COMP 250

Lecture 16

big O ...., big Omega \( \Omega \), big Theta \( \Theta \)

Oct. 17, 2016
Recall motivation for O( )

The time it takes to perform instructions depends on:

• the size of the input (n, m, N, ...)

• implementation details that are unknown
  (various constants c1, c2, .... )
Example: Grade School Addition

\[
\begin{align*}
\text{carry} &= 0 \\
\text{for } i &= 0 \text{ to } N - 1 \text{ do} \\
 & \quad r[i] \leftarrow (a[i] + b[i] + \text{carry}) \mod 10 \\
 & \quad \text{carry} \leftarrow (a[i] + b[i] + \text{carry})/10 \\
\text{end for} \\
\text{r}[N] &\leftarrow \text{carry}
\end{align*}
\]
Example: Grade School Addition

\[ t(n) = c_0 + c_1 N \]

Constants \( c_0, c_1 \) ... include time for:

- Primitive ops +, -, *, /, %
- Array address indexing
- Array get or set
- Assignment
- For loop administration
- Recursive call administration
- .........
Recall last lecture: big O

$t(n)$ is $O(g(n))$. 
Formal Definition of Big O

Let $t(n)$ and $g(n)$ be two functions, where $n \geq 0$.

We say $t(n)$ is $O(g(n))$, if there exist two positive constants $n_0$ and $c$ such that, for all $n \geq n_0$, 

$$t(n) \leq c \cdot g(n).$$
Never write $O(3n), O(5 \log_2 n)$, etc.

The point of big O notation is to avoid dealing with constants.

It is still technically correct to write the above. We just don’t do it.
Sets of $O()$ functions

Each of the following holds for $n$ sufficiently large:

$$1 < \log_2 n < n < n \log_2 n < n^2 < n^3 < \ldots < 2^n < n!$$
Sets of $\mathcal{O}()$ functions

Each of the following holds for $n$ sufficiently large:

$$1 < \log_2 n < n < n \log_2 n < n^2 < n^3 < \ldots < 2^n < n!$$

Suppose $t(n)$ is $O(\ g(n)\ )$, and $g(n) < h(n)$ for $n \geq n_0$.

Then, $t(n)$ is $O(\ h(n)\ )$.

E.g. if $t(n)$ is $O(\ n\ )$, then $t(n)$ is $O(\ n^2\ ), O(\ n^3\ ), \ldots$
Sets of $O(\cdot)$ functions

If $t(n)$ is $O(g(n))$, one often writes $t(n) \in O(g(n))$.

That is, $t(n)$ is a member of the set of functions that are $O(g(n))$. 
Sets of $O()$ functions

We have the following strict subset relationships:

$O(1) \subset O(\log_2 n) \subset O(n) \subset O(n \log_2 n) \subset O(n^2) \ldots$

$\subset O(n^3) \subset \ldots \subset O(2^n) \subset O(n!)$
Asymptotic lower bound

The constant $n_0$ is not unique.
Preliminary Definition

$t(n)$ is asymptotically bounded below by $g(n)$ if there exists an $n_0$ such that, for all $n \geq n_0$,

$$t(n) \geq g(n).$$
Example: \( t(n) = \frac{n(n-1)}{2} \) is asymptotically bounded below by \( g(n) = \frac{n^2}{4} \).

Proof:

\[
\frac{n(n-1)}{2} \geq \frac{n^2}{4}
\]

\( \iff \quad 2n(n - 1) \geq n^2 \)

\( \iff \quad n^2 \geq 2n \)

\( \iff \quad n \geq 2 \quad \text{So take} \quad n_0 = 2. \)
Definition of Big Omega ($\Omega$)

Let $t(n)$ and $g(n)$ be two functions of $n \geq 0$.

We say $t(n)$ is $\Omega(g(n))$, if there exist two positive constants $n_0$ and $c$ such that, for all $n \geq n_0$,

$$t(n) \geq c \cdot g(n).$$
Claim: $\frac{n(n-1)}{2}$ is $\Omega(n^2)$.

Proof (1): Use $c = \frac{1}{4}$ from two slides ago.

\[
\frac{n(n-1)}{2} \geq \frac{n^2}{4}
\]

$\iff$ :

$\iff n \geq 2$ So take $n_0 = 2, \ c = \frac{1}{4}$.
Claim: \( \frac{n(n-1)}{2} \) is \( \Omega(n^2) \).

Proof (2): Try \( c = \frac{1}{3} \)

\[
\frac{n(n-1)}{2} \geq \frac{n^2}{3}
\]

\[\iff n \geq 3\]

So take \( n_0 = 3 \), \( c = \frac{1}{3} \).
Sets of $\Omega()$ functions

Claim: Suppose $t(n)$ is $\Omega(g(n))$, and $g(n) > h(n)$ for $n \geq n_0$.

Then, $t(n)$ is $\Omega(h(n))$. Proof follows straight from definition.

e.g. if $t(n)$ is $\Omega(n^3)$, then $t(n)$ is $\Omega(1)$, $\Omega(n)$, ..., $\Omega(n^2)$
Sets of $\Omega(\cdot)$ functions

If $t(n)$ is $\Omega(g(n))$, one often writes $t(n) \in \Omega(g(n))$.

That is, $t(n)$ is a member of the set of functions that are $\Omega(g(n))$. 
Sets of $\Omega()$ functions

Thus, we have the following strict subset relationships:

$\Omega(1) \subset \Omega(\log_2 n) \subset \Omega(n) \subset \Omega(n \log_2 n) \subset \Omega(n^2) \ldots$

$\subset \Omega(n^3) \subset \ldots \Omega(2^n) \subset \Omega(n!)$
Definition of Big Theta ($\Theta$)

Let $t(n)$ and $g(n)$ be two functions of $n \geq 0$.

We say $t(n)$ is $\Theta(g(n))$, if there exist three positive constants $n_0, c_1, c_2$ such that for all $n \geq n_0$,

$$c_1 \ g(n) \leq t(n) \leq c_2 \ g(n)$$
Definition of Big Theta (Θ)

Let \( t(n) \) and \( g(n) \) be two functions of \( n \geq 0 \).

We say \( t(n) \) is \( \Omega(g(n)) \), if there exist three positive constants \( n_0, \ c_1, \ c_2 \) such that for all \( n \geq n_0 \),

\[
c_1 \ g(n) \leq t(n) \leq c_2 \ g(n)
\]

\( t(n) \) is \( O(g(n)) \)
Definition of Big Theta ($\Theta$)

Let $t(n)$ and $g(n)$ be two functions of $n \geq 0$.

We say $t(n)$ is $\Omega( g(n) )$, if there exist three positive constants $n_0$, $c_1$, $c_2$ such that for all $n \geq n_0$,

$$c_1 \, g(n) \leq t(n) \leq c_2 \, g(n)$$

$t(n)$ is $\Omega( g(n) )$
Example

\[ t(n) \text{ is } \Theta(g(n)). \]
Example

Let $t(n) = 4 + 17 \log_2 n + 3n + 9n \log_2 n + \frac{n(n-1)}{2}$

$t(n)$ is $\Theta(\ ? )$. 
Example

Let \( t(n) = 4 + 17 \log_2 n + 3n + 9n \log_2 n + \frac{n(n-1)}{2} \)

Claim: \( t(n) \) is \( \Theta(n^2) \).

Proof:
\[
\frac{n^2}{4} \leq t(n) \leq (4 + 17 + 3 + 9 + \frac{1}{2}) n^2
\]
Can we write $\Theta()$ for every $t(n)$?

No, as this contrived example shows:

Let $t(n) = \begin{cases} 
  n, & n \text{ is odd} \\
  n^2, & n \text{ is even.} 
\end{cases}$

$t(n)$ is $O(n^2)$ but not $O(n)$.

$t(n)$ is $\Omega(n)$ but not $\Omega(n^2)$.

There does not exist a $\Theta()$ bound for this $t(n)$.
Algorithm best and worst cases

vs.

$\Theta()$, $\Omega()$, $\Theta()$.

What is relationship between these ideas? See Exercises.
Sets of $\Theta()$ functions

If $t(n)$ is $\Theta(g(n))$, one often writes $t(n) \in \Theta(g(n))$,

That is, $t(n)$ is a member of the set of functions that are $\Theta(g(n))$. 

\[\Theta(1) \quad \Theta(\log_2 n) \quad \Theta(n) \quad \ldots \quad \Theta(2^n) \quad \Theta(n!)\]