תרגול 1, 9.3.2006

מתרגל : אריאל פרוקצה

אנליזה אסימפטוטית

יהיו $f,g:N\to R^+$.

הגדרה: $f(n)=O(g(n))$ אם קיים $n_0\in N$ ו $c>0$ כך שלכל $n\ge n_0$, מתקיים $f(n)\le c\cdot g(n)$.

דוגמא:

$$10^{80}=O(1)$$
$$n=O(2n)$$
$$2n=O(n)$$

דוגמא הפכית:

$n^2\neq O(n)$. אכן, יהי $c>0$, לכל $n>c$ מתקיים $n^2>cn$, ועל כן באמת $n^2\neq O(n)$.

הגדרה: נאמר $f(n)=\Omega(g(n))$ אם קיימים $c>0$ ו $n_0\in N$ כך שלכל $n\ge n_0$ מתקיים $f(n)\ge c\cdot g(n)$.

דוגמאות:

$$2n=\Omega(n)$$
$$n=\Omega(2n)$$
$$n\neq\Omega(n^2)$$

הגדרה: נאמר $g(n)=\Theta(g(n))$ אם קיים $n_0\in N$ ו $c_1,c_2>0$ כך שלכל $n\ge n_0$ מתקיים $c_1\cdot g(n)\le f(n)\le c_2\cdot g(n)$.

משפט: $f(n)=\Theta(g(n))$ אמ"מ $f(n)=O(g(n))$ וגם $f(n)=\Omega(g(n))$.

הוכחה:

נראה כי $f(n)=\Theta(g(n))\Leftarrow f(n)=O(g(n))$ וגם $f(n)=\Omega(g(n))$. קיים $n_0\in N$ ו $c_1,c_2>0$ כך ש לכל $n\ge n_0$ מתקיים $c_1\cdot g(n)\le f(n)\le c_2\cdot g(n)$.

בפרט, $\forall n>n_0, f(n)\le c_2\cdot g(n)\Rightarrow f(n)=O(g(n))$ ו $\forall n>n_0, c_2g(n)\le f(n)\Rightarrow f(n)=\Omega(g(n))$. מ.ש.ל.

בכיוון השני:

נניח ש $f(n)=O(g(n))$, ועל כן

$$\exists n_0,c_1>0,\forall n_1\ge n_0, f(n)\le c_1g(n)$$
$$f(n)=\Omega(g(n))\Rightarrow\exists n_2,c_2>0,\forall n\ge n_2, f(n)\ge c_2g(n)$$

נגדיר $n_0=\max\{n_1,n_2\}$

$$\forall n\ge n_2, c_2g(n)\le f(n)\le c_1g(n)\Rightarrow f(n)=\Phi(g(n))$$

עוד הוכחות :

טענה: $n^{\frac{1}{\log n}}=\Phi(1)$.

הוכחה: יהי n כלשהו. קיים m כך ש $n=2^m$. נקבל

$$n^{\frac{1}{\log n}}=\left(2^m\right)^{\frac{1}{\log 2^m}}=\left(2^m\right)^{\frac{1}{m}}=\sqrt[m]{2^m}=2=\Phi(1)$$

טענה: $\log n!=\Phi(n\log n)$

הוכחה:

$$\log n!\leq_{n!<n^n}\log n^n=_{\log a^n=b\log a}n\log n\Rightarrow\log n!=O(n\log n)_{n_0=1,c=1}$$

בה"כ n זוגי

$$n!\geq\frac{n}{2}\cdot\left(\frac{n}{2}+1\right)\cdot....\cdot n\geq\left(\frac{n}{2}\right)^{\frac{n}{2}}$$

$$\log n!\geq\log\left(\frac{n}{2}\right)^{\frac{n}{2}}=\frac{n}{2}\log\frac{n}{2}=_{\log\frac{a}{b}=\log a-\log b}\frac{n}{2}\log(n-\log 2)=\frac{n}{4}\log n+\frac{n}{4}\log n-\frac{n}{2}=$$

$$\frac{n}{4}\log n+\frac{n}{4}\log(n-2)\geq_{n\geq 4}\frac{n}{4}\log n=\frac{1}{4}n\log n\Rightarrow$$

$$\log n!=\Omega(n\log n)_{n_0=4,c=\frac{1}{4}}$$

שילוב המסקנות מביא אותנו למסקנה כי $\log n!=\Phi(n\log n)$. מ.ש.ל

טענה: $2^{\sqrt{\log n}}=O\left(\sqrt{2}^{\log n}\right)$

הוכחה:

$$\log\left(2^{\sqrt{\log n}}\right)=\sqrt{\log n}\cdot\log 2=\sqrt{\log n}$$

$$\log\left(\sqrt{2}^{\log n}\right)=\log n\log\sqrt{2}=\frac{1}{2}\log n$$

$$\frac{1}{2}\log n>_{n>16}\sqrt{\log n}$$

לכל $n\geq 16$ מתקיים $\log\left(2^{\sqrt{\log n}}\right)\leq\log\sqrt{2}^{\log n}$. $\log$ מונוטונית עולה, ועל כן , לכל $n\geq 16$,

$2^{\sqrt{\log n}}\leq\sqrt{2}^{\log n}$, ולכן $2^{\sqrt{\log n}}=O\left(\sqrt{2}^{\log n}\right)$ ($c=1,n_0=16$). מ.ש.ל.

תרגול 2 , 23.3.2006

רקורסיות

משפט (האב): יהי $a \geq 1$, $b > 1$. $f : N \to R^+$. וכן T מוגדרת ע"י

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

נחלק למקרים:

1. אם קיים $\varepsilon > 0$ כך ש $f(n) = O\left(n^{\log_b a - \varepsilon}\right)$, ואז $T(n) = \Theta\left(n^{\log_b a}\right)$
2. אם $f(n) = \Theta\left(n^{\log_b a}\right)$ אז $T(n) = \Theta\left(n^{\log_b a} \cdot \log n\right)$
3. אם קיים $\varepsilon > 0$ כך ש $f(n) = \Omega\left(n^{\log_b a + \varepsilon}\right)$ וגם קיים $c < 1$ ו n_0 כך שלכל $n \geq n_0$

מתקיים $af\left(\frac{n}{b}\right) \leq cf(n)$ אזי מתקיים $T(n) = \Theta\left(f(n)\right)$

נשים לב שלא מדובר בטריכוטומיה – לא כל נוסחת רקורסיה, אפילו אם היא בפורמט, מקיימות את כל התנאים. אבל כן מתקיים שקיום כל אחד מהתנאים גורר שהשני (והשלישי) לא יתקיים.

דוגמאות:

1. בואו ונפתור את נוסחת הרקורסיה $T(n) = 9T\left(\frac{n}{3}\right) + n$. נציב בנוסחה -

$a = 9, b = 3, f(n) = n$. נשים לב שמתקיים ב $n^{\log_b a} = n^2$ נשתמש במקרה 1. יהי $\varepsilon = 1$.

קיבלנו $n = O\left(n^{2-\varepsilon}\right) = O(n)$. מתקיים. נקבל כי פתרון הרקורסיה שווה $T(n) = \Theta\left(n^2\right)$.

2. $T(n) = T\left(\frac{2}{3}n\right) + 1$. במקרה זה $a = 1, b = \frac{2}{3}, f(n) = 1$. נבדוק מה זה $n^{\log_b a} = 1$. ניתן לראות כי $f(n) = \Theta\left(n^{\log_b a}\right)$ ולכן על פי המקרה השני של משפט האב $T(n) = \Theta\left(\log n\right)$.

3. $T(n) = 3T\left(\frac{n}{4}\right) + n\log n$. $a = 3, b = 4, f(n) = n\log n$. נרצה $n^{\log_b a} =\sim n^{0.79}$. להשתמש בתנאי 3. נבחר $\varepsilon = 0.2$, ואז $n\log n = \Omega\left(n^{0.79+\varepsilon}\right)$.

נותר לבדוק כי $af\left(\frac{n}{b}\right) = 3\frac{n}{4}\log\frac{n}{4} \leq \frac{3}{4}n\log n$, ועל כן מספיק לבחור $c = \frac{3}{4}$, ועל כן גם התנאי השני יתקיים. ממשפט האב (מקרה 3) נקבל כי $T(n) = \Theta\left(n\log n\right)$

4. $T(n) = 2T\left(n/2\right) + n\log n$. $a = 2, b = 2, f(n) = n\log n$. נרצה $n^{\log_b a} = n$. אף אחד מהמקרים לא מתקיים – (לדוגמא כי $n\log n \neq \Theta(n)$).

5. דוגמא (עץ רקורסיה): $T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + cn$ עבור $c > 0$. נרצה חסם מלמעלה.

נשים לב שבכל שורה בעץ יש cn. נסמן ב h את גובה העץ. נבדוק את הענף העמוק יותר. הגובה המקסימלי הוא מקיים $n\cdot\left(\frac{2}{3}\right)^h=1\Rightarrow n=\left(\frac{3}{2}\right)^h\Rightarrow h=\log_{3/2} n$. קיבלנו בתור חסם עליון (ניחוש) $\log_{3/2} n\cdot cn=O\left(n\log n\right)$.

<u>טענה:</u> תהי T מוגדרת ע"י $T\left(n\right)=T\left(\frac{n}{3}\right)+T\left(\frac{2n}{3}\right)+cn$ אזי $T\left(n\right)=O\left(n\log n\right)$.

<u>הוכחה:</u> באינדוקציה – ההנחה היא שלכל k, $T\left(k\right)\le d\cdot k\log k$. נשים לב, שלא צריך להוכיח את שלב הבסיס – כי אנו מדברים בצורה אסימפטוטית.

צ"ל ש $T\left(n\right)\le dn\log n$ (תחת הנחת האינדוקציה שהטענה מתקיימת לכל $k<n$).

$$T\left(n\right)=T\left(\frac{n}{3}\right)+T\left(\frac{2n}{3}\right)+cn\le_{Induction\ Hypotesis} d\frac{n}{3}\log\frac{n}{3}+d\frac{2n}{3}\log\frac{2n}{3}+cn$$

$$=_{\log\frac{a}{b}=\log a-\log b}\frac{dn}{3}\log n-\frac{dn}{3}\log 3+\frac{2dn}{3}\log n-\frac{2dn}{3}\log\frac{3}{2}+cn=$$

$$dn\log n-dn\left(\frac{1}{3}\log 3+\frac{2}{3}\log 3-\frac{2}{3}\log 2\right)+cn=$$

$$dn\log n-dn\left(\log 3-\frac{2}{3}\right)+cn\le_{-dn\left(\log 3-\frac{2}{3}+cn\right)\le 0\Rightarrow d\ge\frac{c}{\log 3-\frac{2}{3}}} dn\log n$$

30.3.2006, תרגול 3

אמור להיות נחמד.

מיון בזמן ליניארי

Counting Sort

קלט: מערך $A[1...n]$, כל $A[j] \in \{1,...,k\}$. איך שום הנחה על קשר בין n לk.

פלט: מערך B בעל n איברים, שהוא מכיל את A ממוין.

מערך עזר: מערך C עם k איברים.

פסאודו קוד:

Counting-Sort(A,B,n,k)

- for (i=1 to n) // < zero c
  - c[i] = 0
- for j=1 to n // < Count how many occurrences for each number 1…k
  - $C[A[j]]++$
- for i=2 to k //< After this loop, at c[i] will be the number of variables small or
  // < equal to i
  - $C[i] = C[i] + C[i-1]$
- for $j = n$ downto 1
  - $B[C[A[j]]] = A[j]$
  - $C[A[j]]--$

דוגמא:

$k = 5, n = 6$

$A = 2 \quad 3 \quad 1 \quad 1 \quad 5 \quad 3$

$C_{After\ second\ loop} = 2 \quad 1 \quad 2 \quad 0 \quad 1$

$C_{After\ Third\ loop} = 2 \quad 3 \quad 5 \quad 5 \quad 6$

$B_{After\ forth\ loop} = 1 \quad 1 \quad 2 \quad 3 \quad 3 \quad 5$

למה הולכים ברברס?

הגדרה: מיון יקרא **יציב** אם מספרים זהים מופיעים באותו הסדר בקלט ובסדר. (כלומר, אם יש שני זהים, אז הסדר ביניהם **ישמר**).

הערה: Counting sort, כפי שמומש כאן, הוא יציב.

ניתוח זמן ריצה:

לולאה 1, $\Theta(k)$, לולאה 2 $\Theta(n)$, לולאה 3 $\Theta(k)$, לולאה $\Theta(n)$.

סה"כ זמן ריצה סה"כ $\Theta(n+k)$. אם $k = O(n)$, אזי יש לנו זמן ריצה של $\Theta(n)$. אאוריקה.

Radix Sort

קלט: n מספרים , לכל אחד יש D ספרות. כל ספרה היא בטווח בין $1..k$.

פלט: ממוינים

Pseudo Code:

Radix-Sort

- for i=1 to d // I – means digit. Goes from right (lsd) to left (msd)
  - use stable sort to sort input according to digit i.

לדוגמא:

$$input = \begin{Bmatrix} 243 \\ 657 \\ 225 \\ 438 \\ 500 \end{Bmatrix} \rightarrow \begin{Bmatrix} 500 \\ 243 \\ 225 \\ 657 \\ 438 \end{Bmatrix} \rightarrow \begin{Bmatrix} 500 \\ 225 \\ 438 \\ 243 \\ 657 \end{Bmatrix} \rightarrow \begin{Bmatrix} 225 \\ 243 \\ 438 \\ 500 \\ 657 \end{Bmatrix}$$

<u>טענה:</u> Radix Sort עובד

<u>הוכחה:</u>

<u>הגדרה:</u> לכל מספר x בקלט, נסמן x_i את x מוגבל ל i הספרות הקטנות. לדוגמא - $x = 253$, $253_1 = 3, 253_2 = 53, 253_3 = 253$.

<u>טענת האינדוקציה:</u> אחרי i איטרציות, x_i ממוינים.

<u>בסיס האינדוקציה:</u> בשלב הראשון האלגוריתם ממיין ע"פ הספרה הכי פחות משמעותית, ולכן המספרים x_1 ממוינים.

<u>שלב האינדוקציה:</u> (אחרי איטרציה i). מניחים כי המספרים x_{i-1} ממוינים. נניח כי $x_i^1 < x_i^2$ (האינדקס למעלה). צ"ל כי x^1 מופיע לפני x^2 בפלט.

נפריד לשני מקרים: מקרה 1: הספרה ה i של x^1 קטנה מהספרה ה i של x^2. כיוון שבאיטרציה ה i אנו ממיינים ע"פ הספרה ה i, x^1 יופיע לפני x^2.

מקרה 2: הספרה ה i של x^1 שווה לספרה ה i של x^2. ועל כן, $x_{i-1}^1 < x_{i-1}^2$. ע"פ הנחת האינדוקציה, x^1 מופיע לפני x^2 באיטרציה ה $i-1$, ומכיוון שהמיון יציב, אזי כך יהיה גם אחרי האיטרציה i.

<u>זמן ריצה:</u> $\Theta\left(d\left(n+k\right)\right)$.

תרגול 4, 5.4.2006

הגדרה: עץ מושרש הוא מבנה המוגדר על קבוצה סופית של צמתים, אשר:

א. אינה מכילה צמתים כלל

ב. מכילה צומת מיוחד הנקרא שורש, ועבור $k \geq 0$ מכילה תתי קבוצות זרות של צמתים $T_1,...,T_k$, כאשר כל T_i עץ.

הגדרה: עץ סדור הוא עץ בו יש חשיבות לסדר הצמתים.

הגדרה: שני צמתים צמודים בעץ נקראים הורה וילד. (Parent, Child). צמתים בעלי אותו הורה נקראים אחים (siblings).

הגדרות: מסלול בעץ היא סדרה של צמתים $n_1,...,n_k$ כאשר כל n_i הוא הורה של n_{i+1}. אין דרישה ש n_1 יהיה השורש.

גובה העץ הוא אורך המסלול הארוך ביותר מהשורש אל עלה .

העומק של צומת הוא המרחק מהשורש אל הצומת.

הערה: אורך נמדד במספר הצלעות.

הגדרות: 1. עץ בינארי הוא **עץ סדור**, שבו לכל צומת יש לכל היותר 2 בנים.

2. **עץ בינארי מלא (full)** הוא עץ שבו לכל צומת יש 0 או 2 בנים (אין מצב של בן אחד)

3. **עץ בינארי שלם (complete)**הוא עץ בינארי **מלא** שבו כל העלים הם באותו העומק.

ורבי פרוקצה נתן סימנים – מלא, אני **Full**, שלם – אני לא **Full**, אני אם כבר **Complete**.

הערה:עץ שלם – מספר העלים בו הוא 2^h (כאשר h הוא העומק).

מספר הצמתים הפנימיים הינו $\sum_{i=0}^{h-1} 2^i = \frac{2^h - 1}{2-1} = 2^h - 1$.

סה"כ צמתים $2^{h+1} - 1$. (חיבור מספר העלים עם מספר הצמתים הפנימיים).

כלומר, בעץ שלם $h = \Theta(\log n)$.

טענה: עבור עץ **מלא**, עם גובה **h**, ומספר צמתים **n**, אזי $h = \Omega(\log(n)), h = O(n)$.

הוכחה: עבור עץ בגובה **h**, מספר הצמתים הגדול ביותר מתקבל אם העץ שלם. $n \leq 2^{h+1} - 1$, כלומר $h = \Omega \log(n)$.

עבור מקרה $h = O(n)$ - ניצור מסלול בגובה $\frac{n}{2}$, כאשר מצד ימין יהיה עלה, ומצד שמאל יהיה המשך העץ.

בעץ בגובה h עם הכי מעט צמתים יש $2h+1$ צמתים, כלומר $n \geq 2h+1$, ולכן $h = O(n)$.

איך לממש עץ שלם באמצעות מערך

ניקח מערך באורך $2^{h+1} - 1$ צמתים. את הצמתים בעומק d נסדר במקומות $2^d ... 2^{d+1} - 1$ (אנחנו מתחילים מאינדקס 1!)

לדוגמא:

A

B **C**

D **E** **F** **G**

איך ייוצג במערך:

ABCDEFG

(בעצם , כמו ייצוג **breadth**).

נשים לב, שבאופן כללי, אם i נמצא במקום $2^d + k$ אזי בניו יהיו במקומות $2^{d+1} + 2k, 2^{d+1} + 2k + 1$.

ומה הועילו חכמים בתקנתם?

Left-Child(i) = 2i [$2^{d+1} + 2k = 2\left(2^d + k\right)$ **]**

Right-Child(i) = 2i+1

Parent = $\left\lfloor i/2 \right\rfloor$

<u>תרגול 5 – 27.4.2006</u>

הודעות – שבוע הבא אין תרגול . אחרי זה פו'קצה מטייל לו בחו"ל.

<u>למה:</u> בערמה עם n צמתים יש לכל היותר $\left\lceil \frac{n}{2^{h+1}} \right\rceil$ בגובה h. (מה זה גובה? מרחק מקסימלי אל העלים).

<u>הערה:</u> בעץ שלם קל להוכיח.

<u>הוכחה:</u> באינדוקציה על h. נסמן בH את גובה הערמה .

<u>בסיס האינדוקציה:</u> צ"ל שיש בדיוק $\left\lceil \frac{h}{2} \right\rceil$ עלים. עלים הם כל הצמתים בעומק H, והצמתים בעומק $H-1$ שאינם אבות של בנים בעומק H.

נסמן בx את מספר הצמתים בעומק H . $n-x$ הוא מספר הצמתים בעץ שלם בגובה $H-1$, ולכן $n-x$ אי זוגי.

מכאן נובע כי אם x זוגי אזי n אי זוגי, ולהפך.

<u>נחלק לשני מקרים:</u>

- x זוגי, ועל בעומק $H-1$ יש 2^{H-1} צמתים, מתוכם בדיוק $\frac{x}{2}$ אבות של צמתים בגובה H,

  ועל כן ברמה $H-1$ יש $2^{H-1}-\frac{x}{2}$ צמתים שאינם אבות של צמתים בעומק H.

מספר העלים הוא :

$$x+2^{H-1}-\frac{x}{2}=2^{H-1}+\frac{x}{2}=\frac{2^H+x}{2}=_{x\ even}\left\lceil \frac{2^H+x-1}{2} \right\rceil=_{*}\left\lceil \frac{n}{2} \right\rceil$$

(*) - $\left(2^H-1\right)+x=n$ כי מספר הצמתים בעץ שלם בגובה $H-1$ הוא 2^H-1.

- x אי זוגי. מספר הצמתים בעומק $H-1$ שהם אבות של צמתים בעומק H הוא

$$\frac{x-1}{2}+1=\frac{x+1}{2}$$

מספר העלים הוא $x+2^{H-1}-\frac{x+1}{2}=2^{H-1}+\frac{x}{2}-\frac{1}{2}=\frac{2^H+x-1}{2}=\frac{n}{2}=_{*}\left\lceil \frac{n}{2} \right\rceil$

(*) x אי זוגי. , ועל כן n זוגי, ועל כן השיוויון מתקיים.

<u>שלב האינדוקציה:</u> נסמן ב n_h את מספר הצמתים בגובה h. צ"ל $n_h \le \left\lceil \frac{n}{2^{h+1}} \right\rceil$. נסמן ב T' כעץ בלי העלים (נגזמו). נסמן ב n' את מספר הצמתים ב T', ונסמן ב n'_{h-1} את הצמתים ב T' בגובה $h-1$. מתקיים: $n_h=n'_{h-1}$. כמו כן $n'=n-n_0=_{IH}n-\left\lceil \frac{n}{2} \right\rceil=\left\lfloor \frac{n}{2} \right\rfloor$, לכן

$$n_h=n'_{h-1}\le_{IH}\left\lceil \frac{n'}{2^h} \right\rceil=\left\lceil \frac{\left\lfloor \frac{n}{2} \right\rfloor}{2^h} \right\rceil\le\left\lceil \frac{\frac{n}{2}}{2^h} \right\rceil=\left\lceil \frac{n}{2^{h+1}} \right\rceil.$$

מיון – ערימה (Heap Sort)

HEAP-SORT(A)

- BUILD-HEAP(A)
- for i=length(A) downto 2
    - exchange $A[i] \leftrightarrow A[1]$
    - heap-size(A) = heap-size(A) – 1
    - HEAPIFY(A,1);

18.5.2006

שבוע הבא – לא צריך להגיע לתרגול תחליפי.
עוד שבועיים – להגיע לתרגול תחליפי.

טבלאות גיבוב (Hash Tables)

דוגמא: גודל הטבלא $m := 9$. נרצה להכניס את המפתחות הבאים: 15,3,7,17,21. פונקציית הגיבוב $h(k) = k \bmod 9$.

בשרשור:

0	1	2	3	4	5	6	7	8
			$3 \to 21$			15	$7 \to 16$	

מיעון פתוח:

ליניארית:

$$h'(k,i) = (h(k) + i) \bmod m$$

איך זה יראה:

0	1	2	3	4	5	6	7	8
			3	21		15	7	16

נוצרת תופעה שנקראת Primary clustering.

Double Hashing

$$h'(k,i) = (h(k) + ih_2(k)) \bmod m$$

$$h_2(k) = 2 - (k \bmod 2)$$

סדרת הבדיקה $h'(k,0),...,h'(k,m-1)$ היא תמורה על הקבוצה $\{0,...,m-1\}$ אמ"מ $h_2(k)$ זר ל-m. בנוסף,ל h_2 אסור להחזיר 0.

דוגמא:

0	1	2	3	4	5	6	7	8
16			3	21		15	7	

$$h'(16,1) = (7+2) \bmod 9 = 0$$

$$h'(21,1) = (3+1) \bmod 9 = 4$$

ניתוח זמן ריצה של שרשור:

הגדרה: עבור טבלה עם n מפתחות וגודל m נגדיר את **מקדם העומס** בתור $\alpha = \frac{n}{m}$.

נשתמש בהנחת '**גיבוב אחיד ופשוט**': לכל מפתח k יש סיכוי שווה להתמפות לכל כניסה בטבלא.

משפט: חיפוש לא מוצלח (שלא מוצא את האיבר) בטבלה לוקח בממוצע $\Theta(1+\alpha)$ (שימוש בשרשור, ועם הנחת הגיבוב אחיד ופשוט).

הוכחה: מההנחה נובע כי בכל תא יתמפו בממוצע $\alpha = \frac{n}{m}$ מפתחות, החיפוש דורש סריקה של הרשימה בתא $h(k)$, ולכן לוקח בממוצע α פעולות. יחד עם החישוב של h שלוקח $\Theta(1)$, נקבל $\Theta(1+\alpha)$.

<u>משפט:</u> (תחת אותן הנחות) – חיפוש מוצלח לוקח בממוצע $\Theta\left(1+\alpha\right)$. לפני הכנסת האיבר ה-i בטבלא, האורך הממוצע של רשימה היה $\frac{i-1}{m}$, לכן האיבר ה-i נכנס בממוצע למקום $1+\frac{i-1}{m}$. לכן הממוצע הוא:

$$\frac{1}{n}\sum_{i=1}^{n}\left(1+\frac{i-1}{m}\right)=\frac{1}{n}\sum_{i=1}^{n}1+\frac{1}{n}\sum_{i=1}^{n}\frac{i-1}{m}=1+\frac{1}{nm}\sum_{i=1}^{n}\left(i-1\right)=1+\frac{1}{nm}\cdot\frac{n\left(n-1\right)}{2}=$$

$$1+\frac{n}{2m}-\frac{1}{2m}=1+\frac{\alpha}{2}-\frac{\alpha}{2n}=\Theta\left(1+\alpha\right)$$

<u>פרשנות:</u>

בטבלת גיבוב, עם $n=O\left(m\right)$, אז $\alpha=\frac{n}{m}=\frac{O\left(m\right)}{m}=O\left(1\right)$, ואז זמן הריצה של חיפוש הוא קבוע, ולכן גם זמן הריצה של מחיקה הוא קבוע. ובכל מקרה – עולה $O\left(1\right)$.

31.5.2006

DFS ושימושיו

DFS(G=V,E)

1. for all $v \in V$
   a. color[v] = white
   b. $\pi[v] = NIL$
2. time=0
   a. for each $v \in V$
      i. if $color[v]$=white
         1. DFS-VISIT(v)

DFS-VISIT(v)

1. $color[v] = gray$
2. $d[v] = ++time$
3. for each $u \in Adj(v)$
   a. if $color[u] = white$
      i. $\pi[u] = v$
      ii. DFS-VISIT(u)
4. $color[v] = black$
5. $f[v] = ++time$

$Adj(v)$ - שכנים של v.

משמעות צבעים:

1. לבן – לא גילינו
2. אפור – התחלנו לחקור אבל עדיין לא יצאנו
3. שחור – סיימנו לגמרי עם הקודקוד וקשריו.

הגדרה: יער ה$-DFS$ של G הוא העצים שבהם כל קודקוד v הוא הבן של $\pi[v]$.

(משמעות – השורשים זה 'ההתחלות' של עצים שלא קשורים בגרף. אם הכל קשור, יש עץ אחד).

'משפט המסלול הלבן': בעץ הDFS של G , קודקוד u הוא אב קדמון של v אמ"מ בזמן $d[u]$ יש מסלול מu לv שכולל רק קודקודים לבנים.

מיון טופולוגי – מניחים גרף מכוון וא-ציקלי. מיון של קודקודים כך שאם $[u,v] \in E$ (E – קבוצת הצלעות) – אזי u מופיע לפני v במיון.

TOPOLOGICAL-SORT

1. run DFS to conpute times $f[v]$
2. as each vertex is finished, insert it into the front of a linked list
3. return the linked list

הגדרה:

- **קשת אחורית** צלע (u,v) כך שu הוא צעצע של v ביער הDFS (סוגרת מעגל). מדובר על צלע שעדיין לא ניצלנו (אין כוונה שכל צלע בעץ לא מכוון היא גם אחורית).

למה: כל גרף מכוון G הוא אציקלי, אם בשום יער DFS אין צלע אחורית.

<u>הוכחה:</u> נניח בשלילה שG אציקלי ויש צלע אחורית (u,v) - מהגדרה, v אב קדמון של u בעץ הDFS, כלומר יש מסלול מv לu, ולכן הצלע סוגרת מעגל, בסתירה להנחה.
בכיוון השני – נניח שאין צלעות אחוריות, אבל יש מעגל.

יהי v הקודקוד הראשון המתגלה ע"י DFS במעגל, ונסמן את הקודקוד שלפניו בu. בזמן $d[v]$ כל הקודקודים במעגל לבנים, ולכן יש מסלול לבן בין v לu. ע"פ משפט המסלול הלבן v אב קדמון בעץ הDFS, ולכן הצלע (u,v) אחורית בסתירה.

<u>משפט:</u> Topological sort עובד.
<u>הוכחה:</u> מספיק להראות שאם $(u,v) \in E$, אזי $f[v] < f[u]$.
כאשר האלגוריתם מסתכל על (u,v), לא יתכן כי v אפור, כי אז v הוא אב קדמון של u, כלומר (u,v) צלע אחורית, לפי הלמה יש מעגל, בסתירה להנחה של א-ציקליות של מיון טופולוגי.

אם v שחור, אזי $f(v)$ כבר נקבע, ו $f(u)$ עדיין לא – ולכן $f[v] < f[u]$. אם v לבן, אזי לא ביקרנו בו, DFS מגלה את v דרך u, ולכן יסיים עם v לפני u, כלומר - $f[v] < f[u]$.

8.6.2006, תרגול 10

עוד DFS (Depth First Search)

הערה: v צאצא של u בעץ הDFS אמ"מ בזמן $d[v]$ u היה אפור.

משפט: (הסוגריים) – לכל חיפוש DFS על G, ולכל שני קודקודים u,v בדיוק אחד מהמקרים הבאים מתקיים:

- הקטע $[d[u],f[u]]$ ו$[d[v],f[v]]$ זרים, ואף אחד מבין u וv אינו צאצא של השני ביער הDFS.
- הקטע $[d[v],f[v]]$ מוכל ב$[d[u],f[u]]$ - לכן v צאצא של u ביער הDFS.
- הקטע $[d[u],f[u]]$ מוכל ב$[d[v],f[v]]$ - לכן u צאצא של v ביער הDFS.

הוכחה: נניח ש$d[v]>d[u]$. נחלק לשני תתי מקרים:

- $d[v]<f[u]$ - כלומר, u היה אפור בזמן שv התגלה, ולכן – ע"פ הערה, v הוא צאצא של u ביער הDFS. בפרט, בחיפוש הגיע לv מu, ולכן u לא הסתיים לפני שv מסתיים, כלומר $f[v]<f[u]$.
- $d[v]>f[u]$ - בוודאות מתקיים $f[v]>d[u]$, לכן קיבלנו $d[u]<f[u]<d[v]<f[v]$ - כלומר הקטעים זרים. כמו כן, בזמן שהחיפוש גילה כל אחד מהקודקודים, השני **לא** היה אפור, ולכן אף אחד מהם לא צאצא של השני.

אם $d[u]<d[v]$ נחליף שמות.

מסקנה מיידית מהמשפט: ביער הDFS , v צאצא של u אמ"מ $d[u]<d[v]<f[v]<f[u]$ (הכלת קטעים).

משפט המסלול הלבן: בחיפוש DFS על G, v צאצא של u ביער הDFS שמתאים לחיפוש זה אמ"מ בזמן $d[u]$ היה מסלול בG בין u לv שכולו קודקודים לבנים.

הוכחה: $\Leftarrow$ נניח v צאצא של u, נוכיח שיש מסלול לבן. נתבונן בקודקודים על המסלול בין u לv בעץ הDFS. יהי w קודקוד במסלול, w גם הוא צאצא של u, ולכן ע"פ המסקנה $d[w]>d[u]$, ולכן w היה לבן בזמן $d[u]$.

$\Rightarrow$ מניחים שיש מסלול לבן בין u לv, ונניח בשלילה כי v אינו צאצא של u ביער הDFS., נתבונן במסלול בין u לv בגרף G. כיוון ש v לא צאצא של u קיים v' שהוא הראשון במסלול שלא צאצא. נקרא לקודקוד שלפני v' w. מתקיים $f[w]\le f[u]$, כי אם $w=u$ זה ברור, אחרת הוא צאצא (ע"פ המסקנה). ידוע $d[u]<d[v']$ וגם $f[w]\le f[u]$. נשאר להראות $d[v']<f[w]$ - וזה קורה שאו שv' התגלה לפני w, ואם לא – הוא יתגלה דרך w.

במשפט הסוגריים – המקרה היחיד שייתכן – הוא זה שהקטע $[d[v'],f[v']]$ מוכל בקטע של u, ולכן v' צאצא של u ביער הDFS, בסתירה לבחירת v'.

הערות למבחן:

המבחנים לפני שנתיים דומים. לפני שנה לא כל כך.

6/8 שאלות אמריקאיות.

השאר – 2/3 שאלות רגילות.

תרגול 11, 22.6.2006

דייקסטרה

מייצר את כל המסלולים הקצרים בגרף

הנחות: G גרף מכוון. כל המשקלים על הצלעות של G **אי שליליים**. פונקציית משקל $w: E \to R^+$

Extract-Min – מחלץ את הקודקוד v בעל הערך $d[v]$ מינימלי.

DIJKDSTRA(G,w,s)

1. for each $v \in V$
   a. $\pi[v] = nil$
   b. $d[v] = \infty$
2. $d[s] = 0$
3. $Q = V$
4. $S = \varnothing$
5. while ($Q \neq \varnothing$)
   a. $u = Extract - Min(Q)$
   b. $S = S \cup \{u\}$
   c. for all $v \in Adj[u]$
      i. if $d[v] > d[u] + w(u,v)$
         1. $d[v] = d[u] + w(u,v)$
         2. $\pi[v] = u$

דוגמא: באתר.

סימון: נסמן ב $\delta(s,u)$ את ערך המסלול הקצר ביותר בין s לu.

הערה: לכל קודקוד ובכל שלב $d[u] \geq \delta(s,u)$, אם בשלב כלשהו, $d[u] = \delta(s,u)$, אז זה ישאר כך עד סוף ריצת האלגוריתם.

משפט: dijkstra עובד – כלומר, כשהו מסיים לרוץ – בכל קודקוד v, מתקיים $d[v] = \delta(s,v)$.

הוכחה:

נטען שכשכל קודקוד v נכנס לקבוצה S, מתקיים $d[v] = \delta(s,v)$. זה מספיק ע"פ ההערה – כי ערך $d[v]$ לא ישתנה בהמשך.

נניח בשלילה שלא, אזי יש קודקוד u כך שכאשר u נכנס לS, $d[u] > \delta(s,u)$. בשאר ההוכחה עוסקים בזמן שבו u נכנס ל S.

$u \neq s$, כי $d[s] = 0$ מלכתחילה, וכן $\delta(s,s) = 0$, ולכן לא יתכן כי $d[s] > \delta(s,s)$, בסתירה לאיך שבחרנו את u.

כמו כן, חייב להיות מסלול בין s לu, כי אחרת $\delta(s,u) = \infty$, ולא יתכן כי $d[u] > \delta(s,u) = \infty$.

כיוון שיש מסלול בין s לu, יש מסלול קצר ביותר p בין s לu, כיוון ש p מתחיל בקבוצה S ונגמר ב $V-S$ אז יש על p קודקוד y שהוא הראשון ב $V-S$. נקרא לקודקוד שלפני y על p בשם x (כמובן ש $x \in S$).

נטען כי $d[y]=\delta(s,y)$ - ראשית, תת המסלול של p בין S לY הוא תת מסלול של מסלול קצר ביותר, ולכן בעצמו מסלול קצר ביותר, ולכן $\delta(s,y)=\delta(s,x)+w(x,y)$. בנוסף, $x \in S$, ולכן, מבחירת u, מתקיים $d[x]=\delta(s,x)$.

לכן $\delta(s,y)=\delta(s,x)+w(x,y)=d[x]+w(x,y)=_{*} d[y]$

(*) - כשx נכנס לS האלגוריתם עדכן את $d[y]$ כי y שכן של x.

לכן,(**) $d[y] \le d[u]$. קיבלנו כי $d[y]=\delta(s,y) \le \delta(s,u) \le d[u]$. מצד שני , מתקיים $d[y] \le d[u]$ כי גם $y,u \notin S$ ו u נבחר להכנס לS. לכן $d[u]=d[y]$, כלומר ב** הכל שיוויונים, ובפרט $\delta(s,u)=d[u]$, בסתירה לבחירת u. מ.ש.ל.

זמן ריצת האלגוריתם –

אפשר לממש את Q כמערך עם תא לכל קודקוד, ואז Extract-Min יעלה $O(V)$, אבל עדכון יעלה $O(1)$. עדכון d יעלה ב $O(|E|)$ פעמים , וביצוע $Extract-Min$ יעלה $O(|V|)$ פעמים, סה"כ זמן ריצה $O(|V|^2)$.