תרגיל מס.7

עפיף חלומה 302323001

17 במאי 2009

1 שאלה 1

עבור כל עץ חיפוש בינארי יודעים כי הקודקוד השמאלי הוא יותר קטן מהקודקוד עצמו, והקודקוד הימני יותר גדול מהקודקוד עצמו. אבל כל תת עץ גם מקיים את התכונה הזו, אזי אם מחליפים מילת "מדפיס" באלגוריתם שקיבלנו אז זה ידפיס בסדר עולה.

זו היא כן הוכחה שלמה כי הראנו כי עבור כל קודקוד מדפיסים את כל הקודקודים הקטנים ממנו לפני שמדפישים אותו, ואחר כך מדפיסים כל הקודקודים הגדולים ממנו.

נוסחת הריקורסיה היא $T(n) = T(n+1) + T(n+1) + \Theta(1)$ כאשר n הוא גובה העץ.

$$
\begin{aligned}
T(n) &= 2T(n+1) + \left(2^1 - 1\right)c \\
&= 2\left(2T(n+2) + c\right) + c \\
&= 2^2T(n+2) + \left(2^2 - 1\right)c \\
&= 2^2\left(2T(n+3) + c\right) + \left(2^2 - 1\right)c \\
&= 2^3T(n+3) + \left(2^3 - 1\right)c \\
&= \vdots \\
&= \left(2^n - 1\right)c \\
&= \mathcal{O}\left(2^n\right)
\end{aligned}
$$

2 שאלה 2

3 שאלה 3

נשתמש בעץ חיפוש בינארי עם Order Statstics אשר על כל קודקוד שומרים מספר הקודקודים בתת עץ ימיני ומספר הקודקודים בתת אץ הימיני שלו.

את כל הפעולות ראינו כבר בכיתה חוץ מפעולת Count-Range. נשתמש בפונקצית Find-LVP מהתרגיל הקודם.

Count-Range(T,a,b)	לחןעיפטם 1

```
function Range-count(T,a,b) {
junction=find-LVP(T,a,b);
countme=0;
BiggerThanA=0;
SmallerThanB=0 if(value(junction)!=a && value(junction)!=b) countme=1;
try {
if(value(junction)!=a)
BiggerThanA=countBiggerThanA(left(T), a);
if(value(junction)!=b) SmallerThanA=countSmallerThanB(right(T), a);
}
catch {
return "One of the value does not exist in the tree";
}
return countme+BiggerThanA+SmallerThanA
}
function countBiggerThanA(T, a) {
if(T==null) throw exception("A does not exist in tree");
if(value(T)==a) return rightSubtreeCount(T);
if(value(T)<a) return countBiggerThanA(right(T), a);
if(value(T)>a)
return 1 + rightSubtreeCount(T) + countBiggerThanA(left(T), a);
}
function countSmallerThanB(T, a) {
if(T==null) throw exception("A does not exist in tree");
if(value(T)==a) return leftSubtreeCount(T);
if(value(T)>a) return countBiggerThanA(left(T), a);
if(value(T)<a) return 1 + leftSubtreeCount(T) + countSmallerThanA(left(T),
a);
}
```

פונרציה זו עולה $\Theta\left(h\right)$ כי כל פונקציה פנימית עולה $\Theta\left(h\right)$.

4 שאלה 4

4.1 א

Recursive-Fibonaci	לחןעיפטם 2

```
Recursive-Fibonaci(n) {
if(n==1 OR n==2) return 1;
return Recursive-Fibonaci(n-1)+Recursive-Fibonaci(n-2);
}
```

ניתוח זמן ריצה:

$$T\left(n\in\{3,4,5\ldots\}\right)=T\left(n-1\right)+T\left(n-2\right)+\Theta\left(1\right)$$

רוצים להוכיח כי $T(n) = \Omega\left(2^{n/2}\right)$
נבדוק עבור $n=3$:

$$\begin{aligned} T(3) &\overset{?}{>} t2^{3/2} \\ T(2)+T(1)+c &\overset{?}{>} t2^{3/2} \\ 3c &\overset{?}{>} 4t > t2^{3/2} \\ \frac{3c}{4} &\overset{?}{>} t \end{aligned}$$

מתקיים עבור קבוע $t < \frac{3c}{4}$
נניח כי ההנחה מתקיימת עבור n ומוכיחים עבור $n+1$:

$$\begin{aligned} T(n+1) &> T(n)+T(n-1)+c \\ T(n+1) &> T(n-1)+T(n-2)+T(n-2)+T(n-3)+c \\ &> t\cdot 2^{n/2}+t\cdot 2^{\frac{n-1}{2}}+c \\ &> t\cdot\left(2^{\frac{n}{2}}+2^{\frac{n-1}{2}}\right)+c \\ &> t\cdot\left(\sqrt{2}+1\right)\left(2^{\frac{n-1}{2}}\right)+c \\ &> t\cdot 2\left(2^{\frac{n-1}{2}}\right)+c \\ &> t\cdot 2^{\left(\frac{n+1}{2}\right)}+c \\ &> \Omega\left(2^{\frac{n+1}{2}}\right) \end{aligned}$$

משל.

4.2 ב

Iterative-Fibonacci	לחןעיפטם 3

```
Iterative-Fibonacci(n) {
n1=n2=1
for(i=2;i<n;i++) {
tmp=n1+n2;
n2=n1;
n2=tmp;
}
return n2;
```

ניתוח סיבוכיות ריצה:

$$
\begin{aligned}
T(n) &= \Theta(1) + (n-3)\,Loop(n) \\
&= \Theta(1) + (n-3)\,\Theta(1) \\
&= c + (n-3)\,d \\
&= c + dn - 3c \\
&= \Theta(n)
\end{aligned}
$$

4.3 ג

נוכיח באינדוקציה עבור n.
בדיקה עבור $n=1$

$$
\begin{aligned}
F_1 &\stackrel{?}{=} 1 \\
\frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^1 - \left(\frac{1-\sqrt{5}}{2}\right)^1\right) &\stackrel{?}{=} 1 \\
\frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}-1+\sqrt{5}}{2}\right) &\stackrel{?}{=} 1 \\
\frac{\sqrt{5}}{\sqrt{5}} &\stackrel{?}{=} 1 \\
1 &\stackrel{\checkmark}{=} 1
\end{aligned}
$$

נניח כי זה מתקיים עבור n ונוכיח עבור $n+1$

$$
\begin{aligned}
F_{n+1} &\stackrel{?}{=} F_n + F_{n-1} \\
&= F_n + F_{n-1} \\
&= \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right) + \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^{n-1} - \left(\frac{1-\sqrt{5}}{2}\right)^{n-1}\right) \\
&= \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n + \left(\frac{1+\sqrt{5}}{2}\right)^{n-1} - \left(\frac{1-\sqrt{5}}{2}\right)^{n-1}\right) \\
&= \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n + \left(\frac{1+\sqrt{5}}{2}\right)^{n-1} - \left(\frac{1-\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^{n-1}\right) \\
&= \frac{1}{\sqrt{5}}\left(\left(1+\left(\frac{1+\sqrt{5}}{2}\right)\right)\left(\frac{1+\sqrt{5}}{2}\right)^{n-1} - \left(1+\frac{1-\sqrt{5}}{2}\right)\left(\frac{1-\sqrt{5}}{2}\right)^{n-1}\right)
\end{aligned}
$$

רואים כי $1+\frac{1\pm\sqrt{5}}{2} = \left(\frac{1\pm\sqrt{5}}{2}\right)^2$ אזי

$$
\begin{aligned}
F_{n+1} &\stackrel{?}{=} \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^2\left(\frac{1+\sqrt{5}}{2}\right)^{n-1}-\left(\frac{1-\sqrt{5}}{2}\right)^2\left(\frac{1-\sqrt{5}}{2}\right)^{n-1}\right)\\
F_{n+1} &\stackrel{\checkmark}{=} \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^{n+1}-\left(\frac{1-\sqrt{5}}{2}\right)^{n+1}\right)
\end{aligned}
$$

5 שאלה 5

5.1 א

לחזעיפטם 4 Find-max-range

```
Find-max-distance(T) {
distances=recusiveFind-max-distance(root(T));
if(distances.length==1) return distances[0]; //this tree has only one leaf, return
distance from root to leaf
return distances[0]+distances[1];
}
recusiveFind-max-distance(t){
r=list();
if(t)=null {
return r;
} else if(right(t)==null && left(t)==null) {
r.add(0);
return r;
}else{
r1=recusiveFind-max-distance(left(t));
r2=recusiveFind-max-distance(right(t));
r=findTwoMaxValues(r1,r2); //obvious implementation. this is psudo code.
increaseAllValuesInListByOne(r);
return r;
}
}
```

עבור כל קודקוד הפונקציה מחזירה את המרחק של שתי העלים הרחוקים ביותר בתת העץ של הקודקוד. שתי הקודקודים הרחוקים ביותר אחד מהשני חייבים להיות עלים(נניח בשלילה שהם לא עלים אז קיים קודקוד שהוא בן של אחד מהם שהוא יותר רחוק).

אזי סכום המרחק בין שתי העלים הרחוקים ביותר הוא התשוב הנכונה.

הערה: לא ביקשתם שהמסלול לא יחזור על אותו קודקוד/צלע פעמיים, אז זה נכון(ואין לי כח לתקן אותו בשביל משהוא לא מבוקש)

5.2 ב

האלגוריתם שהצגדתי הוא האכי יעיל, הוא עובד ב$\Theta\left(n\right)$

6 שאלה 6

6.1 א

פרט לצלעות בין $\min T$ ו $\mathrm{root} T$ עוברים על כל צלע פעמיים.
הפעולה successor אומרת לרדת שמאלה או לעלות עד שמוצאים מסלול לרדת ימינה. כלומר יורדים פעם אחת ואז עולים. אם כבר עלינו לא נחזור על מסלול זה עוד פעם.

6.2 ב

מקבלים חסם עליון $\mathcal{O}(n)$ ויותר הדק מקבלים $T(n) \leq 2(n-1)$ יותר מקבלים תשובה יותר הדקה אם מחסרים את הקודקודים בין $\min T$ ו $\mathrm{root} T$: $T(n) \leq 2(n-1) - \log_2 n$