Data Structures and Algorithms: Lecture 2

Barbara Morawska

July 31, 2018

Growth of functions

Asymptotic efficiency of algorithms

How the running time increases with the size of the input in the limit. Asymptotically more efficient algorithm is better for all but very small inputs.

Running time is expressed as a function of the size of input

Asymptotic notation applies to functions... thus to the running time of an algorithm.

Example

Insertion sort running time function: $T(n) = an^2 + bn + c$, where a, b, c are constants.
This function is in $\Theta(n^2)$.
$T(n) = \Theta(n^2)$

Θ notation

Assume that $f(n)$ is asymptotically non-negative,
i.e. $f(n)$ is non-negative for sufficiently large n.

Definition

$$
\begin{aligned}
\Theta(g(n)) = \{f(n) \mid\ & \exists c_1, c_2 - \text{ constants}, \\
& \exists n_0 - \text{ non-negative integer}, \\
& 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \\
& \text{for all } n \geq n_0\}
\end{aligned}
$$

We say that $g(n)$ is asymptotically tight bound for $f(n)$.

Example of Θ notation

Show $\frac{1}{2}n^2 - 3n = \Theta(n^2)$:

Determine c_1, c_2, n_0 such that:

$$c_1 n^2 \leq \frac{1}{2}n^2 - 3n \leq c_2 n^2$$

for all $n \geq n_0$.

Divide by n^2:

$$c_1 \leq \frac{1}{2} - \frac{3}{n} \leq c_2$$

We can choose: $n \geq 1, c_2 \geq \frac{1}{2}$.

But for c_1? $\frac{1}{2} - \frac{3}{1} = -2\frac{1}{2}$ $\frac{1}{2} - \frac{3}{2} = -1$ $\frac{1}{2} - \frac{3}{3} = -\frac{1}{2}$ $\frac{1}{2} - \frac{3}{4} = -\frac{1}{4}$
$\frac{1}{2} - \frac{3}{5} = -\frac{1}{10}$ $\frac{1}{2} - \frac{3}{6} = 0$ $\frac{1}{2} - \frac{3}{7} = \frac{1}{14}$

Example of Θ notation

Show $6n^3 \neq \Theta(n^2)$:

Proof by contradiction.

- Assume: $\exists c_2, n_0 \quad \forall n \geq n_0 \quad (6n^3 \leq c_2 n^2)$.
- Divide by n^2: $6n \leq c_2$.
- Then: $n \leq \frac{c_2}{6} \leftarrow$ constant!
- Impossible for large n.

Asymptotic bound is determined by the highest-order term in a polynomial.

Asymptotic bound for polynomial

Example

- $f(n) = an^2 + bn + c$, where $a > 0, a, b, c$ - constants
- $f(n) = \Theta(n^2)$,
- Claim: $c_1 = \frac{a}{4}, \quad c_2 = \frac{7a}{4}, \quad n_0 = 2|max(\frac{|b|}{a}, \sqrt{\frac{|c|}{a}})|$

Proof

(Sketch) If $c_1 = \frac{a}{4}$ what should be n_0?

- $\frac{a}{4}n^2 \leq an^2 + bn + c$
- $0 \leq an^2 - \frac{a}{4}n^2 + bn + c$
- $0 \leq \frac{3}{4}an^2 + bn + c$
- $n_0 \geq \frac{\sqrt{b^2-3ac}-b}{\frac{3a}{2}} = 2\frac{\sqrt{b^2-3ac}-b}{3a}$
- $\frac{|b|}{a} \geq \sqrt{\frac{|c|}{a}}$ implies $4b^2 \geq b^2 - 3ac$
- and $\frac{|b|}{a} < \sqrt{\frac{|c|}{a}}$ implies $4ac > b^2 - 3ac$

Proof (cnt.)

Easy to show:

if $\frac{|b|}{a} \geq \sqrt{\frac{|c|}{a}}$ then $4b^2 \geq b^2 - 3ac$

if $\frac{|b|}{a} < \sqrt{\frac{|c|}{a}}$ then $4ac > b^2 - 3ac$

In order to get $n_0 \geq 2\frac{\sqrt{b^2-3ac}-b}{3a}$

in the first case:

$2\frac{\sqrt{4b^2}-b}{3a} = 2\frac{2|b|-b}{3a} \leq 2\frac{|b|}{a}$

Hence we can require that $n_0 = 2\frac{|b|}{a}$

in the second case:

$2\frac{\sqrt{4a|c|}-b}{3a} = 2\frac{2\sqrt{a|c|}-b}{3a} < \frac{2}{3}\frac{3\sqrt{a|c|}}{a} = 2\sqrt{\frac{a|c|}{a^2}} = 2\sqrt{\frac{|c|}{a}}$

Hence we can require that $n_0 = 2\sqrt{\frac{|c|}{a}}$

Conclusion: we can require $n_0 = 2\max\{\frac{|b|}{a}, \sqrt{\frac{|c|}{a}}\}$

Similar for $c_2 = \frac{7a}{4}$ we have to show that the Θ notation holds for $n > n_0$.

Polynomials in Θ-notation

In general...

- A polynomial in one variable n has form:
- $p(n) = a_0 + a_1 n + a_2 n^2 + \cdots + a_d n^d = \Sigma_{i=0}^{d} a_i n^i$
  where $a_i, i = \{0, \ldots, d\}$ are constants and $a_d > 0$.
- Then $p(n) = \Theta(n^d)$
- Notice: A polynomial of 0-degree is a constant $a_0 n^0 = a_0$.
  Hence $a_0 = a_0 n^0 = \Theta(n^0) = \Theta(1)$.

Big-O notation

Definition

$$
\begin{aligned}
O(g(n)) = \{f(n) \mid \exists c - &\text{ constant} \\
\exists n_0 - &\text{ non-negative integer}, \\
0 \leq &f(n) \leq cg(n) \\
\text{for all } &n \geq n_0\}
\end{aligned}
$$

- Notice: $f(n) = \Theta(g(n))$ implies $f(n) = O(g(n))$.
- In other words: $\Theta(g(n)) \subseteq O(g(n))$

Example

$an + b \in O(n^2)$, where $a > 0$
Verify that $c = a + |b|$ and $n_0 = \max(1, -\frac{b}{a})$ works.

Example (cnt.)

Show: $an + b \in Q(n^2)$, where $a > 0$
$c = a + |b|$ and $n_0 = \max(1, -\frac{b}{a})$

- Show for which n, $an + b \leq (a + |b|)n^2$.
- Hence $0 \leq (a + |b|)n^2 - an - b$
- $$n \geq \frac{a + \sqrt{a^2 - 4(a + |b|)(-b)}}{2(a + |b|)}$$
  $$= \frac{a + \sqrt{(a + 2b)^2}}{2(a + |b|)} = \frac{a + a + 2b}{2(a + |b|)} = 1$$
- Or:

$$n \geq \frac{a - \sqrt{a^2 - 4(a + |b|)(-b)}}{2(a + |b|)}$$
$$= \frac{a - \sqrt{(a + 2b)^2}}{2(a + |b|)} = \frac{a - a - 2b}{2(a + |b|)} = -\frac{2b}{2(a + |b|)} < -\frac{b}{a}$$

Big-O notation – asymptotic upper bound of running time

- Big-O notation gives upper bound for all cases of running time.
- It is not true of Θ-notation:
  - the worst case of insertion sort is bounded by $\Theta(n^2)$
  - if the input is sorted (the best case), it is bounded by $\Theta(n)$.
    Hence in the best case $c_1 n < T(n)$,
    but not $c_1 n^2 < T(n)$ for arbitrary great n.

Ω-notation – asymptotic lower bound

Definition

$$\begin{aligned}\Omega(g(n)) = \{f(n) \mid\ & \exists c - \text{ constant} \\ & \exists n_0 - \text{ non-negative integer}, \\ & 0 \le cg(n) \le f(n) \\ & \text{for all } n \ge n_0\}\end{aligned}$$

Theorem

For any two functions $f(n), g(n)$,

$$f(n) = \Theta(g(n)) \textit{ if and only if } f(n) = O(g(n)) \textit{ and } f(n) = \Omega(g(n))$$

Ω-notation

lower bound

- Note: Ω-notation provides lower bound for any running time (best, worst).
- Example: Insertion sort runs in $\Omega(n)$ and $O(n^2)$.

Asymptotic notation in equations

Representing anonymous function

$2n^2 + 3n + 1 = 2n^2 + \Theta(n)$
means $2n^2 + 3n + 1 = 2n^2 + f(n)$ where $f(n) \in \Theta(n)$.
For example $f(n) = 3n + 1$
For example $T(n) = 2T(\frac{n}{2}) + \Theta(n)$

Notice:

$$\Sigma_{i=1}^{n} O(i) \neq O(1) + O(2) + \cdots + O(n)$$

$O(i)$ on the left represents one anonymous function. On the right we can have different functions.

$$2n^2 + \Theta(n) = \Theta(n^2)$$

means: for a choice of an anonymous function on the left side, we can choose an anonymous function on the right side.

Asymptotic notation in equations (cnt.)

Let $f(n) \in \Theta(n), g(n) \in \Theta(n^2)$

- $2n^2 + f(n) = g(n)$ for all n
- $2n^2 + \Theta(n) = \Theta(n^2)$
- $$\begin{aligned} 2n^2 + 3n + 1 &= 2n^2 + \Theta(n) \\ &= \Theta(n^2) \end{aligned}$$

o-notation

Definition

- Big O: $f(n) \in O(g(n))$ iff for some $c > 0$, $0 \leq f(n) \leq cg(n)$ $(n \geq n_0)$
- small o: $f(n) \in O(g(n))$ iff for all $c > 0$, $0 \leq f(n) \leq cg(n)$ $(n \geq n_0)$
  - $f(n)$ is much smaller (insignificant) wrt. $g(n)$ when n grows to infinity
  - $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$

ω-notation

Definition

- Big Ω: $f(n) \in \Omega(g(n))$ iff for some $c > 0$, $0 \leq cg(n) \leq f(n)$ $(n \geq n_0)$
- small ω: $f(n) \in O(g(n))$ iff for all $c > 0$, $0 \leq cg(n) \leq f(n)$ $(n \geq n_0)$
  - $g(n)$ is much smaller (insignificant) wrt. $f(n)$ when n grows to infinity
  - $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$
- $f(n) \in \Omega(g(n))$ iff $g(n) \in O(f(n))$
- $f(n) \in \omega(g(n))$ iff $g(n) \in o(f(n))$

Example

$\frac{n^2}{2} = \omega(n)$ but $\frac{n^2}{2} \neq \omega(n^2)$

Comparing functions wrt to the growth

Transitivity

- $f(n) = \Theta(g(n))$ and $g(n) = \Theta(h(n))$ implies $f(n) = \Theta(h(n))$
- $f(n) = O(g(n))$ and $g(n) = O(h(n))$ implies $f(n) = O(h(n))$
- $f(n) = \Omega(g(n))$ and $g(n) = \Omega(h(n))$ implies $f(n) = \Omega(h(n))$
- $f(n) = o(g(n))$ and $g(n) = o(h(n))$ implies $f(n) = o(h(n))$
- $f(n) = \omega(g(n))$ and $g(n) = \omega(h(n))$ implies $f(n) = \omega(h(n))$

Reflexivity

- $f(n) = \Theta(f(n))$
- $f(n) = O(f(n))$
- $f(n) = \Omega(f(n))$

Comparing functions wrt to the growth

Symmetry

$f(n) = \Theta(g(n))$ iff $g(n) = \Theta(f(n))$

Transpose symmetry

- $f(n) = O(g(n))$ iff $g(n) = \Omega(f(n))$
- $f(n) = o(g(n))$ iff $g(n) = \omega(f(n))$

Comparison to $\leq, <, =$

- $f(n) = O(g(n))$ compares to $a \leq b$
- $f(n) = \Omega(g(n))$ compares to $a \geq b$
- $f(n) = \Theta(g(n))$ compares to $a = b$
- $f(n) = o(g(n))$ compares to $a < b$
- $f(n) = \omega(g(n))$ compares to $a > b$

But not all functions are comparable
($f(n) = O(g(n))$ or $f(n) = \Omega(g(n))$).