

## טענה על עצים

- **משפט:** בעץ שדרגת כל קודקודיו חסומה ב-3, מספר העלים  $\geq$  מספר הקודקודים הפנימיים + 2.
- מספר העלים  $L$
- מספר הקודקודים הפנימיים  $I$
- מספר הצמתים הכולל  $V=I+L$ ,  $E$  מספר הקשתות.
- לפי הטענה:  $I+2 \geq L$

## מבנה מחשבים

תרגול מספר 3

## טענה על עצים (המשך)

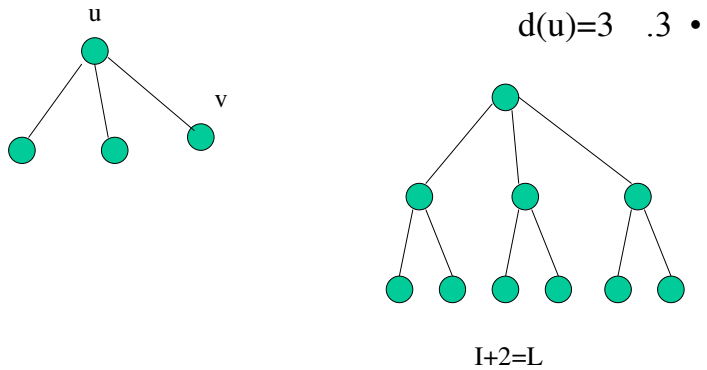
- ניקח עלה  $v$  שלו שכן  $u$  בעל הדרגה המינימלית.
- נסיר את  $v$  מן הגרף ונשתמש בהנחת האינדוקציה על הגרף הקטן יותר.
- נחלק לשלושה מקרים:
- 1.  $d(u)=1$



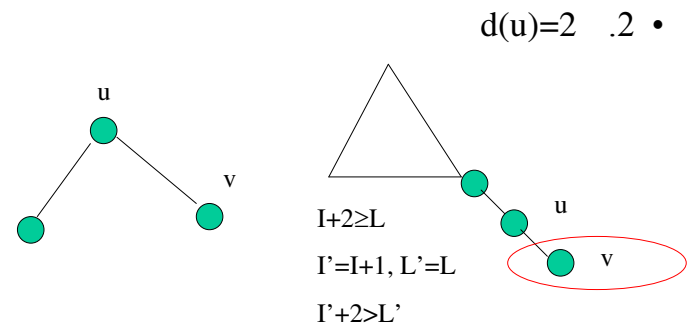
## טענה על עצים (המשך)

- הוכחה: באינדוקציה על  $n$ , מספר הקודקודים בעץ.
- בסיס:  $n=1$  עץ ריק. הטענה טריוויאלית.
- צעד: נניח כי לכל עץ עם עד  $n$  קודקודים ודרגה חסומה ב-3, הטענה נכונה, נוכיח ל  $n+1$  קודקודים.

### טענה על עצים (המשך)



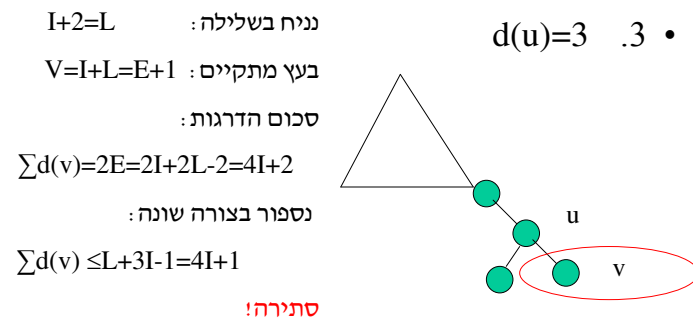
### טענה על עצים (המשך)



### אלגוריתם למיון מערך

- חלק את המערך לשתיים
- מייין כל חצי בנפרד (רקורסיבית)
- מזג את חצי המערכים.
- סיבוכיות:  $T(n) = 2T(n/2) + \Theta(n)$

### טענה על עצים (המשך)



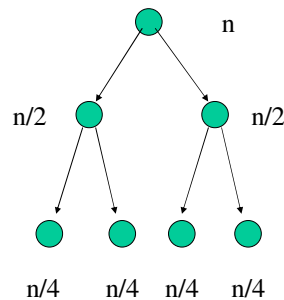
### משפט המאסטר - דוגמאות

- $T(n) = 2T(n/2) + \Theta(n)$ .  $a=2, b=2, \log_2 2=1$
- $f(n) = \Theta(n^1) \rightarrow$
- $T(n) = \Theta(n^{\log_2 2}) \log n = \Theta(n \log n)$  :2 מקרה
- $T(n) = T(n/10) + \Theta(n)$ .  $a=1, b=10, \log_{10} 1 = 0$
- $f(n) = \Omega(n^{0+1})$ ,  $c=1/2, f(n/10) \leq n/2 \rightarrow$
- $T(n) = \Theta(f(n)) = \Theta(n)$ . :3 מקרה

### משוואות נסיגה - משפט המאסטר

- $a, b$  are two constants  $\geq 1$ .  $\epsilon$  is a constant  $> 0$ .
- $f(n)$  is a function.
- $T(n) = aT(n/b) + f(n)$ 
  1. If  $f(n) = O(n^{\log_b a - \epsilon})$  then  $T(n) = \Theta(n^{\log_b a})$
  2. If  $f(n) = \Theta(n^{\log_b a})$  then  $T(n) = \Theta(n^{\log_b a} \log n)$
  3. If  $f(n) = \Omega(n^{\log_b a + \epsilon})$  and  $af(n/b) \leq cf(n)$  for some  $c < 1$ , then  $T(n) = \Theta(f(n))$

### חישוב הסיבוכיות ללא משפט המאסטר



### משפט המאסטר - דוגמאות

- $T(n) = 3T(n/2) + \Theta(n)$ .  $a=3, b=2$   
 $\log_2 3 \approx 1.58 > 1$
- $f(n) = O(n^{\log_2 2}) = O(n)$   $\epsilon \approx 0.58 \rightarrow$
- $T(n) = \Theta(n^{\log_2 3}) = \Theta(n^{1.58...})$  :1 מקרה

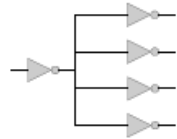
## פתרון לתרגיל בית 2.3

Fan-out(e)

רשת e

In-degree(G), out-degree(G)

שער G



### Question 2.3

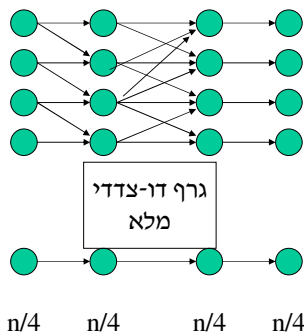
Prove that the total amount of time spent in the relaxation steps is linear in the number nodes if the fan-in of each gate is constant (say, at most 3).

Note that it is not true that each relaxation step can be done in constant time if the fan-in of the gates is not constant.

Prove linear running time in the number of nodes if (i) every net feeds a single input terminal and (ii) the number of outputs of each gate is constant. (You may not assume that the fan-in of every gate is constant.)

## פתרון לתרגיל בית 2.3

• דוגמא ל  $\text{in-degree}(G)$  לא חסום



## פתרון לתרגיל בית 2.3

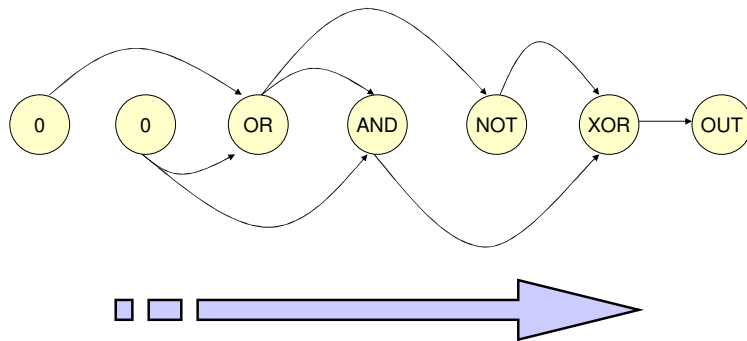
רלקסציה של הרשת  $e_i$

- עדכון  $\text{tpd}$  לכל כניסה שמוזנת ע"י  $e_i$ .
- סימולציה של פונקציונליות: פונקציה בוליאנית על מספר הקלטים של שער  $g$  שפולט את  $e_i$ .
- הסיבוכיות הכוללת

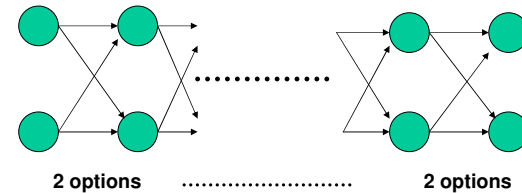
$$\sum_{e_i} O(\text{fanout}(e_i)) + O(\text{in-degree}(G))$$

$G\_feeds\_e_i$

## Finding the Maximum Delay



## A circuit with $2^{n/2}$ paths



Note:  $n/2$  stages with 2 options each, resulting in  $2^{n/2}$  paths.

→ There is no need to check all the paths, only the longest. This takes a linear time.

### Definition:

A Boolean function  $f: \{0,1\}^n \Rightarrow \{0,1\}$  is *monotone* if  $x \geq y \Rightarrow f(x) \geq f(y)$

(where  $x \geq y$  means : for every  $i$   $x_i \geq y_i$  ).

Prove the following claim:

$f: \{0,1\}^n \Rightarrow \{0,1\}$  is *monotone* **iff**  $f$  can be implemented by a combinational circuit that contains only **AND-gates** and **OR-gates**.

## A circuit with $2^n$ paths

- ...cannot be built!
- Why? A combinational circuit is a DAG, therefore we cannot reorder the gates to create different paths. Our only option is to include or exclude gates to create different paths.
- But, having  $n$  gates, we only have  $2^n$  such paths. Each gate can be included or excluded, therefore  $2^n$ .
- We cannot build this circuit since we will require an unbounded in-degree of the gates.