הצורך - $O(h) = O(\log n)$

סה"כ: $O(\log n)$

סיבוכיות - Split

1. חיפוש x במעלה ה-spine ואז בתת העץ - $O(\log n)$
2. איתחול T_1, T_2 - $O(1)$
3. שירשור* האיברים הקטנים מ-x ל T_1 והגדולים ל-T_2 - $O(n)$
4. שירשור*: בשירשור* בין שני עצים T_1, T_2 'נשלם' על גובה בעץ משום שצריך לרדת בשיפוע מהם עד לגובה של הנמוך, להוסיף מצביע ולתקן צמתים גבוליים כלפי מעלה עד השורש. אזי $O(|h_1-h_2|)$ נשים לב שבשירשור הזה שצריך לשרשר עם T_1 או עם T_2 גבוה יותר ויותר, ומולאים השורש הוא עד בגובה של כגובה מבניהם. אזי נניח שבשלב הצ"ג חזרה ל- z נקבלנו ב- k_1 שרשורים ל- T_1 ו k_2 שרשורים ל- T_2, אזי:
   סך הכל השירשורים תהיה:
   $$\sum_{i=1}^{k_1} c_i(T_1) + \sum_{j=1}^{k_2} c_j(T_2) =$$
   $$= \sum_{i=1}^{k_1}[h(T_{ai}) - h(T_1^i)] + \sum_{i=1}^{k_2}[h(T_{bi}) - h(T_2^i)]$$
   כאשר T_1^i, T_2^i הם העץ T_1, T_2 לפני האיחוד ה-i ו T_{ai}, T_{bi} אלו העצים שמשורשרים* ל T_1^i, T_2^i בשלב ה-i בהתאמה.
   אבל הגובה של T_{ai}, T_{bi} הוא הגובה של T_1^{i+1}, T_2^{i+1} בהתאמה
   $$\sum_{i=1}^{k_1}[h(T_1^{i+1}) - h(T_1^i)] + \sum_{i=1}^{k_2}[h(T_2^{i+1}) - h(T_2^i)] = h(T_1^{k_1+1}) - h(T_1^1) + h(T_2^{k_2+1}) - h(T_2^1)$$
   טורים טלסקופיים
   $$= h(T_1^{k_1+1}) + h(T_2^{k_2+1}) = O(\log n)$$
   T_1', T_2' מקוריים
5. מחיקת z תיקח לכל היותר $O(\log n)$ אם מתקנים את השורש.
6. שרשור יחיד של T_1 או T_2 עם T ללא z כאחד שינויים $O(\log n)$

1.ב.2.

תחילה נראה שגם פעולת $insert^*(x)$ כאשר נתון לי מצביע למיקום שבו מכניס את x לוקח $O(1)$ amortized.

נשתמש בפונקציית פוטנציאל* שראינו בכיתה:

$$\Phi = 2\cdot\#4\text{-vertices} + \#2\text{-vertices}$$

נניח ביצענו k פיצולים בפעולת $insert^*$ ועודנו, אזי $ac(insert)$ ייקח זמן של k, נניח ואפילו פיצלנו את השורש אז $ac(insert) \le k + c$, עבור c קבוע כלשהו.

בכל פיצול היינו צומת של 4 ונהיו צומת של 2 וצומת של 3 אזי $\Delta\phi = -k$, אם נוסיף שורש מדרגה 2 אז $\Delta\phi = -k+1$

$$am(insert^*) = ac(insert^*) - \Delta\phi = k+c-k+1 = c+1 = O(1)$$

Concatenate:

1. לעלות במעלה ה-spine עד לגובה הנמוך מבין העצים: $am = O(\min(\log n_1, \log n_2)) = O(\log(\min(n_1,n_2)))$
2. לבצע $insert^*$ שאחרי יש לנו כבר מצביע ל $p(v)$ בו צריך להכניס את השורש הנמוך יותר, הראינו: $am = O(1)$

סה"כ: $am(concatenate) = O(\log(\min(n_1,n_2)))$

insert:

1. עולים במקביל משני הכיוונים, לכל צומת שני בניו לשמאל ולכן ב-spine השמאלי נצטרך לעלות עד לגובה $O(\log d_i)$ וב-spine הימני נצטרך לעלות עד לגובה $O(\log(|T_i|-d_i))$

עוצרים באלגוריתם כשמגיעים לצומת הראשון שx בתת העץ שלו ולכן זה ייקח $O(\min(\log d_i, \log(|T_i|-d_i)))$, לעבור בתת העץ אל דרגים ייקח אותו דבר.

2. ברגע שהגענו לצומת בה צריך להכניס את x, הכנסתו שם זה לוקח $am(insert^*) = O(1)$

סה"כ: $am(insert) = O(\log(\min(d_i,|T_i|-d_i)))$

:Split

1. כמו בשלב הראשון של insert, חיפוש אחר x ייקח $O(\log(\min(d_i,|T_i|-d_i)))$

ולכן זה ה amortized וגם* כן שווה לכך.

2. הראינו בכיתה ב w.c של split שפרשור* הבנים ל T_1

ו T_2 עוקבת. $h(T_1^{k_1+1})\cdot h(T_2^{k_2+1})$, כאשר הקומה של

שניהם חסום ע"י בגובה של הצומת z של ה spine (בהו בפרשור*

האחרון מסוג זה) ולצומת זו היה הצומת הראשון שמצאנו ש x

בתת עץ שלה $\Leftarrow$ $O(\log(\min(d_i,|T_i|-d_i)))$.

3. מחיקת הצומת z מ- t

נניח כי יש בו k איחודים, אזי $ac(delete)\le k+c$ עבור c קבוע

כלומר, $\Delta\Phi=-2\cdot k$ (כי בכל צמתים שאיחדנו היו דרגה 2

וכבר לדרגה 3). $am(delete)=ac(delete)+\Delta\Phi\le k+c-2k\le O(1)$

4. נותר לאחד את הענפים של T ללא z עם T_1 או T_2 וזה

כפי שהראינו ב concatenate, $O(\log(\min(n_1,n_2)))$, T_2, T_1

הם תתי קבוצה של העץ ש z מוגש ולכן ~~[illegible]~~

$|T_1|,|T_2|\le\min(d_i,|T_i|-d_i)$ ולכן עושים של איחוד מבנה עם

T ללא z ועולה $O(\log(\min(d_i,|T_i|-d_i))$

סה"כ: $am(split(x_i))=O(\log(\min(d_i,|T_i|-d_i)))$

$\frac{10}{10}$

1. ג. הראינו שעלות $am(split)$ היא $O(\log(\min(d_i, |T_i|-d_i)))$

כלומר $O(\log d)$, $d \le \frac{n}{2}$.

לכן עלות סדרת פיצולים (amortized) ניתנת ע"י:

$$T(n) = T(n-d_i) + T(d_i) + O(\log d_i)$$

עבור קבוע c, $T(n) \le \max_{d \le \frac{n}{2}} [T(n-d) + T(d) + c \cdot \log d)]$

נוכיח באינדוקציה ש: $T(n) \le n - 1 - c \cdot \log n$:

בסיס: $n=1$: $T(1) \le 1 - 1 \cdot c \cdot \log 1 = 0$

פעולת split על עץ בגודל 1 עולה באופן ריק 0.

ולכן מתקיים.

הנחה: $T(n) \le n - 1 - c \cdot \log n$

צעד: $T(n+1) \le \max_{d \le \frac{n}{2}} [T(n+1-d) + T(d) + c \cdot \log d] \le$

$$\le \max_{d \le \frac{n}{2}} [n+1-d-1-\log(n+1-d) \cdot c + d - 1 - c \cdot \log d + c \cdot \log d]$$

$$= \max_{d \le \frac{n}{2}} [n-1-\log(n+1-d) \cdot c] \le -c \cdot \log(n+1) + n - 1 \le n - c\log(n+1)$$

אז: הראינו ש $T(n) = O(n)$

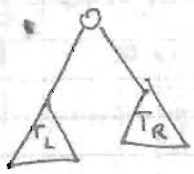
$$\sum ac(\overset{split}{\cancel{insert}}) \le \sum am(\overset{split}{\cancel{insert}}) + \phi_0 - \phi_n = O(n) + n$$

פ' הפוטנציאל $2 \cdot \#4\text{-}V + \#2\text{-}V$ היא לכל היותר n לכל הבחירות.

סה"כ עלות כל ה splits $O(n)$.

2. א. תחילה נבין מהו מחיר של עץ: 10/10

של T:



$$cost(T) = cost(T_L) + cost(T_R) + \sum_{i \in T} p_i$$

הוכחה: עומק כל איבר בתת העץ השמאלי/ימני גדל ב-1:

$$d_i^T = d_i^{T_L} + 1 \quad \text{or} \quad d_i^T = d_i^{T_R} + 1$$

$$cost(T) = \sum_{i \in T} p_i \cdot d_i^T = \sum_{i \in T_L} p_i \cdot d_i^T + \sum_{i \in T_R} p_i \cdot d_i^T = \sum_{i \in T_L} p_i \cdot (d_i^{T_L} + 1) + \sum_{i \in T_R} p_i \cdot (d_i^{T_R} + 1)$$

$$= \sum_{i \in T} p_i + \sum_{i \in T_L} p_i \cdot d_i^{T_L} + \sum_{i \in T_R} p_i \cdot d_i^{T_R} = \sum_{i \in T} p_i + cost(T_L) + cost(T_R)$$

עץ T יהיה בנוי בצורה דומה לבינארית עם מפתח t כאשר כל המפתחות ב T_L יהיו קטנים מ-t וכל המפתחות ב T_R יהיו גדולים ממש מ-t. בחירה של t תחלק את האיברים של T באופן יחיד ל T_L ו T_R.

⊛ ולכן קבוצת כל העצים T עם האיברים i to j היא קבוצת כל צמדי הקבוצות הזרות T_L ו T_R המחולקים ע״י t כך ש $i \le t \le j$.

מחיר עץ אופטימלי:

$$\min_{\substack{i \in T \\ \{d_i\}}} [cost(T)] = \min_{\{d_i | i \in T\}} \left[\sum_{j \in T} p_j + cost(T_L) + cost(T_R)\right] = \sum_{j \in T} p_j + \min_{\{d_i | i \in T\}} [cost(T_L) + cost(T_R)] =$$

אפשר להוציא קבוע מהמינימום.

עפ״י

$$⊛ = \sum_{j \in T} p_j + \min_{i \le t \le j} \left[\min_{\{d_i | i < t\}} [cost(T_L)] + \min_{\{d_i | i > t\}} [cost(T_R)] \right] = \sum_{j \in T} p_j + \min_{i \le t \le j} \left(cost(optT_L) + cost(optT_R) \right).$$

לאחר שהבנו את מחירו האופטימלי של עץ T, ניתן להראות ש-B תקין:

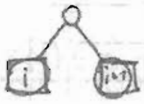
עכשיו נראה באינדוקציה ש- $B[i,j]$ הוא המחיר האופטימלי

לעץ שמחזיק את המפתחות $k_i,\dots,k_j$

באינדוקציה על $j-i=\alpha$:

בסיס: $\alpha=1$: $B[i,j]=B[i,i+1]=w(i,i+1)=p_i+p_{i+1}$

ועץ האופטימלי שמחזיק שני מפתחות הוא:



ומחירו (כל אחד מהם בעומק 1) הוא p_i+p_{i+1}.

צעד: נניח נכונות עבור $j-i\le\alpha$

עבור $j'-i'=\alpha+1$: $B[i',j']=w(i',j')+\min_{i'\le t\le j'}(B_{i',t-1}+B_{t,j'})$

$$=\sum_{k\in T}p_k+\min_{i'\le t\le j'}(B[i',t-1]+B[t,j'])$$

אבל $(t-1)-i'\le\alpha$ וגם $j'-t\le\alpha$

ולכן $B[t,j']=cost(optT_R)$ ו- $B[i',t-1]=cost(optT_R)$
עפ"י הנחת האינדוקציה

כאשר T_R מחזיק את המפתחות $t:j'$, ו- T_L את $i':(t-1)$.

$$B[i',j']=\sum_{k\in T}p_k+\min_{i'\le t\le j'}(cost(optT_L)+cost(optT_R))=cost(optT)$$

עפ"י ההוכחה של החלק הקודם.
□

ולכן $B[1,n]$ הוא המחיר האופטימלי של עץ המכיל את המפתחות 1:n.

אלגוריתם לבניית עץ כזה:

1. נחשב את B בצורה איטרטיבית מהאלכסון הראשון (הראשי) לאלכסון השני וכן הלאה, כאשר כל פעם צריך להשוות בין סכום איברים שכבר חישבנו באיטרציה הקודמת.
2. בזמן חישוב $B[i,j]$ נשמור גם את ה-t עבורו קיבלנו מינימום
3. מ- $B[1,n]$ והלאה נשחזר את הצמתים המינימליים לפי ערכי ה-t עד שנגיע לכל האיברים על האלכסון שמעל הראשי. ($B[i,i]$ – באותו i [illegible]).

סיבוכיות:

1. חישוב B איבר איבר דורש לקיחת מינימום של B לכל היותר n מספרים – $O(n)$
2. יש $O(n^2)$ איברים ב-B, סה"כ $O(n^3)$.

2. ב.

$$1\le i\le i'\le j\le j'\le n$$

$$B[i,j]+B[i',j']\le B[i',j]+B[i,j']$$

10/10

נוכיח באינדוקציה שאי השוויון מתקיים על $x=j'-i$:

בסיס: $x\le 1 \Rightarrow i=i'$ או $j=j'$ (ואז אי שוויון מתקיים באופן טריוויאלי).

הנחה: נניח נכונות עבור $x\le k$

צעד: $x=k+1$:

ישנן מספר אפשרויות:

א. $i<i'=j<j' \Rightarrow$

נניח $\underset{i<t\le j'}{\arg\min}\,(B[i,t-1]+B[t,j'])=t_1$

אזי יש שני מקרים:

i) $t_1\le j$:

$B[i,j]\le w(i,j)+B[i,t_1-1]+B[t_1,j]$ לפי הגדרה.

$$B[i,j]+B[i',j']=B[i,j']+B[j,j']\le w(i,j)+B[i,t_1-1]+B[t_1,j]+B[j,j']$$

$$\underset{\substack{\Updownarrow\\ w(i,j)\le w(i,j')}}{\le} w(i,j')+B[i,t_1-1]+B[t_1,j]+B[j,j']\underset{\text{הנחת האינדוקציה}}{\le} w(i,j')+B[i,t_1-1]+B[t_1,j']+B[j,j]$$

$$=B[i,j']+B[i',j].$$

ii) $t_1>j$: המקרה סימטרי ל-i.א.

ב. $i<i'<j<j'$

נניח

$\underset{i<t\le j'}{\arg\min}\,(B[i,t-1]+B[t,j'])=t_1$

$\underset{i'<t\le j}{\arg\min}\,(B[i',t-1]+B[t,j])=t_2$

יש לנו שוב שני מקרים סימטריים:

i) $t_1\le t_2$

$B[i',j']\le w(i',j')+B[i',t_2-1]+B[t_2,j']$ לפי הגדרה.

$B[i,j]\le w(i,j)+B[i,t_1-1]+B[t_1,j]$ לפי הגדרה.

$$B[i,j] + B[i',j'] \leq w(i,j) + w(i',j') + B[i,t_1-1] + B[t_1,j] + B[i',t_2-1] + B[t_2,j']$$

$$\leq w(i',j) + w(i,j') + B[i,t_1-1] + \underset{1}{B[t_1,j]} + B[i',t_2-1] + \underset{2}{B[t_2,j']} \leq \dots$$

$w(i',j) + w(i,j') = w(i,j) + w(i',j')$

$$\leq w(i',j) + w(i,j') + B[i,t_1-1] + \underset{3}{B[t_1,j']} + B[i',t_2-1] + \underset{4}{B[t_2,j]} = B[i',j] + B[i,j']$$

$1+2 \leq 3+4$

לפי הנחת האינדוקציה

ii) $t_1 \geq t_2$ המקרה סימטרי ל ב.i

2 ב.

$\frac{10}{10}$

$i \le t \le i+d \le n \qquad B^d[i,t] = w(i,i+d) + B[i,t-1] + B[t,i+d]$

$1 \le i \le i' \le j \le j' \le i+d \le n$

$B^d[i,j] + B^d[i',j'] \le B^d[i,j'] + B^d[i',j]$

$$\underbrace{B[i,j-1]}_{1} + \underbrace{B[j,i+d]}_{5} + \underbrace{B[i',j'-1]}_{2} + \underbrace{B[j',i+d]}_{6} \le \underbrace{B[i,j'-1]}_{4} + \underbrace{B[j',i+d]}_{7} + \underbrace{B[i',j-1]}_{3} + \underbrace{B[j,i'+d]}_{8}$$

$1+2 \le 3+4$ מתכונות Monge ב-w.

$5+6 \le 7+8$ מתכונות Monge ב-B.

כרגע נרצה להגדיר מטריצה $\hat{B}^d$:

$$\hat{B}^d_{i,j} = \begin{cases} B^d[i,j] & \text{מוגדר } B^d[i,j] \\ \infty & \text{אחרת} \end{cases}$$

בהנחת אריתמטיקת אינסוף: $\infty+\infty=\infty$, $\infty \le \infty$

נוכיח $\hat{B}^d$ היא Monge: $\hat{B}^d[i,j] + \hat{B}^d[i',j'] \le \hat{B}^d[i,j'] + \hat{B}^d[i',j]$
$i \le i', j \le j'$

ישנן מספר אפשרויות:

1. 4 אינדקסים באיזור המוגדר $B^d[i,j]$, נובע מהוכחה קודמת.
2. 3 אינדקסים באיזור המוגדר: (ואחד לא).
   - i) $\hat{B}^d[i',j]=\infty$ באיזור הלא מוגדר ← מתקיים באופן טריוויאלי.
   - ii) $\hat{B}^d[i,j']=\infty$ באיזור הלא מוגדר, מתקיים באופן טריוויאלי.
   - iii) $\hat{B}^d[i,j]=\infty$ באיזור הלא מוגדר, זה לא יכול לקרות!
     - א. $j<i \Rightarrow j<i' \Rightarrow \hat{B}^d[i',j]$ לא מוגדר סתירה למקרה 2.
     - ב. $j>i+d \Rightarrow j'>i+d \Rightarrow \hat{B}^d[i,j']$ לא מוגדר סתירה למקרה 2.
   - iv) $\hat{B}^d[i',j']=\infty$ באופן דומה ל iii)
3. 2 אינדקסים באיזור המוגדר:

בבחירה אלו שני אינדקסים משני צידי האי שוויון (בדומה ל 2. iii)

ואז באופן טריוויאלי $\infty \le \infty$.

4. 3 אינדקסים באיזור הלא מוגדר, אזי ובוודאי לפחות אחד בכל צד של אי השוויון ואז באופן טריוויאלי $\infty \le \infty$

אזי הוכחנו ש $\hat{B}^\lambda$ monge בכל התחום.

<u>אלגוריתם:</u>

1. נריץ SMAWK על $-\hat{B}^\lambda$
2. לכל איבר e_i שב SMAWK מחזיר, נחזיר $-e_i$.

<u>הוכחה:</u>

האלגוריתם מחזיר את המינימום בכל שורה של B^λ:

1. $\hat{B}^\lambda$ היא monge $\Leftrightarrow$ לכל רביעיה $B[i,j]+B[i',j'] \le B[i,j']+B[i',j]$ $i\le i', j\le j'$
$\Leftrightarrow$ לכל רביעיה $-B[i,j]-B[i',j'] \ge -B[i,j']-B[i',j]$ $i\le i', j\le j'$ $\Leftrightarrow$ $-\hat{B}^\lambda$ inverse monge.
2. מטריצה A היא inverse monge $\Leftarrow$ Totally monotone. הראינו בכיתה.
3. הוכחנו ש SMAWK על מטריצה Totally monotone מחזיר את האיברים המקסימליים בכל שורה.
4. הרצת SMAWK על $-\hat{B}^\lambda$ שהיא Totally monotone (לפי 2) מחזירה מקסימום בכל שורה $\hat{e}_i$ (לפי 3). $\Leftarrow$ $-e_i$ מינימום בכל שורה ב $\hat{B}^\lambda$.
5. נותר להראות ש e_i הוא מינימום ב B^λ ולא מינימום של תוספת:
כל האינסופים שהוספנו הוא מוגדר (גדול) מכל ∞ ב $\hat{B}^\lambda$ ולכן $-\infty$ ב $-\hat{B}^\lambda$
ולכן בוודאי לא יוחזרו כמקסימום ע"י SMAWK.

10/10

2. d.

אלגוריתם:

1. נחשב $w(1,j)$ לכל $1 \le j \le n$.
2. עבור $d=1:n-1$:
3. נריץ SMAWK על $-\hat{B}^{d}$, כאשר ה-SMAWK יבקש את האיבר $-\hat{B}^{d}[i,t]$:

אם לא בתחום אז $-\infty$

אם בתחום, נחשב:

i) $w(i,i+d) = w(1,i+d) - w(1,i)$

ii) $B[i,t-1] = w(i,t-1) + \min\limits_{i<k\le t-1} (B[i,k-1] + B[k,t-1]) =$

$= \min\limits_{i<k\le t-1} \hat{B}^{t-1-i}[i,k] = \max\limits_{i<k\le t-1} (-\hat{B}^{t-1-i}[i,k])$

iii) $B[t,i+d] = w(t,i+d) + \min\limits_{t<k\le i+d} (B[t,k-1] + B[k,i+d]) =$

$= \min\limits_{t<k\le i+d} \hat{B}^{i+d-t}[t,k] = \max\limits_{i<k\le t-1} (-\hat{B}^{i+d-t}[t,k])$

ואז:

$\hat{B}^{d}[i,t] = \underset{i}{w(i,i+d)} + \underset{ii}{B[i,t-1]} + \underset{iii}{B[t,i+d]}$

4. הצד יבנה כמו בסעיף ה-1 נעקוב אחרי איזה צמודה ממצאת את $\hat{B}^{n-1}[1,*]$ (נניח t) אז נחזור רקורסיבית על $\hat{B}^{t-1-i}[i,*]$ ו $\hat{B}^{n-t}[t,*]$

נכונות:

1. הראינו כבר ב c.2 שבאלגוריתם אנחנו מחשבים את המינימום בשורות המטריצה $\hat{B}^{d}$.
2. המינימום בשורה הראשונה ב $\hat{B}^{n-1}$:

$w(1,n) + \min\limits_{1\le t\le n} (B[1,t-1] + B[t,n]) = B[1,n]$

הראינו כבר בסעיפים הקודמים שבנוסחה הרקורסיבית לחישוב האיבר של הצד האופטימלי, כאשר ניתן לחזור את הצד לפי ערכי t שממצעים את הנוסחה בכל צעד רקורסיה.

סיבוכיות:

1. חישוב $w(1,j)=w(1,j-1)+p_j$ בצורה איטרטיבית − $O(n)$
2. חישוב $w(i,i+d)$ − $O(n)$
3. חישוב $B[i,t-1]=\max_k(-\hat{B}^{t-1-i}[i,k])$ − $O(n)$ חישבנו את

המקסימום של כל השורות של $\hat{B}^{\alpha}$ − $\alpha<d$

4. חישוב $B[t,i+d]=\max_k(-\hat{B}^{i+d-t}[t,k])$ − $O(n)$ אם כן.
5. חישוב SMAWK בצורה איטרטיבית, כאשר יש מטריצה של $-\hat{B}^{d}[i,j]$

סכום ב $O(n)$ רצי 2,3,4 − $O(n)$.

סה"כ: n איטרציות − $O(n^2)$.

3. א. <u>אלגוריתם:</u>

1. נבצע חיפוש טווח רגיל של 2D-RT, כאשר "נאסוף" את כל הצמתים הקנוניים שנתקל בהם במהלך החיפוש. $v_1,\dots,v_p$
2. נבנה ערימת מקסימום מהערך המקסימלי של העץ המקושר ל-v_i לפי y לכל הצמתים.
3. נמחק את המקסימום מהערימה ונכניסו ל-k-list ונוסיף את הצומת הבא הגדול ביותר מתוך העץ של האיבר אותו מחקנו. נבצע k פעמים.
4. נחזיר את ה-k-list

* מציאת המקסימום נעשית ב-$O(1)$ יש מצביע finger ימיני שלו
ומכן לצמוד מהאיבר ה-d_i לאיבר הבא נמצא חברה לכל k
d_i: אם יש לו בן שמאלי אז נרד לבן השמאלי ואחרת נעלה לאב
עד למצוא שיש לו בנים שמאליים ונרד במסלול חיפוש ימני ביותר משם.
(לא באמת צריך לשמור מצביע אב, מספיק לזכור את המסלול מה-spine עד ל-d_i כל פעם).

<u>נכונות:</u>

1. אוספנו את כל הצמתים בטווח $[x_1,x_2]$ ואת כל האיברים כטווח $[x_1,x_2]$
נובע מנכונות חיפוש טווח ב-Range-tree.
2. החזרנו רשימה בגודל k שמכילה את האיברים המקסימליים לפי y בטווח $[x_1,x_2]$:

⇐ בכל איטרציה החזרנו את האיבר המקסימלי מבין האיברים בערימה.

בתחילת כל איטרציה האיבר המקסימלי בטווח נמצא בערימה:
~~נניח~~ נניח בשלילה שלא, אזי יש איבר ש שגדול מכל בערימה בערימה. בפרט. גדול מהאיבר בערימה, שנמצא אותו בעץ הצמת, סתירה לכך שאנחנו לוקחים את האיבר המקסימלי בכל פעם.

__סיבוכיות:__

1. חיפוש צומת ומיסוף הצדדים - $O(\log n)$

2. בניית ערימת מקסימום היא לינארית במספר האיברים, יש $O(\log n)$ איברים (אחד מכל תת עץ) - $O(\log n)$

3. מחיקת מקסימום מהערימה - $O(\log\log n)$, k פעמים $O(k\log\log n)$

4. מציאת האיבר הבא בתת העץ של המקסימום הנמחק:

עבור עץ T_i נניח: k_i איברים: ~~אלו כל המקומות הבינוניים~~

נניח v הוא הצומת של ה spine שבאינו k_i (וכל הצמתים ממנו) בתת העץ. אזי, יש ב-v $O(k_i)$ צמתים בניהול ובמבנה של האיטרציות עברנו בכל צומת לכל היותר פעמיים. ובדיקת הצמתים 10

זה בהנחה ש-v הצורך הוא השמאלי הימני

ולכן $O(k_i)$. נסכום על כל העצים $\sum_i k_i = k$

סה"כ: $O(\log n + k + k\log\log n) = O(\log n + k\log\log n)$

10/10

3.ט. נשים לב למספר אבחנות:

אבחנה 1: ניתן לבצע $O(h)$ בכניסות לעץ 2-4 בגובה h, על מנת ששורשו יתפצל לשני צמתים, שעץ כל אחד בגובה (h)

הוכחה:

נתחיל עם עץ שבו ה spine השמאלי שלו מורכב מצמתים בדרגה 3, כלומר לשורש יש 3 בנים, לשמאלי מביניהם 3 בנים גם כן וכך הלאה.

נניח שהמשך העץ שבאשר צומת מתפצל הוא מתפצל ל: ② ③

כך שאר הצמתים בעץ הם בינאריים (2 בנים), נסמן את העץ ב T^3.

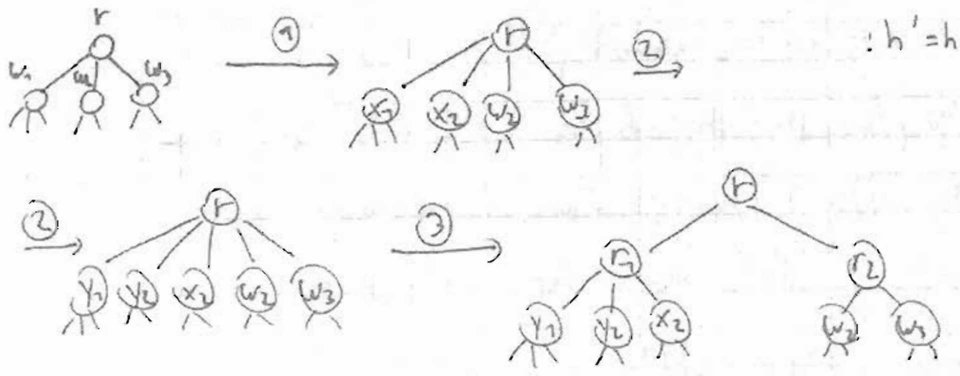
נראה שניתן לפצל את שורשו של עץ T^3 ע"י פיצול בנו השמאלי ביותר פעמיים ואת העץ של הבן השמאלי השורש החדש "ישאר" עץ T^3, והימני בינארי.

באינדוקציה מבנית על גובה העץ:

בסיס: $h=1$:

נניח נכונות עבור h

צעד: $h'=h+1$:



1. פיצול w_1 לפי הנחת האינדוקציה על x_1, x_2: T_{x_1} הוא T^3 וT_{x_2} בינארי

2. פיצול x_1 לפי הנחת האינדוקציה על y_1, y_2: T_{y_1} הוא T^3 וT_{y_2} בינארי.

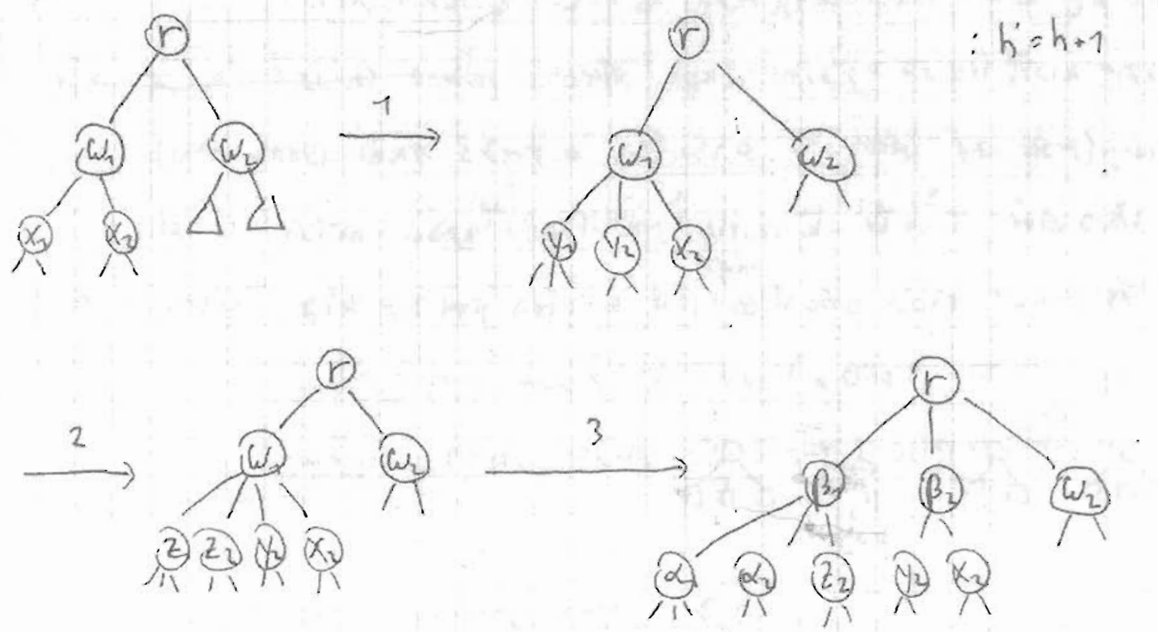
אז הוכחנו שניתן לפצל שורש של עץ T^3 לעץ בינארי ולעץ T^3 ע"י פיצול 2 בני T^3 בשמאלי ביותר, כלומר ניתן לפצל עץ T^3 ע"י $T^3(h)$ הכניסות: $T^3(h) = 2\cdot T^3(h-1) = 2^h$ כאשר $T^3(1)=2$

עכשיו נראה שניתן להגיע מעץ בינארי לעץ T^3 ע"י פיצול הבן השמאלי של שורש העץ, באינדוקציה:

בסיס: $h=2$

הנחה: נכון עבור $h' \le h$

צעד: $h'=h+1$:



1. נפצל את הבן השמאלי של w_1, x_1 ולפי הנחת האינדוקציה T_{w_1} יהיה T^3

2+3. נפצל את w_1 ל β_1, β_2 ע"י פיצול הבן השמאלי פעמיים: T_{β_2} בינארי ו T_{β_1} הוא עץ T^3 (לפי האינדוקציה הקודמת).

הוכחנו שניתן לעבור מעץ בינארי לעץ T^3 ע"י פיצול בנו השמאלי ולכן ניתן לפצל את שורשו ע"י $T(h)$ הכנסות:

$$T(h) = \underbrace{T(h-1)}_{\text{בפיצול ל }T^3} + \underbrace{2\cdot T^3(h)}_{\text{פיצול }T^3(h)} = 2^{h+1}$$

נחזור להוכחה הראשית של אבחנה 1, ברור שיש מנת לחסוך את מספר ההכנסות בעץ T^3 שהוא מ מנת לפצל שורשו מספיק לחסום את מספר ההכנסות של עץ בינארי, כי כל עץ אחר יקח פחות הכנסות.

עץ בגודל n הוא בגובה לכן היותר $h=\log_2 n$

ולכן מספר ההכנסות לפיצול שורש עץ הוא $2^{1+\log_2 n} = 2\cdot n$

התחלנו עם n איברים וסיימנו עם $3n$ איברים ולכן לא כל הפיצ בהכנסה לא יהיה יותר מ- $O(n)$ איברים. אם הוא יהיה $o(n)$ אז ב $O(n)$ הכנסות נוספות ניתן להגדילו ל $\Theta(n)$. □

אבחנה 2: עץ דרגה-4 בגובה h יכול להיות בגודל k עד גודל k^2, כאשר

$$k = 2^h$$

הוכחה: עץ אשר לכל צומת שני בנים יהיה בגודל $k = 2^h$

עץ אשר לכל צומת ארבעה בנים יהיה בגודל $4^h = 2^{2h} = (2^h)^2 = k^2$

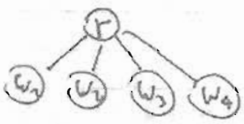
אם נוריד/נוסיף איבר אחד לעץ הראשון/שני יתקבלו מיזוגים/פיצולים

עד לשורש והגובה יקטן/יגדל בהתאמה.

[illegible] חסם כ-Ω

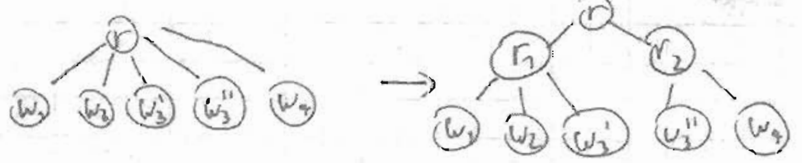
רצף של m פעולות ($\sqrt{n} < m$) יקח $O(m\sqrt{n})$ זמן:

הוכחה: נתחיל עם עץ אשר שורשו נראה כך:



$K_i = S(w_i)$ $1 \le i \le 4$

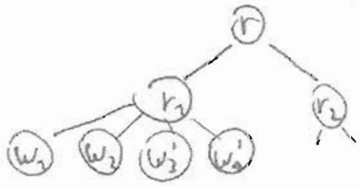
לפי אבחנה 1 נבצע $O(k_3)$ פעולות ונגיע למצב הבא:



כאשר $S(w_3'), S(w_3'') = O(k_3)$

עכשיו נוסיף $O(k_4)$ איברים הגדולים ~~מכל האיברים ב-w_3'~~

מהמפתח הימני של r_1 וקטנים מהמפתח השמאלי של r:



כאשר $S(w_4') = O(k_4)$

חזרנו למצב ההתחלתי כאשר

r_1 עכשיו בתפקיד r, יש לו

4 בנים בגדלים $O(k_1), O(k_2), O(k_3), O(k_4)$ וניתן לבצע את הפעולה

באותו מחזור.

סיבוכיות: הכנסה של O איבר לוקחת $O(\log n)$, לכן $O((k_3+k_4)\log n)$

אבל פיצול של r ל-r_1 ו-r_2 לוקח זמן של $O(n)$.

משום שצריך לבנות את העץ המשני ב-r והוא מאוזן לפי y.

עפ"י אבחנה 2 $S(k_3), (k_4)$ יכולים להיות בגודל $O(\sqrt{n})$ ולכן

שאחר $O(\sqrt{n})$ הכנסות ביצענו פיצול יקר נבצע

אי אפשר להכניס
באופן נתצנרי
נספר התיבריב נהיה
גדול מ-n
צריך למצוא לעצב
הפירוקים לצורת מחיקות

ולכן לאחר m הכנסות יהיו לנו $O(\frac{m}{\sqrt{n}})$ פיצולים כאלו

וכל פיצול הראינו שעולה $O(n)$, סה"כ $O(m \cdot \sqrt{n})$.

3. ג. נוכיח באינדוקציה על גובה העץ!

בסיס: $h=1$ $\quad 4=\frac{1}{2}\cdot 8^{1}\le S(v)\le 2\cdot 8^{1}=16$

מכיוון שכל בניו עלים זה יהיה מספר בניו הישירים.

נניח נכונות עבור h

צעד: $h'=h+1$ $\quad \frac{1}{2}\cdot 8^{h+1}\le S(v)\le 2\cdot 8^{h+1}$

מספר בניו הישירים של v יהיה מינימלי אם מספר צאצאיו יהיה
מינימלי אך מספר צאצאין של בניו יהיה מקסימלי:

$$\frac{\frac{1}{2}\cdot 8^{h+1}}{2\cdot 8^{h}}=\frac{8}{4}=2$$

מספר בניו הישירים של v יהיה מקסימלי אם מספר צאצאיו יהיה
מקסימלי אך מספר צאצאיו של בניו יהיה מינימלי:

$$\frac{2\cdot 8^{h+1}}{\frac{1}{2}\cdot 8^{h}}=\frac{4\cdot 8}{1}=32$$

$\square$

על השורש לא חל התנאי ש $S(v)\ge\frac{1}{2}\cdot 8^{h}$ ולכן האינדוקציה
לא תקינה עליו.

10/10

אין כאן אינדוקציה, לא השתמש בהנחה

9/10

insert(x):

1. נרד במסלול חיפוש אחר x, עד שנגיע לצומת u שכל בניה עלים
2. בדרך, בכל צומת v שנעבור דרכו נוסיף את x לעץ המשנה של צומת זו T_v.
3. נעלה חזרה באותו מסלול מהעלה החדש x עד לשורש ובכל צומת v נבדוק, אם $s(v) > 2\cdot 8^{h(v)}$:

ל-v יש $2 \le k \le 32$ בנים: $w_1,\dots,w_k$

נפצל את v לשני צמתים v_1, v_2 כך ש:

בניו של v_1 יהיו $w_1,\dots,w_{\lfloor \frac{k}{2} \rfloor}$

ובניו של v_2 יהיו $w_{\lfloor \frac{k}{2}\rfloor+1},\dots,w_k$

ומפתחות ינוהלו כמו בסעיף 1.a.

את עץ המשנה T_v נפצל לשני עצים T_{v_1}, T_{v_2} (נעבור על כל האיברים לפי סדר-y ונבדוק אם הם קטנים מ-f בציר ה-x יעברו ל-T_{v_1} (לפי סדר y), אחרת יעברו ל-T_{v_2} (לפי סדר y).

כאשר f הוא המפתח שהפריד בין $w_{\lfloor \frac{k}{2}\rfloor}, w_{\lfloor \frac{k}{2}\rfloor+1}$

4. אחרת נמשיך הלאה עד לשורש. (כמובן שאם השורש יתפצל, נוסיף צומת חדש ש-v_1,v_2 יהיו בניו והוא יהיה השורש החדש.

נכונות:

1. כל האיברים יהיו באותם הנכונים, מפני שהוספנו את x לפי מסלול חיפוש, הוספנו את x לכל עץ משנה.
2. העץ ישמור על המבנה:

כל צומת שלא עברנו דרכה לא השתנו מספר צאצאיה $(s(v))$.

כל צומת שעברנו דרכה:

i) $s(v) \le 2\cdot 8^{h(v)}$ - המבנה מתקיים

ii) $s(v) > 2\cdot 8^{h(v)}$ - פוצלה ל v_1,v_2 (ניתן לפצל כי יש לפחות 2 בנים כפי בסעיף הקודם).

$s(v_1), s(v_2) \le$ ~~[illegible]~~ $s(v) - s(w_i) \le 2\cdot 8^{h}+1-\frac{1}{2}\cdot 8^{h-1} < 2\cdot 8^{h}$

משום ש... היה תקין לפני insert, והוספנו רק איבר אחד.

3. פיצלנו את עצי המשנה T_{v_1}, T_{v_2} לפי המפתח f ולכן כל האיברים ב T_{v_1} נמצאים בטווח תת העץ של v_1 (וכנ"ל לגבי כל אלו).
ובאותן צורה גם עבור T_{v_2}.

אזי העץ הראשי שבנינו שמר על המבנה המוגדר (בנוי על חיפוש דו מימדי).
לנקודות $p=(x,y)$.

delete(x):

1. נרד במסלול חיפוש אחר x, עד שנגיע לצומת עם שני בנים עלים ובינהם x.
2. בדרך, בכל צומת v שנעבור דרכו נמחק את x מעץ המשנה של v.
3. נמחק את העלה x, ואת המפתח המתאים באחיו שהפריד בינו לבין אחיו:
אם הוא בן ימני ביותר את המפתח הימני ביותר, אם שמאלי ביותר בהתאם ואם אמצעי אז אחד המפתחות (או מימינו או משמאלו).
4. נעלה חזרה באותו מסלול לשורש ובכל צומת v, נבדוק אם $s(v) < \frac{1}{2}\cdot 8^{h(v)}$
ל-v יש לפחות אח אחד (למשל אליו). w.
i) $s(w) < [redacted] \cdot 8^h$:
נאחד את v ואת w לצומת יחיד x, כאשר הבנים של x יהיו הבנים של v, w (הסדר יישמר ביניהם). והמפתח אשר אותיהם שהפריד ביניהם ירד ל-x ויפריד בין שני הבנים הקיצוניים של v,w המצדים ב-x.
ואת עצי המשנה T_v, T_w נאחד לעץ משנה יחיד T_x (mergesort)
ii) $s(w) \geq 8^h$:
נעביר בן קיצוני של w (צמוד ל v) להיות בן של v.
המפתח שהפריד בין v,w ירד להפריד בין הבן החדש של v לקיצוניים והמפתח שהפריד בין הבן הישן של w לשאר יעלה לאב להפריד בין v,w. העץ T_w יבנה מחדש כמו ב split של insert.
והעץ T_v יבנה מחדש כמו בסעיף i) ב merge.
5. אחרת נמשיך הלאה עד לשורש. (במקרה שאם שני בניו היחידים של השורש מתאחדים אז מורידים את השורש ולוקחים את האחד הבא והוסף להיות השורש).

נכונות:

1. כל האיברים באותם הבנים, הורדנו את x ~~מהבנים~~ מה זז מצדי המשנה. וכל איחוד/פיצול השתמש במפתחות הקיימים שהפרידו בצורה נכונה לפני הפעולה.

2. כדי שימור זה מבנה:

כל צומת שלא עברנו דרכה לא השתנה מספר צאצאיה.

כל צומת שעברנו דרכה:

i) $s(v) \geq \frac{1}{2} \cdot 8^h$ - המבנה מתקיים.

ii) $s(v) < \frac{1}{2} \cdot 8^h$:

א. $s(w) < 8^h$ ~~[illegible]~~ $s(x) = s(w) + s(v) < \frac{3}{2} \cdot 8^h < 2 \cdot 8^h$

מתקיים.

ב. $s(w) \geq 8^h$:
$s(w') = s(w) - s(x) \geq 8^h - 2 \cdot 8^{h-1} \geq \frac{1}{2} \cdot 8^h$

$s(v') = s(v) + s(x) \leq \frac{1}{2} \cdot 8^h - 1 + \frac{1}{2} \cdot 8^{h-1} < 2 \cdot 8^h$

כאשר x הוא הבן המועבר של w ו-w', v' הצמתים לאחר הצעד

3. בשחזור v,w איחדנו בין תתי העצים של v,w אזי T_x נשאר בגובה של x וכל המפתחות בתת העץ שלו נמצא ב T_x.

כשפיצלנו, עשינו זאת כמו ב insert.

אבחנות:

1. פיצול/מיזוג של צומת v, לינארי בגודל תת העץ - $O(s(v))$, משום שכיוון שאנו ממיינים לפי סדר את כל האיברים ומיזוג אפשרי עם mergesort, (אבחנה 2).

2. צומת בעץ זה יכול להיות גדול לכל היותר פי 4 מאחיו ולכן המקרה הפתולוגי של 6.3 לא יכול לחזור על עצמו.

סיבוכיות: נחשב את סיבוכיות ה פעולות ע"י Amortized analysis:

$\Phi = \sum_{v,\, s(v)=2\cdot 8^{h(v)} \text{ or } s(v)=\frac{1}{2}\cdot 8^{h(v)}} 4 \cdot s(v)$, נשים לב ש $\Phi \geq 0$ וחסום ב $O(n\log n)$ בכל שלב:

נניח כל הצמתים נספרים, בכל רמה סכום גדלי הצמתים הוא n, סה"כ $4n\log n$.

$am(insert) = ac(insert) + \Delta\Phi = O(\log^2 n) +$ עלות הפיצולים $+ \Delta\Phi = \dots$

נניח היו k פיצולים בצמתים $v_1, \dots, v_k$ אזי עלות כל פיצול לפי אבחנה 1 $s(v_i)$

$\dots = O(\log^2 n) + s(v_1) + \dots + s(v_k) - 4 \cdot s(v_1) - \dots - 4 \cdot s(v_k) = O(\log^2 n)$

מאחר שכל צומת שהתפצלה יש נספרת יותר בכל פוטנציאל, היא נוספה לצומת אחרת.

$$am(delete) = ac(delete) + \Delta\phi = O(\log^2 n) + \text{עלות פיצול או מיזוג} + \Delta\phi$$

באותה צורה נניח היו k מיזוגים (פיצולים כבר ראינו) ב $v_1, \dots, v_k$

$$am(delete) = O(\log^2 n) + 4\cdot s(v_1) + \dots + 4\cdot s(v_k) - 4\cdot s(v_1) - 4\cdot s(v_k) = O(\log^2 n)$$

לפי שהראינו 1,2

ולכן סה״כ m פעולות של insert/delete:

$$ac(m\text{-insert/delete}) = am(m\text{-insert/delete}) + \phi_0 - \phi_n = O(m\log^2 n) + O(n\log n)$$

(הוכחנו $0 \le \phi_i \le O(n\log n)$)

ולכן אם $m \geq n$: $ac(m\text{-insert/delete}) = O(m\log^2 n)$

הרושם שאינן לוקח בחשבון מקרים בהם הפוטנציאל גדל

צריך להתחשב בו גם שלילית