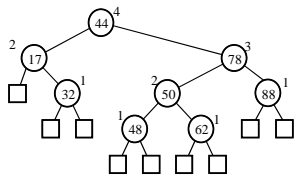


- עץ חיפוש בינארי מאוזן (מבחינת גובה)
- עבור כל צומת פנימית v של עץ T
- גובה תתי העצים מתחת ל- v יכול להיות שונה בעד 1



מבני נתונים

תרגול 3
ליאור שפירא



גובה של עץ AVL

$$n(h-1) > n(h-2) \Rightarrow n(h) \geq 2n(h-2)$$

$$n(h) \geq 2n(h-2)$$

$$n(h) \geq 4n(h-4)$$

...

$$n(h) \geq 2^i n(h-2i)$$

$$\Rightarrow n(h) \geq 2^{h/2-1}$$

פתרון הסדרה

$$\Rightarrow h \leq 2 \log n(h) + 2$$

לוג לשני הצדדים

$$\Rightarrow h = O(\log n)$$

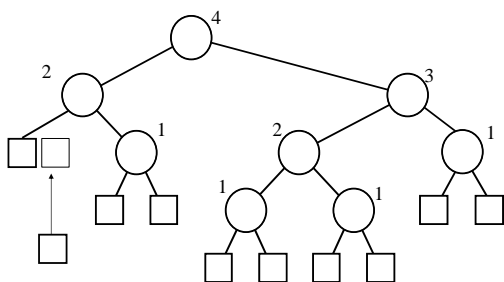
מש"ל

גובה של עץ AVL

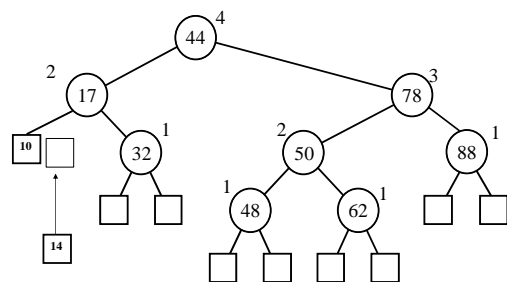
- למה: גובה של עץ T המחזיק n מפתחות הוא $O(\log n)$
- הוכחה:

- נמצא את $n(h)$, מס' מינימלי של צמתים פנימיים בעץ AVL בעל גובה h
- בבירור $n(1)=1, n(2)=2$
- עבור $h \geq 3$, עץ AVL בעל גובה h מכיל את השורש, תת עץ בגובה $h-1$ ותת עץ בגובה של לפחות $h-2$
- ז"א: $n(h) \geq 1 + n(h-1) + n(h-2)$

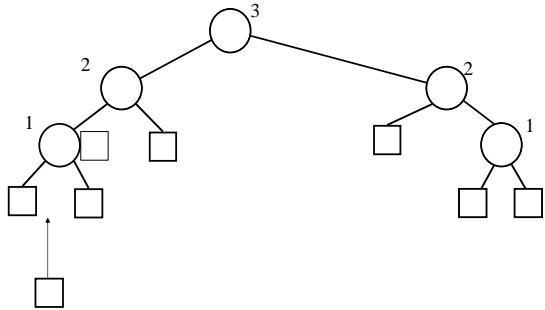
נתעלם מהערכים, נתמקד באיזון העץ



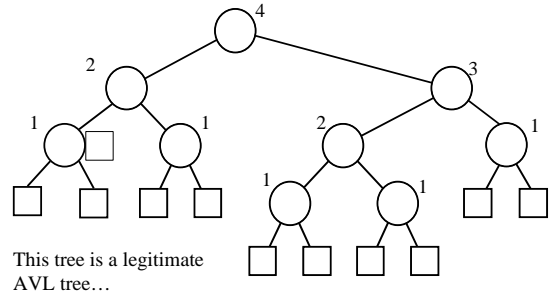
Insertion



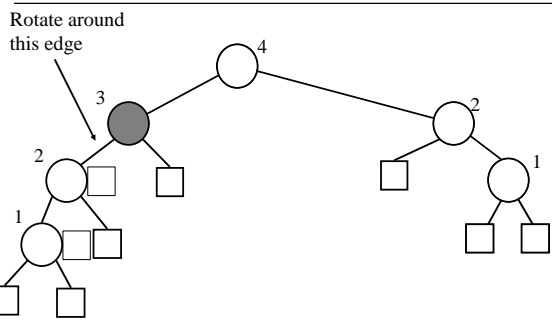
עוד דוגמה



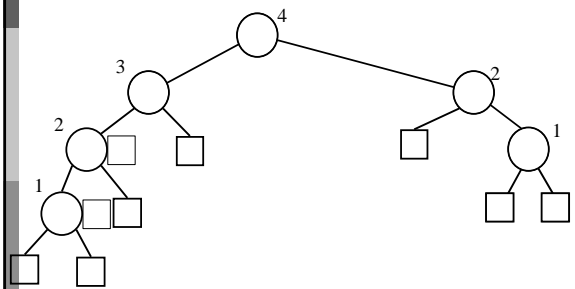
צומת פנימית חדשה



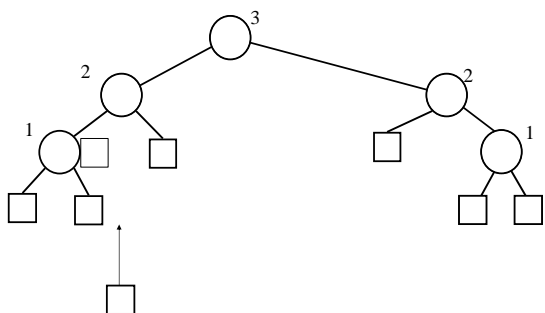
הפרנו את חוקיות העץ



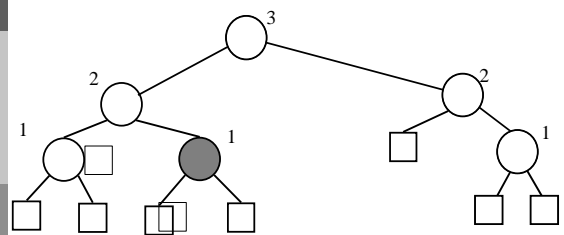
נוסיף את הצומת החדש



עוד דוגמה

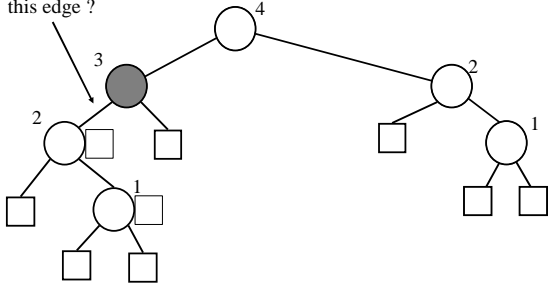


נתקן את השגיאות

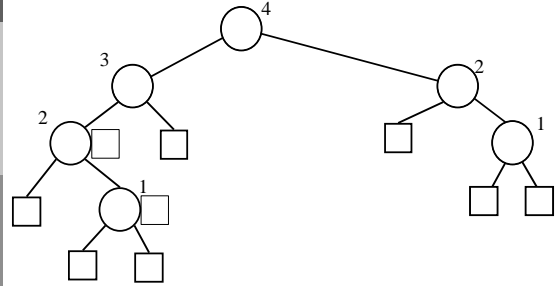


שוב קיבלנו הפרה של החוקים

Rotate around this edge ?

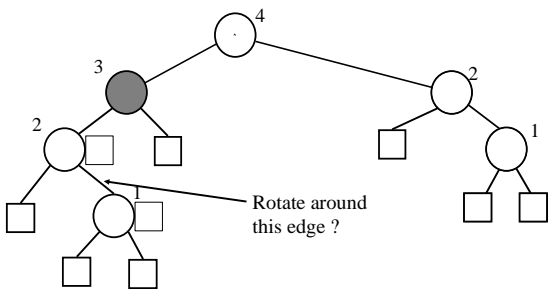


נוסיף את הצומת החדש

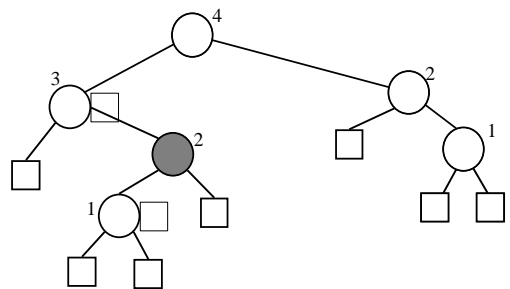


מה ניתן לעשות?

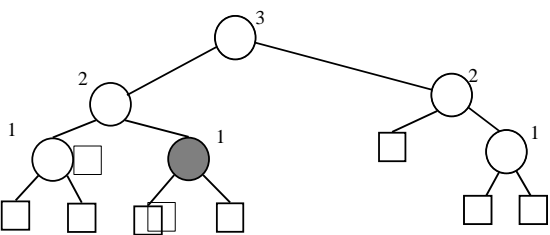
Rotate around this edge ?



הפעם הרוטציה לא עוזרת

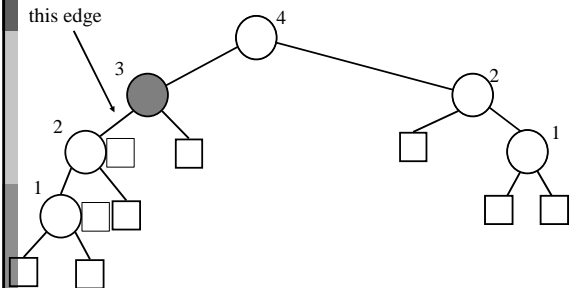


תיקנו את הפרת החוקים

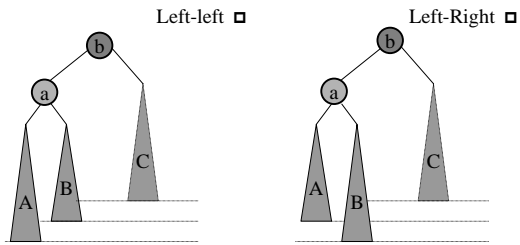


חזרנו למקרה הקודם

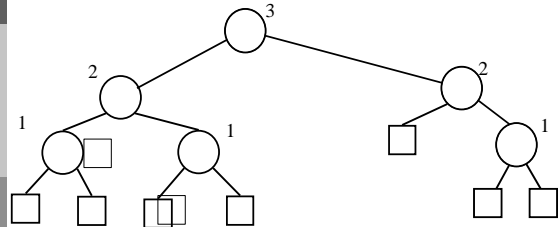
Rotate around this edge



סוגי חוסר שיווי משקל



אז מה האלגוריתם הכללי?

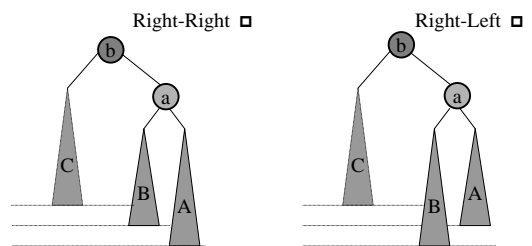


נסתכל על הבעיה לוקאלית

שני עקרונות מנחים

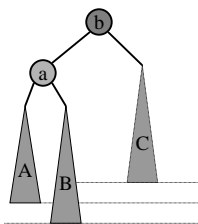
- Imbalance will only occur on the path from the inserted node to the root (only these nodes have had their subtrees altered - local problem)
- Rebalancing should occur at the *deepest unbalanced node* (local solution too)

ותמונות המראה שלהם

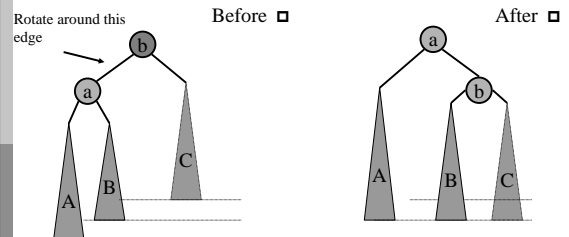


Left-Right fixing

• Before:

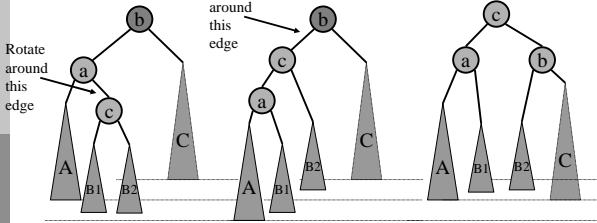


Left-left → single rotation



We need two rotations here...(double rotation)

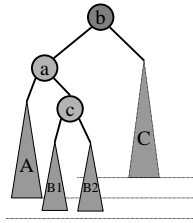
• Before:



• After:

Lets look at this more carefully

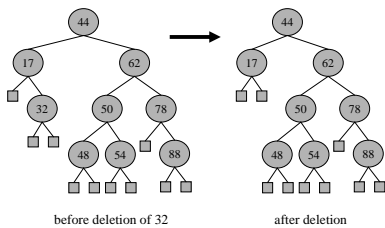
• Before:



מחיקת צומת

□ מחיקת צומת מתחילה כמו עץ בינארי רגיל, הצומת שמסירים הופך לצומת קצה ריק

□ ייתכן חוסר איזון



סיכום

- We fix the first node x on the way up where there is a violation
- After the fix, x is at the same height as it was before, so no nodes further up towards the root will need to be updated.
- Can implement using just 2 bits per node for rebalancing (the difference between the heights of the children)

זמני ריצה

- a single restructure is $O(1)$
 - using a linked-structure binary tree
- find is $O(\log n)$
 - height of tree is $O(\log n)$, no restructures needed
- insert is $O(\log n)$
 - initial find is $O(\log n)$
 - Restructuring up the tree, maintaining heights is $O(\log n)$
- remove is $O(\log n)$
 - initial find is $O(\log n)$
 - Restructuring up the tree, maintaining heights is $O(\log n)$

מחיקת צומת

□ בניגוד להכנסה, נצטרך לעשות restructure רקורסיבית על כל הצמתים הלא מאוזנים עד שנגיע לשורש העץ

