Design and Analysis
of Algorithms I

# Asymptotic Analysis

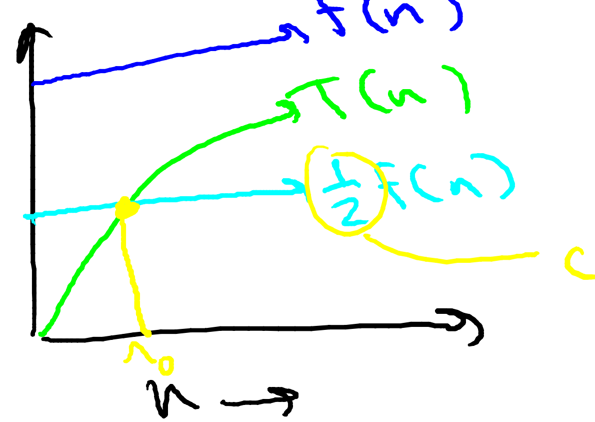
## Big-Oh: Relatives (Omega & Theta)

# Omega Notation

Definition: $T(n) = \Omega(f(n))$ if and only if $\exists$ constants $c, n_0$ such that

$$T(n) \geq c \cdot f(n) \quad \forall n \geq n_0.$$

Picture




$T(n) = \Omega(f(n))$

Tim Roughgarden

# Theta Notation

<u>Definition</u>: $T(n) = \Theta(f(n))$ if and only if $T(n) = O(f(n))$ <u>and</u> $T(n) = \Omega(f(n))$.

<u>Equivalent</u>: $\exists$ constants $c_1, c_2, n_0$ such that

$$c_1 f(n) \le T(n) \le c_2 f(n)$$

for all $n \ge n_0$.

Tim Roughgarden

Let $T(n) = \frac{1}{2}n^2 + 3n$. Which of the following statements are true? (Check all that apply.)

- ☐ $T(n) = O(n)$.
- ☐ $T(n) = \Omega(n)$. $\quad [n_0 = 1,\ c = \frac{1}{2}]$
- ☐ $T(n) = \Theta(n^2)$. $\quad \{n_0 = 1,\ c_1 = \frac{1}{2},\ c_2 = 4\}$
- ☐ $T(n) = O(n^3)$. $\quad \{n_0 = 1,\ c = 4\}$

# Little-Oh Notation

Definition: $T(n) = o(f(n))$ if and only if for all constants $c > 0$, $\exists$ a constant $n_0$ such that

$$T(n) \leq c \cdot f(n) \quad \forall n \geq n_0.$$

Exercise: for all $k \geq 1$, $n^{k-1} = o(n^k)$.

Tim Roughgarden

# Where Does Notation Come From?

"On the basis of the issues discussed here, I propose that members of SIGACT, and editors of compter science and mathematics journals, adopt the $O$, $\Omega$, and $\Theta$ notations as defined above, unless a better alternative can be found reasonably soon".

*-D. E. Knuth, "Big Omicron and Big Omega and Big Theta", SIGACT News, 1976. Reprinted in "Selected Papers on Analysis of Algorithms."*