CS 240 — Lecture 2

Bartosz Antczak *Instructor: Eric Schost* *January 5, 2017*

2.1 More on order notation

2.1.1 Proofs with big-O notation

Proving that $f(n) \in O(g(n))$ from *first principles* means that you have to find one value of c and n_0 such that $0 \leq f(n) \leq c\, g(n), \forall\, n \geq n_0$. Let's see some examples:

Example 2.1.1. *Prove that* $2n^2+3n+11 \in O(n^2)$*. (To prove this, we use a similar approach to the example from last lecture).*

For $n \geq 1$,

$$11 \leq 11n^2 \tag{2.1}$$
$$3n \leq 3n^2 \tag{2.2}$$
$$2n^2 \leq 2n^2 \tag{2.3}$$
$$2n^2 + 3n + 11 \leq 16n^2 \qquad \text{adding (2.1), (2.2) and (2.3)} \tag{2.4}$$

Let $c = 16$ and $n_0 = 1$. By first principles, $2n^2 + 3n + 11 \in O(n^2)$.

Example 2.1.2. *Prove that* $(n + 1)^5 \in O(n^5)$*.*

For $n \geq 1$,

$$n + 1 \leq 2n \tag{2.5}$$
$$(n + 1)^5 \leq (2n)^5 \tag{2.6}$$
$$(n + 1)^5 \leq 32n^5 \tag{2.7}$$

Let $c = 32$ and $n_0 = 1$. By first principles, $(n + 1)^5 \in O(n^5)$.

Example 2.1.3. *Prove that* $n^2 + n \log_2(n) \in O(n^2)$*.*

For $n \geq 1$,

$$\log_2(n) \leq n \tag{2.8}$$
$$n \log_2(n) \leq n^2 \qquad \text{multiply (2.8) by } n \tag{2.9}$$
$$n^2 \leq n^2 \tag{2.10}$$
$$n^2 + n \log_2(n) \leq 2n^2 \tag{2.11}$$

Let $c = 2$ and $n_0 = 1$. By first principles, $n^2 + n \log_2(n) \in O(n^2)$.
Later on we'll prove big-O notation using other methods, but right now we're just learning the basics.

2.1.2 Some rules with big-O notation

Suppose $f(n) \geq 0$ and $g(n) \geq 0$.

1) If $a > 0$, then $f(n) \in O(a\,f(n))^*$ and $a\,f(n) \in O(f(n))^{**}$

2) If $f(n) \in O(g(n))$ and $g(n) \in O(h(n))$, then $f(n) \in O(h(n))^{***}$

Proof of (*):

$$0 \leq f(n) \leq \frac{1}{a} \cdot a \cdot f(n) \qquad \forall\, n \geq 1 \tag{2.12}$$

We let $c = \frac{1}{a}$ and $n_0 = 1$, and using first principles we have proven that $f(n) \in O(a\,f(n))$.

Proof of (**):

$$0 \leq a\,f(n) \leq a\,f(n) \qquad \forall\, n \geq 1 \tag{2.13}$$

We let $c = a$ and $n_0 = 1$, and using first principles we have proven that $a\,f(n) \in O(f(n))$.

Proof of (***):
By our assumption, there exists c_1 and n_1 such that

$$0 \leq f(n) \leq c_1\,g(n) \qquad n \geq n_1 \tag{2.14}$$

and also there exists c_2 and n_2 such that

$$0 \leq g(n) \leq c_2\,h(n) \qquad n \geq n_2 \tag{2.15}$$

Now, if $n \geq n_1$ and $n \geq n_2$

$$0 \leq f(n) \leq c_1\,g(n) \leq c_1\,c_2\,h(n) \tag{2.16}$$

We let $c = c_1 c_2$ and $n_0 = \max(n_1, n_2)$, and using first principles we have proven $f(n) \in O(h(n))$.

2.1.3 Proofs with big-Omega notation

Big-Omega notation is the reverse of big-O notation (rather than f being at most $O(n)$, f is at least $\Omega(n)$). To prove that $f(n) \in \Omega(g(n))$, we have to find one value c and one integer n_0 such that $0 \leq c\,g(n) \leq f(n)$ (this is first principles).

Example 2.1.4. *Prove that* $n^3 \log_2(n) \in \Omega(n^3)$.

For all $n \geq 2$

$$\log_2(n) \geq 1 \tag{2.17}$$

$$n^3 \log_2(n) \geq n^3 \qquad \text{multiply 2.17 by } n^3 \tag{2.18}$$

We take $c = 1$ and $n_0 = 2$. By first principles we have proven $n^3 \log_2(n) \in \Omega(n^3)$.

2.1.4 Proofs with big-Theta notation

To prove this, we use the same approach as we did with big-O and big-Omega notation.

2.1.5 Little-O notation

$f(n) \in o(g(n))$: this notation is similar to big-O notation, but we use this when we want to say that $f(n)$ is *much less* than $o(g(n))$ (rather than less than or equal to $O(n)$).
To prove that $f(n) \in o(g(n))$, we are given $c > 0$, and you have to find n_0 such that $0 \leq f(n) < c\,g(n)$, $\forall\, n > n_0$.

Example 2.1.5. *Prove that $n \in o(n^2)$, given that $c > 0$*

We have to find n_0 such that

$$\begin{aligned} & n < c\,n^2 \qquad n \geq n_0 && (2.19)\\ \iff\ & 1 < c\,n && (2.20)\\ \iff\ & \frac{1}{c} < n && (2.21)\end{aligned}$$

We take $n_0 = \frac{1}{c}$ and using first principles, we have proven $n \in o(n^2)$.

Example 2.1.6. *Prove that $2010n^2 + 1388n \in o(n^3)$, given that $c > 0$*

We have to find n_0 such that $2010n^2 + 1388n < c\,n^3$ for $n \geq n_0$.
For $n \geq 1$,

$$\begin{aligned} n &\leq n^2 && (2.22)\\ 1388n &< 1388n^2 && (2.23)\\ \implies 2010n^2 + 1388n &< 3398\,n^2 < 4000n^2 && (2.24)\end{aligned}$$

To finish this proof, we just have to find n_0 such that $4000n^2 < c\,n^3$ for $n \geq n_0$

$$4000n^2 < c\,n^3 \iff 4000 < c\,n \iff \frac{4000}{c} < n \tag{2.25}$$

We take $n_0 = \frac{4000}{c}$ and using first principles, we have proven $2010n^2 + 1388n \in o(n^3)$.

2.1.6 Comparing big-O and little-O

Example 2.1.7. $O(1)$ *and* $o(1)$

1) $f(n) \in O(1)$ means that f is bounded and that there exists a constant M such that $0 \leq f(n) \leq M$
2) $f(n) \in o(1)$ means that $\lim\limits_{n\to\infty} f(n) = 0$

2.2 Complexity of Algorithms

Let $T_A(I)$ denote the running time of an algorithm A on instance I. We consider two cases for T:

- **Average-case**: the average running time of A over all instances of size n
- **Worst-case**: the longest running time of A over all instances of size n