

## תזכורת – B-trees

- לכל צומת  $x$  יש השדות הבאים
  - $n[x]$  מס' מפתחות ב- $x$
  - המפתחות עצמם בסדר לא יורד
  - כל צומת פנימי מכיל גם  $n[x]-1$  מצביעים לילדי הצומת
  - המפתחות בצומת מפרידים את ערכי ילדי הצומת
- כל העלים בעץ הינם באותו עומק
- יש חסם עליון ותחתון למס' מפתחות שצומת מכיל
- נבטא חסם זה ע"י  $t \geq 2$ , הדרגה המינימלית של העץ
  - לכל צומת חוץ מהשרש יש לפחות  $t-1$  מפתחות
  - לפיכך, לכל צומת פנימית (חוץ מהשרש) יש לפחות  $t$  ילדים
  - בכל צומת יש לכל היותר  $2t-2$  מפתחות, ז"א מקסימום  $2t$  ילדים
  - צומת תיקרא מלאה אם מכילה בדיוק  $2t-1$  מפתחות

## מבני נתונים – עוד עצים



תרגול 10  
ליאור שפירא  
תשס"ח

## הכנסה ל-B-tree (אופציה ב')

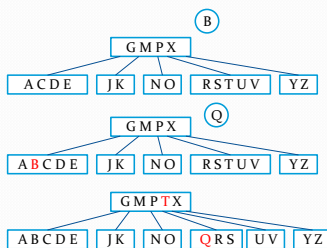
- B-Tree-Insert( $T, k$ )
  - $r \leftarrow \text{root}[T]$
  - If  $n[r] = 2t-1$ 
    - Create new root  $s$  and split  $r$  under it
    - B-Tree-Insert-Nonfull( $s, k$ )
  - Else
    - B-Tree-Insert-Nonfull( $r, k$ )

## הכנסה ל-B-tree

- אופציה א'
  - מצא את הצומת בו אמור להיכנס הערך
  - אם הצומת מלאה פצל אותה וחזור רקורסיבית
- אופציה ב'
  - תוך כדי חיפוש מקום להכניס את הערך החדש נפצל כל צומת מלאה שניתקל בה
  - יתרון: לא נצטרך לגשת פעמיים לצמתים (ניתכן שיושבות בויכרון יקר לגישה)
  - חסרון: דורש שמס' בנים מכסימלי יהיה  $2t$  (עצי 2-3 אינם B-Trees לפי הגדרה זו)

## דוגמה – עץ בו $t=3$

$t=3$  ז"א זהו עץ 3-6



## הכנסה ל-B-tree (אופציה ב')

- B-Tree-Insert-Nonfull( $x, k$ )
  - $i < n[x]$
  - If leaf $[x]$ 
    - Find correct place and insert value  $k$
  - Else
    - Find  $c_i[x]$  - child of  $x$  into which recursion continues
    - If  $n[c_i[x]] = 2t-1$ 
      - Split  $c_i[x]$
    - B-Tree-Insert-Nonfull( $c_i[x], k$ )

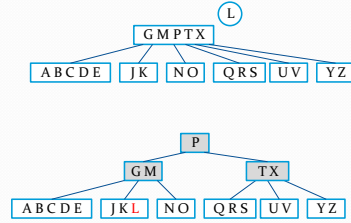
## עצי 2-4 (b-trees)

- תרגיל: אם נתחיל מעץ 2-4 ריק ונבצע  $m$  פעולות insert, כאשר בכל פעולה נקבל את מיקום ההכנסה, אזי העלות הכללית של הסדרה היא  $O(m)$  זמן אינטואיציה
- פעולות זולות – כשיש הרבה "חופשי" בעץ
- פעולות יקרות – כשאין "חופשי" בעץ
- מה מסמל "חופשי" בעץ? צמתים בדרגה 2-3



## דוגמה – עץ בו $t=3$

•  $t=3$  ז"א זהו עץ 3-6

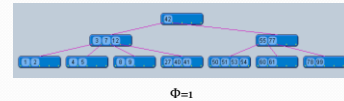


## הוכחה

- נצטרך להוכיח שני דברים
  1. פי הפוטנציאל 0 בהתחלה ואי-שלילית בכל שלב
  2. לכל פעולה  $op$  מתקיים:  $cost(op) + \Delta\Phi = O(1)$
- 1 ברור
- עבור 2 נוכיח כי כל פעולה עולה  $O(1) + k$  כאשר  $k$  מס' פעולות ה-split שמתבצעות.

## הוכחה

- נקרא לרמת ה"חופשי" בעץ – slack
- נגדיר פונקציית פוטנציאל על פי slack
- אם אין slack – הפוטנציאל גבוה
- אם אין slack בצומת – יש 4 ילדים
- פונקציית הפוטנציאל:  $\Phi(D) = \# \text{vertices of degree } 4$



## האם זה נכון לעצי 2-3?

- ניתן להראות כי בעצי 2-4 עבור כל סדרת פעולות insert ו-delete, עלות amortized נשארת  $O(1)$  לפעולה
- האם הדבר נכון לעצי 2-3?
- הדבר לא נכון, בעץ 2-3 מלא, אם נבצע סדרת פעולות insert ו-delete של אותו אבר, כל הפעולות יהיו יקרות ולכן  $O(\log n)$  לפעולה!

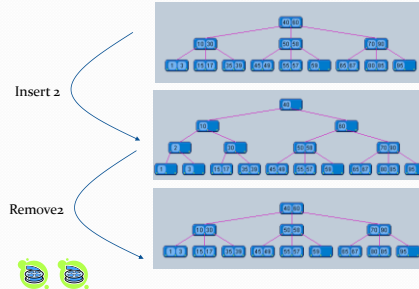
## הוכחה

- כל split מפצל צומת 4 לצומת דרגה 2 וצומת דרגה 3
- לכן, כל פעולת split (כמעט) מורידה את הפוטנציאל ב-1
- ה-split הראשון, על עלה, מפצל צומת 0 לשני צמתים מדרגה 0
- ה-split האחרון מפצל צומת  $v$  מדרגה 4 לשני צמתים (2 ו-3) אך אביו עלול להשתנות לדרגה 4 – אין שינוי פוטנציאל
- לכן:
  - $\Delta\Phi \leq -(k-2)$
  - $cost(op) + \Delta\Phi = O(1)$

## שאלה מתוך מועד א' תשס"ז

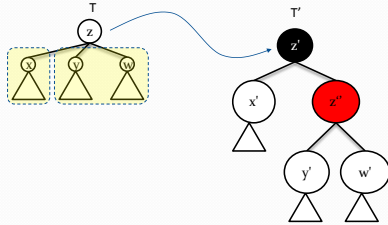
- האם יש שקילות בין עצי אדום שחור לעצי 2-4?
- האם ניתן להגדיר התאמה שבהינתן עץ 2-4 T הופך אותו לעץ אדום שחור T'?
- האם ניתן להגדיר התאמה הפוכה?

## האם זה נכון לעצי 2-3?



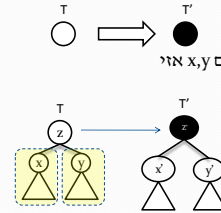
## שאלה מתוך מועד א' תשס"ז

- אם ל-z השורש של T יש שלוש בנים אזי



## שאלה מתוך מועד א' תשס"ז

- בהינתן T עץ 2-4+ נהפוך אותו לעץ אדום שחור T' כך ש:
- במידה ו-T מכיל עלה יחיד
- עבור T עם שורש z ולו שני בנים x, y אזי

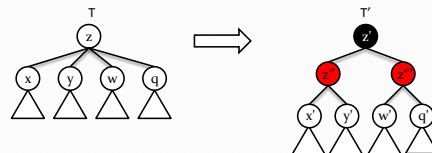


## שאלה מתוך מועד א' תשס"ז

- האם נקבל עץ אדום-שחור חוקי עבור כל קלט?
- האם נשמר החוק האדום?
- כן, צמתים אדומים נוצרים רק ב"שכבות ביניים", שורש של כל תת עץ שנתאים רקורסיבית יהיה שחור
- האם נשמר החוק השחור? האם כל מסלול יכול את אותו מס' צמתים שחורים?
- מההגדרה - העומק השחור של עלה ב-T שווה לעומק של העלה המתאים בעץ 2-4. מכיוון שעומק העלים ב-T שווה, אזי גם ב-T'.

## שאלה מתוך מועד א' תשס"ז

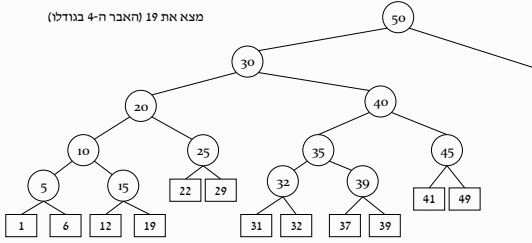
- אם ל-z השורש של T יש ארבעה בנים אזי



### תרגיל

- הראו כי בשינויים קלים למבנה עץ בינארי (מאוזן), ניתן למצוא את האבר ה-k בגודלו, בזמן  $O(\log k)$

מצא את 19 (האבר ה-4 בגודלו)



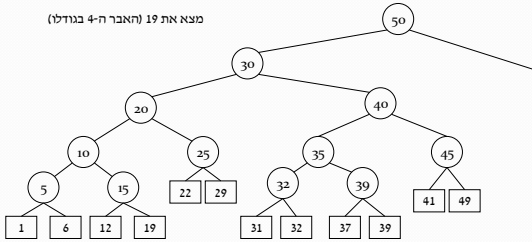
### שאלה מתוך מועד א' תשס"ז

- האם ניתן להגדיר התאמה "הפוכה"?
- יש יותר מפתרון אחד ... נסו לפתור בבית

### תרגיל

- הראו כי בשינויים קלים למבנה עץ בינארי (מאוזן), ניתן למצוא את האבר ה-k בגודלו, בזמן  $O(\log k)$

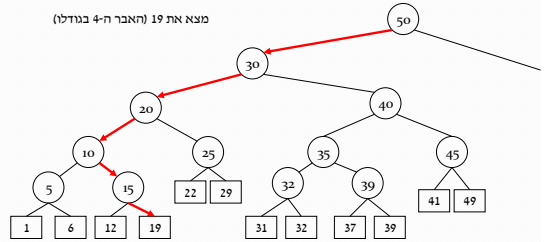
מצא את 19 (האבר ה-4 בגודלו)



### תרגיל

- הראו כי בשינויים קלים למבנה עץ בינארי (מאוזן), ניתן למצוא את האבר ה-k בגודלו, בזמן  $O(\log k)$

מצא את 19 (האבר ה-4 בגודלו)



### תרגיל

- הראו כי בשינויים קלים למבנה עץ בינארי (מאוזן), ניתן למצוא את האבר ה-k בגודלו, בזמן  $O(\log k)$
- פתרון: נניח עץ AVL בעל n צמתים

- מבצע לעלה השמאלי ביותר (קטן)  $p \leftarrow p$
- While (p!=root) do
  - $p \leftarrow p.parent$
  - If (p.key > x) then
    - x is found in left sub-tree so search there
- End while
- We got to the root so perform regular find

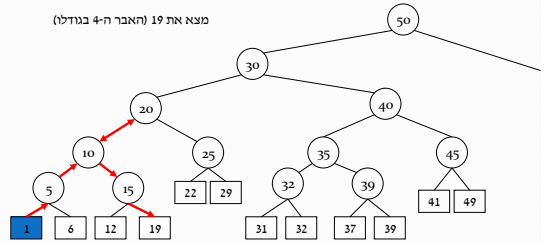
Max.  $O(\log k)$  iterations

$O(k)$  elements in sub-tree  $\rightarrow O(\log k)$  search

### תרגיל

- הראו כי בשינויים קלים למבנה עץ בינארי (מאוזן), ניתן למצוא את האבר ה-k בגודלו, בזמן  $O(\log k)$

מצא את 19 (האבר ה-4 בגודלו)



הסוף