Logarithms

A refresher

What are logarithms? And why do we care?

- Logarithms are incredibly important in the study of algorithms—you really do need to be very familiar with them and their properties.

First, some discussion of notation

- In the ISO standard (and, in particular, in China):
  - $\lg(x)$ is the same as $\log_{10}(x)$
- Here in the US, we don't believe in international standards, so:
  - $\lg(x)$ is the same as $\log_2(x)$
- Fortunately, everywhere:
  - $\ln(x)$ is the same as $\log_e(x)$
- The shape of the logarithm function is always the same. The base merely changes scale, i.e. a different constant.

What is a logarithm?

- The (natural) log of *x* is the integral of the reciprocal of *x*, with respect to *x* and integrated over all possible positive values of *x*:
  - $\ln(x) = \int 1/x \, dx$
- Keep in mind the series:
  - $x^2 = 1/2 \int x \, dx,$ i.e. $dx^2/dx = 2x$
  - $x = \int 1 \, dx,$ i.e. $dx/dx = 1$
  - $\ln(x) = \int 1/x \, dx,$ i.e. $d \ln(x)/dx = x^{-1}$
  - $1/x = - \int 1/x^2 \, dx,$ i.e. $dx^{-1}/dx = - x^{-2}$
- Clearly, the formula $dx^k/dx = k \, x^{k-1}$ breaks down when $k = 0$

Useful identities

- $b^{log_b(x)} = x$
- $log_b\ xy = log_b\ x + log_b\ y$
- $log_b\ x^p = p\ log_b\ x$
- $log_b\ x = log_k\ x\ /\ log_k\ b$

Application to algorithms

- By far the most common and most effective way of speeding up an algorithm is to replace a ***O****(N)* growth rate with an ***O****(log N)* growth rate.

- *N* is the integral of *dx* over *x*=0..*N*
    - That's to say that it's the total cost when every element contributes equally to the cost.

- l*og N* is the integral of *1/x dx* over *x*=0…*N*
    - *That's to say that it's the total cost when every element contributes an increasingly smaller cost to the total.*

How do we achieve this?

- Normally, we achieve a reduction from *N* to *log N* by introducing an extra degree of freedom to our data structure:
  - instead of a linear structure, we form a tree where, at each node of the tree, we can make a choice based on a comparison.
  - To navigate such a structure (searching for an element), we take (max) $1 + \log_k N$ steps instead of (max) *N* steps, assuming that at each node/comparison, we have *k* choices.

One further identity

- $\int \ln x \, dx = x \ln x - x$
  - So, you can see how we might get polynomials which include both N and $N \log N$ terms

Conversion factors

- Given a number *x*, the numerical value of *ln x* will be **smaller** than the numerical value of *lg x* by a factor of 1.44 (i.e. *lg e*)
- Given a number *x*, the numerical value of *lg x* will be **larger** than the numerical value of *ln x* by a factor of 1.39 (i.e. *ln 2*)