UNION-FIND

ADT זה מטפל בנתונים שהם קבוצות (סופיות) זרות זו לזו. מתחילים כרגיל ממבנה ריק ללא קבוצות כלל, פעולות העדכון הן:

① הוספת קבוצה חדשה בת איבר אחד לבד x (חדש כמובן) Make-Set(x).

② איחוד 2 קבוצות לקבוצה אחת

בכל קבוצה מסומן אחד האיברים כנציג של הקבוצה. הנציג של קבוצה בת איבר אחד {x} יהיה כמובן x.

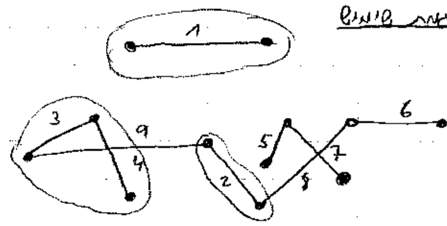
שאילתא ב- Union-find היא:

Find(x): מקבלת מצביע חיצוני לאחד האיברים באחת הקבוצות ויש להחזיר את <u>הנציג</u> של הקבוצה שבה הוא נמצא.

לצורך פעולת האיחוד ניתן להניח שמקבלים שני <u>נציגים</u> של קבוצות x,y (ואלו הפרמטרים); Union(x,y) מאחדת במבנה את 2 הקבוצות ובוחרים נציג לאיחוד (שיהיה x או y).

פעולות	
make-set(7)	{7}
make-set (13)	{13} , {7}
make-set (A)	{A} , {13} , {7}
Find (7)	התשובה לשאילתה 7
union (7,A)	{A , 7} , {13}
find(A)	התשובה לשאילתה 7.

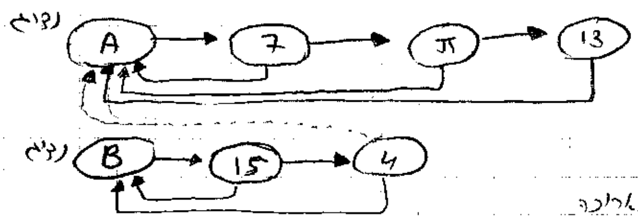
<u>דוגמת שימוש</u>



גרף מתאר רשת שרתים. כל נקודה היא שרת, קשתות מציינות חיבורים אפשריים. אם נקודות מוקפות בעיגול מייצגות נקודות עם קשרים ביניהם.

<u>מימוש I</u> (נאיבי) בעזרת רשימות מקושרות.

כל קבוצה תיוצג ע"י רשימה מקושרת כאשר הנציג בראש הרשימה ויש מצביעים מכל איברי הרשימה לנציג.



מימוש מלבני - עדיף הפוך (רשימה קצרה הקבוצה לנציג הרשימה ארוכה)

make-set	$\Theta(1)$
Find	$\Theta(1)$
Union	נתייחס למצביע עזר לסוף כל רשימה. בביצוע איחוד נחבר את זנב הרשימה הארוכה לראש הרשימה הקצרה ונשנה את "מצביע הנציג" של איברי הרשימה הקצרה לנציג הרשימה הארוכה. באופן זה איחוד של 2 רשימות מתבצע ב- $\Theta($אורך הרשימה הקצרה$)$

מבני נתונים - שיעור

ניתוח amortized

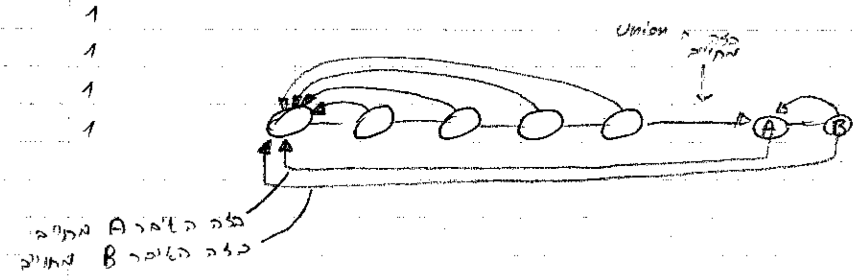
מה הסיבוכיות הכוללת של ביצוע רצף של L פעולות (כאשר n הוא מספר האיברים הכולל בקבוצות = כלומר n הוא מספר ה- make-set).

נגדיר מדיניות חיוב כזו

make-set, union, find יחויבו כולם ב- $\Theta(1)$. וכן כל איבר מ-n האיברים יחויב בפעולות של הכיווץ בו במהלך פעולת ה union, כלומר בכל פעולה שבה משתנה "מצביע הנציג" של האיבר.

(*) ניתן בשלב הזה להתייחס לחיוב האיבר כחיוב הפעולה make-set.

פעולות	חיובים	חיובים של איברים
⋮		
m-s	1	
m-s	1	
m-s	1	
find	1	
union	1	חיוב A, חיוב B
m-s		
union		
union		



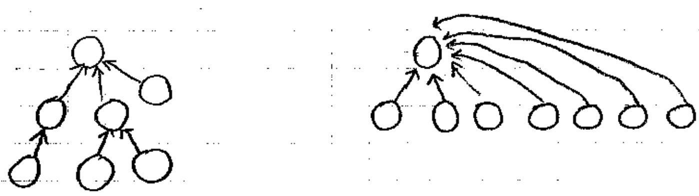
כמה פעמים לכל היותר יכול להיות מחויב איבר מסויים?

לכל היותר $\log n$ כי בכל פעם כי בכל פעם שאיבר מחויב (כי הקבוצה שלו משתנה) גדלה לפחות פי 2 (כי הקבוצה בה נמצא הנציג גדולה ממנה).

לסיכום, החיוב הכולל של L הפעולות יהיה $O(\underbrace{L}_{\text{חיוב פעולות}} + \underbrace{n\log n}_{\text{חיוב איברים}})$

מימוש II (Galler-Fisher)

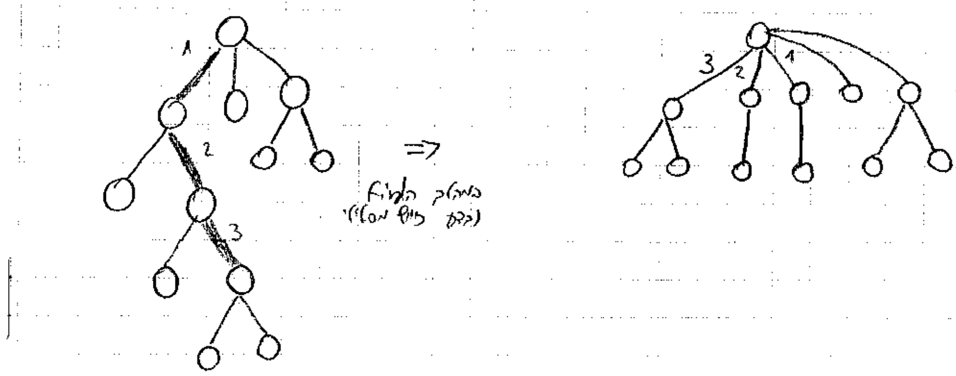
עץ (הפוך). כל קבוצה תיוצג ע"י עץ כאשר הנציג בשורש.



באיחוד קיימות 2 גישות טובות באותה מידה:

union-by-size: להפוך את שורש העץ הקטן יותר (מבחינת מספר קודקודים כולל) לבן של שורש העץ הגדול יותר.

union-by-rank: להפוך את שורש העץ הנמוך יותר לבן של שורש העץ "הגבוה" יותר.
למעשה נחליט מי הופך לבן של מי לפי פרמטר הקרוב לפרמטר הגובה שסימנו rank.



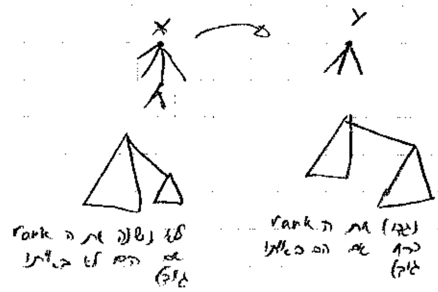
תוך כדי חיפוש, במעלה העץ, כל הנודים של איבר מסויים x, נהפוך את כל קודקודי הביניים במסלול מ-x לשורש העץ <u>לבנים ישירים</u> של שורש העץ.

Make-Set(X)

$parent(x) \leftarrow x$
$rank(x) \leftarrow 0$

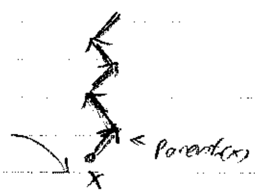
Union(x,y) ← בהנחה שx ו-y נציגים

```
if rank(x) < rank(y)
   then parent(x) ← y
   else parent(y) ← x
        if rank(x) = rank(y)
           then rank(x) ← rank(x)+1
```



Find(x)

```
if x ≠ parent(x)      אם x לא שורש
   then parent(x) ← find(parent(x))
return parent(x)
```



ניתוח Worst-case

Make-set	$\Theta(1)$
Union	$\Theta(1)$
find	$O(\log n)$

כי אפשר (תרגיל) להוכיח שמדיניות union-by-rank מבטיחה שכל עץ יהיה <u>בגובה</u> $\log n$ לכל היותר.

ניתוח amortized

Fischer הראה ב 1972 שרצף של l פעולות יקח $O(l \cdot \log\log n)$.
Hopcroft-Ullman הראו ב 1973 שזה בעצם $O(l \cdot \log^* n)$

$$\log^* n = \min\{k \mid \underbrace{\log\log\ldots\log n}_{k \text{ פעמים}} \le 1\}$$

$\log^* 1 = 0$, $\log^* 2 = 1$, $\log^* 4 = 2$, $\log^* 8 = 3$
$\log^* 16 = 3$, $\log^* 17 = 4$, ..., $\log^* 2^{16} = 4$, $\log^*(2^{16}+1) = 5$
..., $\log^* 2^{2^{16}} = 5$, $\log^* 2^{2^{16}}+1 = 6$.

Tarjan הראה ב-1975 שזה בעצם $O(l \cdot \alpha(n))$ כאשר $\alpha(n)$ היא פונקציית אקרמן ההפוכה.

14.7.08

מבני נתונים - שיעור

פונקצית אקרמן (הישנות):

$$A_k(n) = \underbrace{(A_{k-1}(A_{k-1}\dots(A_{k-1}(2))\dots))}_{n-1}$$

$$A_1(n) = n+2$$

$$A_2(n) = \underbrace{A_1(A_1\dots(A_1(2)))}_{n-1} = 2n$$

$$A_3(n) = \underbrace{A_2(A_2\dots(A_2(2)))}_{n-1} = 2^n$$

$\vdots$

$$A_4(n) = A_3(A_3\dots A_3(2)) = \left.2^{2^{\cdot^{\cdot^{2}}}}\right\}n \quad \text{Tower}$$

$\left(2,\ 4 = 2^2,\ 16 = 2^{2^2},\ 2^{16} = 2^{2^{2^2}}\right.$ המקומות שבהם $\log^*$ קופצת)

$A_k(n)$	$k \backslash n$	2	3	4	5	6	2^{16}
$n+2$	1	4	5	6	7	8 ...	
$2n$	2	4	6	8	10	12	
2^n	3	4	8	16	32	64	
Tower	4	4	16	2^{16}	$2^{2^{16}}$	$2^{2^{2^{16}}}$	
	5	4					