Asymptotic Notation Summary

January 9, 2001

1 Background

When analyzing the running time or storage requirements of an algorithm we ideally want exact formulas. These formulas are hard to obtain for the following reasons:

- The answer might be computer/compiler specific. i.e. The time it takes to perform `a = b + c;` is computer specific, the amount of space taken up by an `int` is compiler specific.
- In most "real world" algorithms even when all the above parameters are known an exact solution is mathematicly hard (and tedious) to obtain.

To get around these problems we use "asymptotic notation". Asymptotic notation has the following advantages:

- Shows us the "big picture", i.e. how a function "behaves".
- Hides the computer/compiler specific constants.
- Makes the analysis of the algorithm mathematicly simpler.

2 "big Oh" notation

The "big Oh" notation was introduced by Paul Bachman in 1894 and popularized in the field of analysis of algorithms by Knuth. The definition:

$$O(g(n)) \triangleq \{f(n) : \exists c > 0, n_0 > 0 \, \forall n > n_0 \; 0 \leq f(n) \leq cg(n)\}$$

Notice that $O(g(n))$ is a *set* of functions. However, the convention is to write $3n^2 = O(n^2)$ and not $3n^2 \in O(n^2)$. Don't let the equal sign confuse

you: $n^2 = O(n^2)$ and $3n = O(n^2)$ *does not* imply $n^2 = 3n$. It is customary to always write the O on the right hand side of the equation so we remember this fact.

What does the "big Oh" notation mean, intuitively? if $f(n) = O(g(n))$ then we can think of $f(n)$ as being "not greater than" $g(n)$ (sort of). More formally, if $f(n) = O(g(n))$ then, beyond a sufficiently large n_0, $g(n)$ is at most a constant multiple of $f(n)$. Examples: $10000 = O(n^2)$, $n = O(n^2)$, $4n^2 + 15n - 17 = O(n^2)$, $\frac{1}{1000}n^3 \neq O(n^2)$. We use the "big Oh" notation to give an upper bound, up to a multiplicative constant and an n_0, of a function.

3 "big Omega" and "big Theta"

We used the O notation to define asymptotic upper bounds. Let's define an analogous notation for lower bounds:

$$\Omega(g(n)) \triangleq \{f(n) : \exists c > 0, n_0 > 0 \, \forall n > n_0 \; 0 \leq c\,g(n) \leq f(n)\}$$

As before, $\Omega(g(n))$ is a set of functions. We use the equal sign and write the Ω on the right hand side of the equation.

What does the "big Omega" notation mean, intuitively? If $f(n) = \Omega(g(n))$ then we can think of $f(n)$ as being "not less than" $g(n)$ (sort of). More formally, if $f(n) = \Omega(g(n))$ then, beyond a sufficiently large n_0, $f(n)$ is at least a constant multiple of $g(n)$. Examples: $\frac{1}{10}n^2 = \Omega(n)$, $n^2 = \Omega(1)$, $n^2 \neq \Omega(n^3)$. We use the "big Omega" notation to give a lower bound of a function, up to a multiplicative constant and an n_0.

What if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$? This happens if and only if $f(n) = \Theta(g(n))$ (definition). Examples: $5n^4 = \Theta(n^4)$, $n^4 \neq \Theta(n^2)$, $5 + 3\sin(n) = \Theta(1)$. A notation less used is: $f(n) \asymp g(n)$.

4 "small oh" and "small omega"

These notations are used less frequently. Assume f(n) and g(n) are positive from a certain n_0 onward (asymptotically positive). Define:

$$f(n) = o(g(n)) \iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

meaning $f(n)$ is asymptotically (much) smaller than $g(n)$. Examples: $n = o(n^2)$, $n^2 \neq o(n^2)$. A notation less used is $f(n) \prec g(n)$

Define:

$$f(n) = \omega(g(n)) \iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

meaning $f(n)$ is asymptotically (much) bigger than $g(n)$. Examples: $n^2 = \omega(n)$, $n \neq \omega(n)$. The other notation for this is: $f(n) \succ g(n)$.

Define:

$$f(n) \sim g(n) \iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = 1$$

meaning $f(n)$ is asymptotically equal to $g(n)$. Examples: $1 + \frac{1}{n} \sim 1$, $n^2 + n \sim n^2$, $2n \not\sim 3n$. I have never seen this definition used in analysis of algorithms, I have included it for completeness.

5 Use in expressions

What does the following expression mean?

$$f(n) = n^2 + O(1)$$

It means: $f(n) = n^2 + l(n)$, where $l(n) = O(1)$. Thus, the notations described above can also be used in mathematical equations, very helpful...

6 Results

Assume $f(n)$ and $g(n)$ are asymptotically positive. Following is a list of equations resulting from the above definitions (easily proved):

- Transitivity
  1. $f(n) = \Theta(g(n)) \wedge g(n) = \Theta(h(n)) \Rightarrow f(n) = \Theta(h(n))$
  2. $f(n) = O(g(n)) \wedge g(n) = O(h(n)) \Rightarrow f(n) = O(h(n))$
  3. $f(n) = \Omega(g(n)) \wedge g(n) = \Omega(h(n)) \Rightarrow f(n) = \Omega(h(n))$
  4. $f(n) = o(g(n)) \wedge g(n) = o(h(n)) \Rightarrow f(n) = o(h(n))$
  5. $f(n) = \omega(g(n)) \wedge g(n) = \omega(h(n)) \Rightarrow f(n) = \omega(h(n))$
  6. $f(n) \sim g(n) \wedge g(n) \sim h(n) \Rightarrow f(n) \sim h(n)$
- Reflexivity

1. $f(n) = \Theta(f(n))$
2. $f(n) = O(f(n))$
3. $f(n) = \Omega(f(n))$
4. $f(n) \sim f(n)$

- Symmetry

1. $f(n) = \Theta(g(n)) \iff g(n) = \Theta(f(n))$
2. $f(n) \sim g(n) \iff g(n) \sim f(n)$

- Transpose symmetry

1. $f(n) = O(g(n)) \iff g(n) = \Omega(f(n))$
2. $f(n) = o(g(n)) \iff g(n) = \omega(f(n))$

- Little to big

1. $f(n) = o(g(n)) \Rightarrow f(n) = O(g(n))$
2. $f(n) = \omega(g(n)) \Rightarrow f(n) = \Omega(g(n))$

- Use in expressions

  The equal sign here says that we can always replace the left hand side by the right hand sign in an expression.

1. $n^m = O(n^{m'})$, when $m \leq m'$
2. $O(f(n)) + O(g(n)) = O(f(n) + g(n))$
3. $f(n) = O(f(n))$
4. $c \cdot O(f(n)) = O(f(n))$, if c is constant.
5. $O(O(f(n))) = O(f(n))$
6. $O(f(n))O(g(n)) = O(f(n)g(n))$