מבני נתונים ואלגוריתמים

תרגול 1

נכתב על ידי שקד לב, 2014

חישוב זמן ריצה - מוטיבציה

אלגוריתם – פתרון בעיה נתונה בעזרת סט של פעולות שונות.

אלגוריתם נחשב תקין אם לכל קלט שנקבל (לרבות מקרי קצה) הוא יסתיים עם פלט (תוצאה) תקינה.

אלגוריתם תקין פותר בעיה חישובית נתונה.

אלגוריתמים המיועדים לפתרון בעיה מסוימת לעתים קרובות **שונים מאוד מבחינת יעילותם.** מדד עיקרי ליעילות האלגוריתם הינו **מהירותו**, כלומר כמה זמן לוקח לו להגיע לתוצאה.

כל פעולה באלגוריתם צורכת זמן, על כן עלינו לדעת כיצד לבנות אלגוריתם נכון.

בשיעור זה נלמד כיצד לחשב זמן ריצה המשקף את היעילות של האלגוריתם.

חישוב זמן ריצה - מוטיבציה מספרית

	10	100	1000
1	1	1	1
$\log_{10} n$	1	2	3
n	10	100	1000
n^2	100	10000	10^6
2^n	1024	מספר בן 31 ספרות	מספר בן 302 ספרות
$n!$	$36x10^6$	מספר בן 161 ספרות	מספר גדול באמת

חישוב זמן ריצה

חישוב זמן ריצה הינו סכימת שורות האלגוריתם (הקוד) הפותר את הבעיה.

עבור פעולות בסיסיות (כולל משפטי תנאי פשוטים):

	Cost	Times
if (n<0)	c_1	1
val= - n;	c_2	1
else		
val= n;	c_3	1

חישוב זמן ריצה

עבור לולאה פשוטה (לולאת for או while העוברת על כל הפרטים בה):

	Cost	Times
i=1;	c_1	1
sum=0;	c_2	1
while (i<=n){	c_3	n+1
i=i+1;	c_4	n (1)
sum= sum+;	c_5	n (1)

חישוב זמן ריצה

עבור לולאה מקוננת (לולאות for או while העוברת על כל הפרטים בה):

	Cost	Times
`i=1;`	c_1	1
`sum=0;`	c_2	1
`while (i<=n){`	c_3	n+1
`    j=1;`	c_4	n
`    while(j<=n){`	c_5	n*(n+1)
`        sum=sum+I;`	c_6	n*n
`        j=j+1;`	c_7	n*n
`    } i=i+1`	c_8	n
`}`		

חישוב זמן ריצה – סיכום

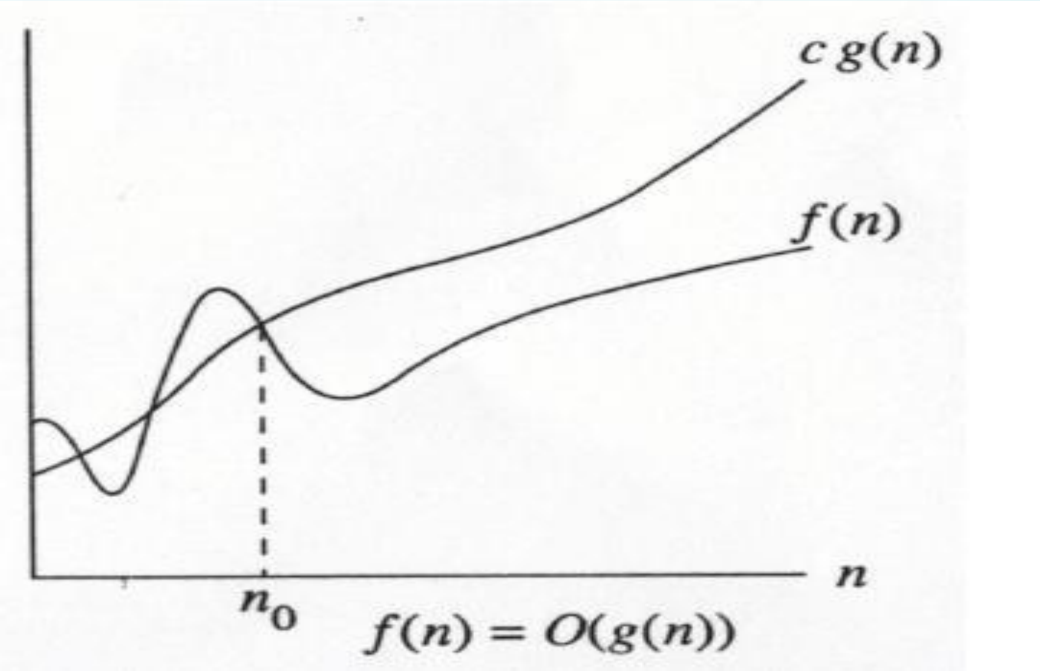
- **<u>משפט תנאי</u>** – לכל היותר לוקח זמן בדיקה ובנוסף זמן מקסימאלי מבין האופציות של התנאי.
- **<u>לולאה</u>** – זמן ביצוע שווה לזמן ביצוע הפעולה בגוף הלולאה מוכפל במס' האיטרציות.
- **<u>לולאה מקוננת</u>** – זמן הביצוע של הפעולה של הלולאה הפנימית (מחושב כמו לולאה רגילה) מוכפל במס' האיטרציות של הלולאה החיצונית.

חסם אסימפטוטי עליון הדוק

נסמן חסם עליון אסימפטוטי הדוק ב-O (האות O גדולה).

<u>הגדרה</u>: נאמר כי f(n)=O(g(n)) אם **קיים** קבוע c כך שעבור **כל** $n>n_0$ מתקיים: $f(n) \le c*g(n)$.

בהסתכלות מתמטית, הרי צריך להתקיים: $\lim_{n\to\infty}\frac{f(n)}{g(n)} \le c < \infty$

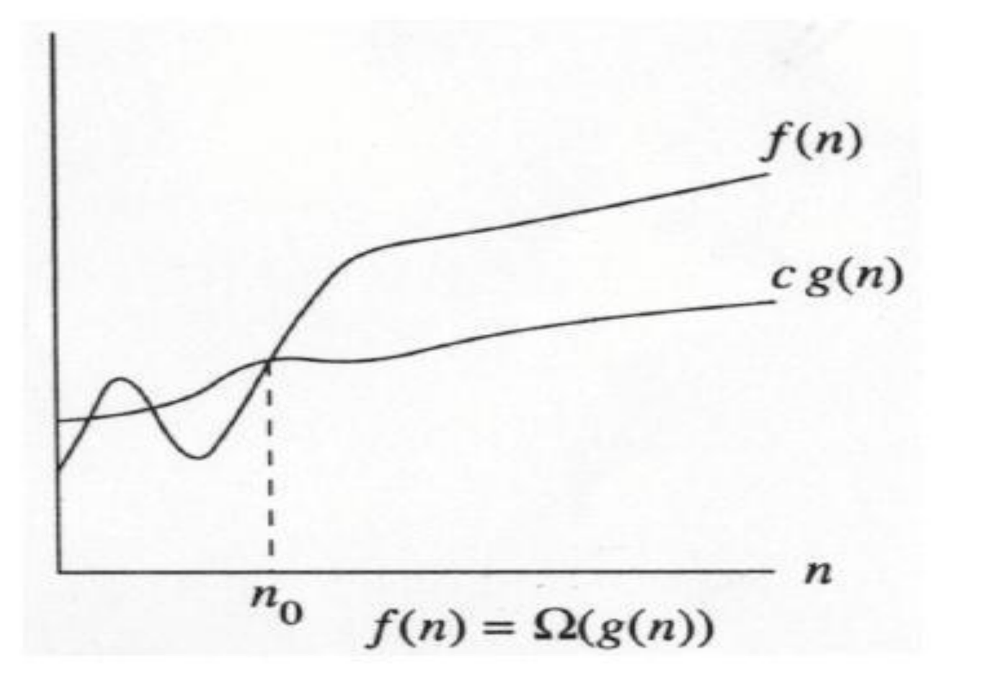


חסם אסימפטוטי תחתון הדוק

נסמן חסם תחתון אסימפטוטי הדוק ב-Ω (האות אומגה גדולה).

<u>הגדרה</u>: נאמר כי $f(n)=\Omega(g(n))$ אם **קיים** קבוע c כך שעבור **כל** $n>n_0$ מתקיים: $f(n)\geq c*g(n)$.

בהסתכלות מתמטית, הרי צריך להתקיים: $\lim_{n\to\infty}\frac{g(n)}{f(n)}\leq c<\infty$

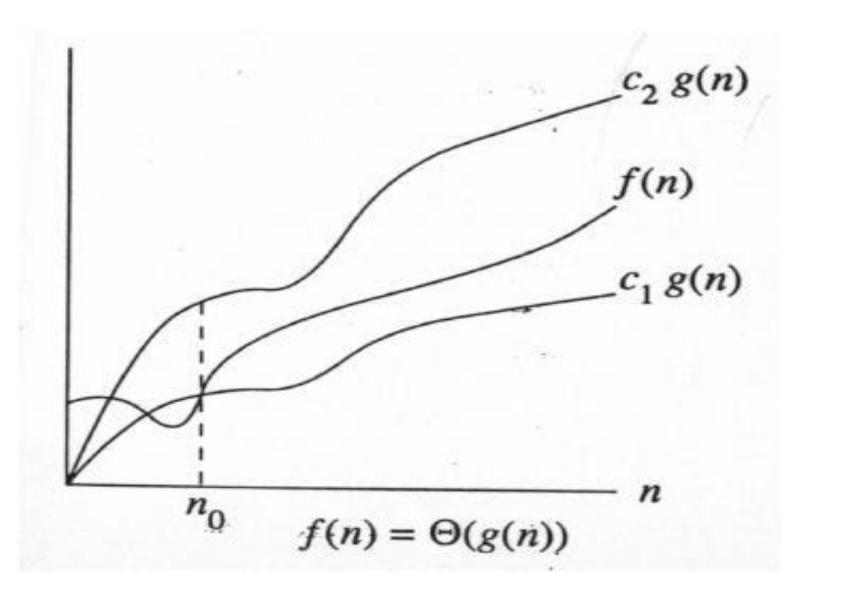


חסם אסימפטוטי הדוק

נסמן חסם אסימפטוטי הדוק ב-Θ (האות טטה).

<u>הגדרה</u>: נאמר כי $f(n)=\Theta(g(n))$ אם מתקיים כי $f(n)=O(g(n))$ וגם $f(n)=\Omega(g(n))$, כלומר **קיימים** קבועים c_1, c_2 כך שעבור **כל** $n>n_0$ מתקיים: $c_1*g(n) \le f(n) \le c_2*g(n)$.

בהסתכלות מתמטית, הרי צריך להתקיים: $\lim_{n\to\infty}\frac{f(n)}{g(n)} = c < \infty$



היררכיה של מחלקות סיבוכיות

- אלגוריתמים "יעילים" יחסית
  - קבוע: $O(1)$
  - לוגריתמי: $O(\log n)$, $O(\log^2 n)$, $\vdots$, $O(\log^k n)$
  - $\vdots$
  - פולינומי: $O(n)$, $O(n^2)$, $\vdots$, $O(n^k)$

$\vdots$

- תת-אקספוננציאלי: $O(2^{\log^2 n})$, $\vdots$, $O(2^{\log^k n})$

$\vdots$

- אקספוננציאלי: $O(2^n)$, $O(3^n)$, $\vdots$, $O(n^n) = O(2^{n \log n})$

$\vdots$

- אקספוננציאלי כפול: $O(2^{2^n})$, $\vdots$