CSE373: Data Structures and Algorithms

Math Review

Steve Tanimoto
Autumn 2016

This lecture material represents the work of multiple instructors at the University of Washington.
Thank you to all who have contributed!

Today

- Review of math essential to algorithm analysis
  - Motivating example: binary vs linear search
  - Logarithms and exponents
  - Floor and ceiling functions
  - Numbers of orderings
  - Numbers of combinations
  - Arithmetic series
  - Geometric series
  - Squared harmonic series
- Begin algorithm analysis

Motivating Example: Binary Search vs Linear Search

- Given a sorted list of n items, how long does it take (in the worst case) to determine whether some value is in the list?

17	18	21	23	24	28	31	44	52	56	60	69	70	72	90	95

- Binary search: ??
- Linear search: ??
- Which is faster?

Powers of 2

- Most modern computers use hardware that implements base-2 arithmetic.
- Any fixed-precision integer is implemented as a sequence of bits (often of length 8, 16, or 32). The value is given by

$$\sum_{i=0}^{i=n-1} a_i 2^i$$

Here n is the number of bits. The least significant bit is bit 0.

- Bit 0 has value 1; bit 1 has value 2; bit 2 has value 4; ...; bit n-1 has value 2^{n-1}
- With n bits, there are 2^n possible values. Call this number p. Then $n = \log_2 p$

Logarithms and Exponents

- Definition: $\log_2 x = y$ if and only if $x = 2^y$
  - $\log_2 8 = 3$, because $8 = 2^3$.
  - $\log_2 65536 = 16$ because $65536 = 2^{16}$.
- The exponent of after a number says how many times to use the number in a multiplication.
  e.g. $2^3 = 2 \times 2 \times 2 = 8$

  (2 is used 3 times in a multiplication to get 8)

Logarithms and Exponents (cont.)

- A logarithm tells how many of one number (the base of the logarithm) to multiply to get another number. It asks "what exponent produced this?"
  e.g. $\log_2 8 = 3$

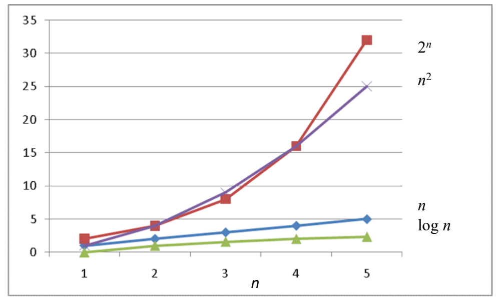
(2 makes 8 when used 3 times in a multiplication)

Logarithms and Exponents

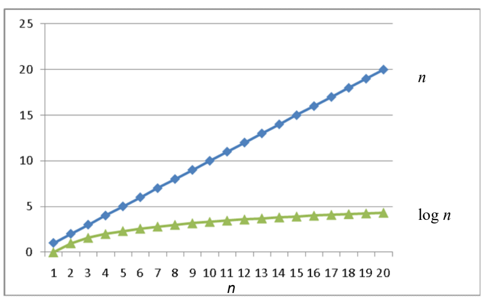
- Since so much is binary in CSE, $\log$ almost always means $\log_2$
- $\log_2 n$ tells you how many bits needed to identify one element from a set of n elements.
- So, $\log_2 1{,}000{,}000$ = "a little under 20"

- Logarithmic functions and exponential functions are inverses of one another. Just as exponential functions grow *very* quickly, logarithmic functions grow *very* slowly.

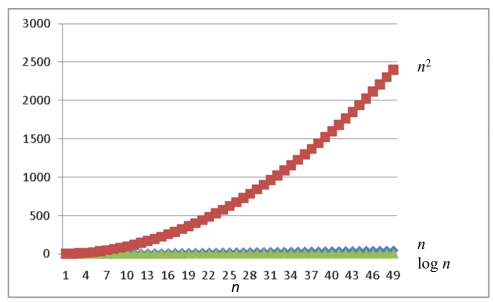
Logarithms and Exponents



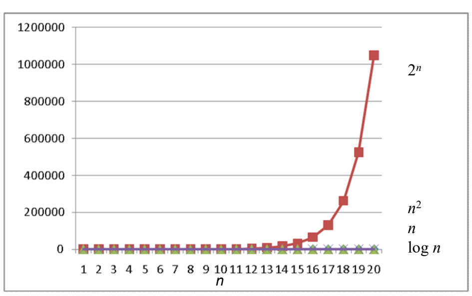
Logarithms and Exponents



Logarithms and Exponents



Logarithms and Exponents



Properties of logarithms

- $\log(x \cdot y) = \log x + \log y$

- $\log(n^k) = k \log n$

- $\log(x/y) = \log x - \log y$

- $\log(\log x)$ is written $\log_y \log x$
  - Grows as slowly as 2^{2^y} grows quickly

- $(\log x)(\log x)$ is written $\log^2 x$
  - It is greater than $\log x$ for all $x > 2$
  - It is not the same as $\log \log x$

Log base often doesn't matter much!

"Any base *B* logarithm is equivalent to a base 2 logarithm within a constant factor"

- And we are about to stop worrying about constant factors!
- In particular, $\log_2 x = c \log_{10} x$ where $c \approx 3.22$
- In general we can convert log bases via a constant multiplier
- To convert from base *A* to base *B*:

$$\log_B x = (\log_A x) / (\log_A B)$$

A Combinatorial Problem

Airplane boarding is not always smooth. Some folks want to board first; others don't care. Some need the aisles clear, others don't.

After some complaints at Amazing Airlines, one of the tech folks decides to implement an algorithm that will generate all possible boarding orders and evaluate each one in terms of how fast and how happy the passengers will be.

A Combinatorial Problem (cont.)

It will take 1 second to evaluate each possible ordering. Since the airline doesn't know until 15 min. before boarding who is actually in the gate area, they can't run the program until 15 min. before boarding.

Will this method find the best ordering in time? (For what values of n is it feasible?)

A Combinatorial Problem (cont.)

It will take 1 second to evaluate each possible ordering. Since the airline doesn't know until 15 min. before boarding who is actually in the gate area, they can't run the program until 15 min. before boarding.

Will this method find the best ordering in time? (For what values of n is it feasible?)

Now they optimize the code and get a machine that is much faster and it only takes 1 microsecond to evaluate an ordering. For what values of n is it now feasible?

Permutations

- The number of possible orderings of n distinct items.
- Ex: n = 3. {(a,b,c), (a,c,b), (b,a,c),(b,c,a),(c,a,b),(c,b,a)}
- The number of permutations of n items is n factorial (n!).

n	n!
1	1
2	2
3	6
4	24
5	120
6	720
7	5040

n	n!
12	479,001,600
13	6,227,020,800

Ice Cream Choices -- Combinations

Kombi-Krazy Ice Cream carries 100 flavors of ice cream.

Normally, you get *k* scoops in a bowl for *k* dollars, e.g., 3 scoops for $3.00.

The special promotion is half off if your *k* flavors are all different.

Now they want to brag about how many different combinations they offer.

How many distinct combinations are possible for *k* = 1 through 5?

Ice Cream Choices -- Combinations (cont.)

k	flavor 1 choices, flavor 2 choices...	repeats,	net.
1	**100**		**100**
2	100 99		
3	100 99 98		
4	100 99 98 97		
5	100 99 98 97 96		

Ice Cream Choices -- Combinations (cont.)

k	flavor 1 choices, flavor 2 choices...	repeats,	net.
1	100		100
2	100 99	2	9900/2 = **4950**
3	100 99 98		
4	100 99 98 97		
5	100 99 98 97 96		

Ice Cream Choices -- Combinations (cont.)

k	flavor 1 choices, flavor 2 choices...	repeats,	net.
1	100		100
2	100 99	2	9900/2 = 4950
3	100 99 98	6	970200/6 = 161700
4	100 99 98 97	24	94109400/24 = 3921225
5	100 99 98 97 96	120	9034502400/120 = **75,287,520**

Combinations

- Choosing *k* items from *n* possibilities:
- "*n* choose *k*"
- $C(n, k) = \binom{n}{k} = n! / ((n-k)!\, k!)$
- n things chosen k at a time
- Binomial coefficient C^{n}_{k}
- Example: Number of possible partnerships in CSE 373 assuming 175 students:
  C(175, 2) = 175! / 173! 2! = 15225

Converting Fractions to Integers

- When analyzing algorithms, we will often need to divide an integer by 2 and somehow end up with an integer, or take a log and end up with an integer.
- For example, "I'm thinking of a number between 1 and 100. How many yes-no questions would you have to ask, in the worst case (but playing optimally) to discover my number?"
- Note:

$$\log_2 100 \approx 6.64385619$$

Ans: the smallest integer greater than or equal to $\log_2 100$.

$$= \lceil \log_2 100 \rceil = 7$$

Floor and ceiling

$\lfloor X \rfloor$ Floor function: the largest integer ≤ X

$\lfloor 2.7 \rfloor = 2 \qquad \lfloor -2.7 \rfloor = -3 \qquad \lfloor 2 \rfloor = 2$

$\lceil X \rceil$ Ceiling function: the smallest integer ≥ X

$\lceil 2.3 \rceil = 3 \qquad \lceil -2.3 \rceil = -2 \qquad \lceil 2 \rceil = 2$

Facts about floor and ceiling

1. $X - 1 < \lfloor X \rfloor \leq X$
2. $X \leq \lceil X \rceil < X + 1$
3. $\lfloor n/2 \rfloor + \lceil n/2 \rceil = n$ if n is an integer

Sequences and Series

- A *sequence* is an ordered collection of numbers, possibly but not necessarily repeating numbers.
- Example: 13, 2, 14, 1. (this is a finite sequence).

  0, 2, 4, 6, . . . (this is an infinite sequence).

  ellipsis

- A *series* is the sum of a sequence
- Example: 13 + 2 + 14 + 1 = 30

  0 + 2 + 4 + 6 + ... = ∞

Arithmetic Sequences and Series

- In general for a sequence we have
- $n_0, n_1, n_2, \ldots, n_i$ (finite case)
- or $n_0, n_1, n_2, \ldots, n_i, \ldots$ (infinite case)

- The sequence is an arithmetic sequence if
- $n_i = n_0 + i\,d$ for some real number d.
- Example where $n_0 = 10$ and $d = -2$:

  10, 8, 6, 4

  An arithmetic series is the sum of an arithmetic sequence.

  E.g., 10 + 8 + 6 + 4 = 28.

Arithmetic Series Formula

- A quick way to evaluate an arithmetic series is to multiply the number of terms by the average of the first and last terms:

- 10 + 8 + 6 + 4 = 4 (10 + 4)/2 = 4 (7) = 28.

$$\sum_{i=0}^{k-1} n_i = k\,(n_0 + n_{k-1})/2$$

Geometric Sequences and Series

- A geometric sequence is one in which we get the next element by multiplying by a constant.
- Example. 1, 3, 9, 27, 81. (a finite example)

  1, 3, 9, 27, 81, ... (an infinite example)

- A geometric series is the sum of a geometric sequence.
- Example. 1 + 3 + 9 + 27 + 81 = 121

  1 + 3 + 9 + 27 + 81 + ... = ∞

Zeno's Dichotomy Paradox

- Homer wants to run from A to B.
- Before he goes all the way from A to B, he must go half way to B.
- But before he can go from the midpoint to B, he must go half of the remaining distance, etc.
- So he must complete an infinite number of tasks, which Zeno said is impossible.

Convergent Geometric Series

- If the (absolute value of the) multiplicative constant is less than 1, the series for an infinite geometric sequence converges (i.e., the sum is finite).

- Example. 1 + 2 + 4 + 8 + 16 + ... = ∞
  1 + 1/2 + 1/4 + 1/8 + 1/16 + ... = 2

- $$a + ar + ar^2 + ar^3 + \cdots + ar^{n-1} = \sum_{k=0}^{n-1} ar^k = a\,\frac{1-r^n}{1-r},$$ (finite case)

  $$a + ar + ar^2 + ar^3 + ar^4 + \cdots = \sum_{k=0}^{\infty} ar^k = \frac{a}{1-r}, \text{ for } |r| < 1.$$ (infinite case)

Simple Example

- Suppose we have an array of 32 elements.
- We have a recursive procedure that processes half of the unprocessed elements and calls itself recursively on the other half.
- Assume that if there is just one unprocessed element, it does process that element and returns.
- How many elements does it process?
- 16 + 8 + 4 + 2 + 1 + 1 = 32. (It processes all of the elements).

- Note that $\sum_{i=0}^{4} 16 \cdot (1/2^i)$ =
  = $16 \cdot (1 - 1/2^5)/(1 - ½)$ = $32 \cdot (31/32) = 31$
  ≈ 16 / (1 - 1/2) = 32.

The Harmonic Series

- The (infinite) harmonic series is the following. Its value is infinity (i.e., it diverges).

$$\sum_{n=1}^{\infty} \frac{1}{n} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \cdots.$$

The Squared Harmonic Series

- The squared harmonic series is the following. Its value is finite (i.e., it converges).

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = 1 + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \cdots$$