CSE373: Data Structures and Algorithms

Lecture 2: Proof by Induction & Algorithm Analysis

Lauren Milne

Summer 2015

Today

- Did everyone get email sent on Monday about TA Sections starting on Thursday?
- Homework 1 due 10:59pm next Wednesday, July 1st.
- Review math essential to algorithm analysis
  - Proof by induction
  - Exponents and logarithms
  - Floor and ceiling functions
- Begin algorithm analysis

Homework 1

- Download Eclipse and Java
- Implement Stack ADT for Java *double*
  - Array
    - ***ArrayStack:*** push(double n), pop(), peek(), constructor
    - Starts small (approx. 10 elements) and doubles in size when full, copy elements over
    - Throw exception if stack is empty for pop() and peek()
  - List
    - ***ListStack*****:** push(double n), pop(), peek(), constructor
    - Inner *ListStackNode class*
    - Throw exception if stack is empty for pop() and peek()
- Test
  - *Reverse.java* and *Dstack.java*
  - Using .dat sound files (created using .wav files through sox), and .dat files you create manually (edge cases)
- Write up: README.txt
  - ***QueueStack*** – push() and pop()

Background on Induction

- Type of mathematical proof
- Typically used to establish a given statement for all natural numbers (integers > 0)
- Proof is a sequence of deductive steps
    1. Show the statement is true for the first number.
    2. Show that if the statement is true for any one number, this implies the statement is true for the next number.
    3. If so, we can infer that the statement is true for all numbers.

Think about climbing a ladder

1. Show you can get to the first rung (base case)

2. Show you can get between rungs (inductive step)

3. Now you can climb forever.

Why you should care

- Induction turns out to be a useful technique
  - AVL trees
  - Heaps
  - Graph algorithms
  - Can also prove things like $3^n > n^3$ *for* $n \geq 4$
- Exposure to rigorous thinking

Example problem

- Find the sum of the integers from 1 to *n*
- 1 + 2 + 3 + 4 + … + (n-1) + n

$$\sum_{i=1}^{n} i$$

- For any $n \geq 1$
- Could use brute force, but would be slow
- There's probably a clever shortcut

Finding the formula

- Shortcut will be some formula involving n
- Compare examples and look for patterns
  - Not something I will ask you to do!
- Start with n = 10:

  1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10
  - Large enough to be a pain to add up
  - Worthwhile to find shortcut

Finding the formula

$$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10$$

$$= 5 \times 11$$

Finding the formula

$$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9$$

$$= 4\times 10 + 5$$

Finding the formula

$$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8$$

$$= 4\times 9$$

Finding the formula

$$1 + 2 + 3 + 4 + 5 + 6 + 7$$

$$= 3 \times 8 + 4$$

Finding the formula

$n=7$	$3\times8 + 4$
$n=8$	4×9
$n=9$	$4\times10 + 5$
$n=10$	5×11

Finding the formula

n=7	3×8 + 4	n is odd
n=8	4×9	n is even
n=9	4×10 + 5	n is odd
n=10	5×11	n is even

Finding the formula

When n is even

$$1 + 2 + 3 + 4 + \dots + (n-3) + (n-2) + (n-1) + n$$

$$= (n/2)\times(n+1)$$

Finding the formula

$3\times8 + 4$	
4×9	$n(n+1)/2$
$4\times10 + 5$	
5×11	$n(n+1)/2$

Finding the formula

When n is odd

$$1 + 2 + 3 + 4 + \ldots + (n-3) + (n-2) + (n-1) + n$$

$$= ((n-1)/2)\times(n+1) + (n+1)/2$$

Finding the formula

When n is odd

$$1 + 2 + 3 + 4 + \ldots + (n-3) + (n-2) + (n-1) + n$$

$$= ((n-1)/2)\times(n+1) + (n+1)/2$$

Finding the formula

When n is odd

$$1 + 2 + 3 + 4 + \ldots + (n-3) + (n-2) + (n-1) + n$$

$$= ((n-1)\times(n+1) + (n+1))/2$$

Finding the formula

When n is odd

$$1 + 2 + 3 + 4 + \ldots + (n-3) + (n-2) + (n-1) + n$$

$$= ((n-1)\times(n+1) + (n+1))/2$$

Finding the formula

When n is odd

$$1 + 2 + 3 + 4 + \ldots + (n-3) + (n-2) + (n-1) + n$$

$$= ((n-1 + 1)\times(n+1))/2$$

Finding the formula

When n is odd

$$1 + 2 + 3 + 4 + \ldots + (n-3) + (n-2) + (n-1) + n$$

$$= ((n-1+1)\times(n+1))/2$$

Finding the formula

When n is odd

$$1 + 2 + 3 + 4 + \ldots + (n-3) + (n-2) + (n-1) + n$$

$$= (n\,(n+1))/2$$

Finding the formula

$3 \times 8 + 4$	$n(n+1)/2$
4×9	$n(n+1)/2$
$4 \times 10 + 5$	$n(n+1)/2$
5×11	$n(n+1)/2$

Are we done?

- The pattern seems pretty clear
  - Is there any reason to think it changes?
- But we want something for any $n \geq 1$
- A mathematical approach is skeptical

$$\frac{n(n+1)}{2}$$

Are we done?

- The pattern seems pretty clear
  - Is there any reason to think it changes?
- But we want something for *any* $n \geq 1$
- A mathematical approach is *skeptical*
- All we know is $n(n+1)/2$ works for 7 to 10
- We must *prove* the formula works in all cases

Proof by induction

- Prove the formula works for all cases.
- Induction proofs have four components:

1. Relationship that you want to prove, e.g., *sum of integers from 1 to n = n(n+1)/2*
2. The base case (usually "let n = 1"),
3. The assumption step ("assume true for n = k")
4. The induction step ("now let n = k + 1").

n and *k* are just *variables*!

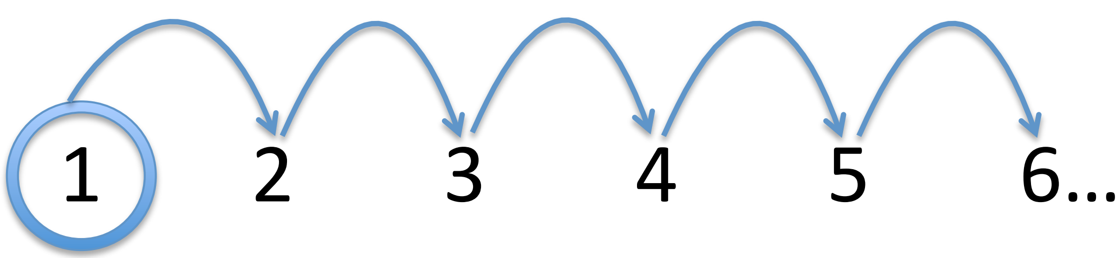
Proof by induction

- P(n) = sum of integers from 1 to n
- We need to do
  - Base case *prove for P(1)*
  - Assumption *assume for P(k)*
  - Induction step *show for P(k+1)*

- *n* and *k* are just *variables*!

Proof by induction

- P(n) = sum of integers from 1 to n
- We need to do
  - Base case *prove for P(1)*
  - Assumption *assume for P(k)*
  - Induction step *show for P(k+1)*



Proof by induction

- What we are trying to prove: *P(n) = n(n+1)/2*
- Base case
  - P(1) = 1
  - 1(1+1)/2 = 1(2)/2 = 1(1) = 1 ✓

Proof by induction

- What we are trying to prove: $P(n) = n(n+1)/2$
- Assume true for k: $P(k) = k(k+1)/2$
- Induction step:
  - Now consider $P(k+1)$

  $= 1 + 2 + \ldots + k + (k+1)$

Proof by induction

- What we are trying to prove: $P(n) = n(n+1)/2$
- Assume true for k: $P(k) = k(k+1)/2$
- Induction step:
  - Now consider $P(k+1)$

  $= 1 + 2 + \dots + k + (k+1)$

  $= k(k+1)/2 + (k+1)$

Proof by induction

- What we are trying to prove: $P(n) = n(n+1)/2$
- Assume true for k: $P(k) = k(k+1)/2$
- Induction step:
  - Now consider $P(k+1)$

  $= 1 + 2 + \ldots + k + (k+1)$

  $= k(k+1)/2 + (k+1)$

  $= k(k+1)/2 + 2(k+1)/2$

Proof by induction

- What we are trying to prove: $P(n) = n(n+1)/2$
- Assume true for k: $P(k) = k(k+1)/2$
- Induction step:
  – Now consider $P(k+1)$

  $= 1 + 2 + \ldots + k + (k+1)$

  $= k(k+1)/2 + (k+1)$

  $= k(k+1)/2 + 2(k+1)/2 = (k(k+1) + 2(k+1))/2$

Proof by induction

- What we are trying to prove: $P(n) = n(n+1)/2$
- Assume true for k: $P(k) = k(k+1)/2$
- Induction step:
  - Now consider $P(k+1)$

$= 1 + 2 + \ldots + k + (k+1)$

$= k(k+1)/2 + (k+1)$

$= k(k+1)/2 + 2(k+1)/2 \;\; = (k(k+1) + 2(k+1))/2$

$= (k+1)(k+2)/2$

Proof by induction

- What we are trying to prove: $P(n) = n(n+1)/2$
- Assume true for k: $P(k) = k(k+1)/2$
- Induction step:
  - Now consider $P(k+1)$

$= 1 + 2 + \ldots + k + (k+1)$

$= k(k+1)/2 + (k+1)$

$= k(k+1)/2 + 2(k+1)/2 = (k(k+1) + 2(k+1))/2$

$= (k+1)(k+2)/2$

Proof by induction

- What we are trying to prove: $P(n) = n(n+1)/2$
- Assume true for k: $P(k) = k(k+1)/2$
- Induction step:
  - Now consider $P(k+1)$

$= 1 + 2 + \ldots + k + (k+1)$

$= k(k+1)/2 + (k+1)$

$= k(k+1)/2 + 2(k+1)/2 \quad = (k(k+1) + 2(k+1))/2$

$= (k+1)(k+2)/2 \quad = (k+1)((k+1)+1)/2$

Proof by induction

- What we are trying to prove: $P(n) = n(n+1)/2$
- Assume true for k: $P(k) = k(k+1)/2$
- Induction step:
  - Now consider $P(k+1)$

$= 1 + 2 + \ldots + k + (k+1)$

$= k(k+1)/2 + (k+1)$

$= k(k+1)/2 + 2(k+1)/2 \;= (k(k+1) + 2(k+1))/2$

$= (k+1)(k+2)/2 \;= (k+1)((k+1)+1)/2$

We're done!

- P(n) = sum of integers from 1 to n
- We have shown
  - Base case — *proved for P(1)*
  - Assumption — *assumed for P(k)*
  - Induction step — *proved for P(k+1)*

Success: we have proved that P(n) is true for any $n \geq 1$ ☺

Another one to try

- What is the sum of the first *n* powers of 2?
- $2^0 + 2^1 + 2^2 + \ldots + 2^n$
- $n = 0$: $2^0 = 1$
- $n = 1$: $2^0 + 2^1 = 1 + 2 = 3$
- $n = 2$: $2^0 + 2^1 + 2^2 = 1 + 2 + 4 = 7$
- $n = 3$: $2^0 + 2^1 + 2^2 + 2^3 = 1 + 2 + 4 + 8 = 15$

- For general n, the sum is $2^{n+1} - 1$

How to prove it

P(n) = "the sum of the first *n* powers of 2 is 2^{n+1}-1"

Theorem: *P(n)* holds for all $n \geq 0$

Proof: By induction on *n*

- Base case: *n*=0. Sum of first power of 2 is 2^0 , which equals 1 = 2^1 - 1.
- Inductive case:
  - Assume the sum of the first *k* powers of 2 is 2^{k+1}-1
  - Show the sum of the first (*k*+1) powers of 2 is 2^{k+2}-1

How to prove it

- The sum of the first k+1 powers of 2 is

$$\underbrace{2^0 + 2^1 + 2^2 + \ldots + 2^{(k-1)} + 2^k}_{} + 2^{k+1}$$

$$= \quad 2^{k+1} - 1 \qquad\qquad + 2^{k+1}$$

$$= \quad 2(2^{k+1}) - 1$$

$$= \quad 2^{k+2} - 1$$

Conclusion

- Mathematical induction is a technique for proving something is true for all integers starting from a small one, usually 0 or 1.
- A proof consists of three parts:
  1. Prove it for the base case.
  2. Assume it for some integer k.
  3. With that assumption, show it holds for k+1
- It can be used for complexity and correctness analyses.

End of Inductive Proofs!

Powers of 2

- A bit is 0 or 1 (just two different “letters” or “symbols”)
- A sequence of *n* bits can represent 2^n distinct things
  - For example, the numbers 0 through 2^n-1
- 2^{10} is 1024 (“about a thousand”, kilo in CSE speak)
- 2^{20} is “about a million”, mega in CSE speak
- 2^{30} is “about a billion”, giga in CSE speak

Java: an **`int`** is 32 bits and signed, so what is “max int”?

Powers of 2

- A bit is 0 or 1 (just two different "letters" or "symbols")
- A sequence of *n* bits can represent 2^n distinct things
  - For example, the numbers 0 through 2^n-1
- 2^{10} is 1024 ("about a thousand", kilo in CSE speak)
- 2^{20} is "about a million", mega in CSE speak
- 2^{30} is "about a billion", giga in CSE speak

Java: a **`long`** is 64 bits and signed, so what is max long?

$2^{63}-1$

Therefore…

Could give a unique id to…

- Every person in the U.S. with 29 bits
- Every person in the world with 33 bits
- Every person to have ever lived with 38 bits (estimate)
- Every atom in the universe with 250-300 bits

So if a password is 128 bits long and randomly generated,
do you think you could guess it?

Logarithms and Exponents

- Definition: **$x = 2^y$ if $\log_2 x = y$**
  - $8 = 2^3$, so **$\log_2$** $8 = 3$
  - $65536 = 2^{16}$, so **$\log_2$** $65536 = 16$

- The exponent of 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)

- A logarithm says how many of one number 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

- Definition: $\mathbf{x = 2^y}$ **if** $\mathbf{\log_2 x = y}$
  - $8 = 2^3$, so $\mathbf{\log_2} 8 = 3$
  - $65536 = 2^{16}$, so $\mathbf{\log_2} 65536 = 16$

- Since so much is binary in CS, **log** almost always means $\mathbf{\log_2}$

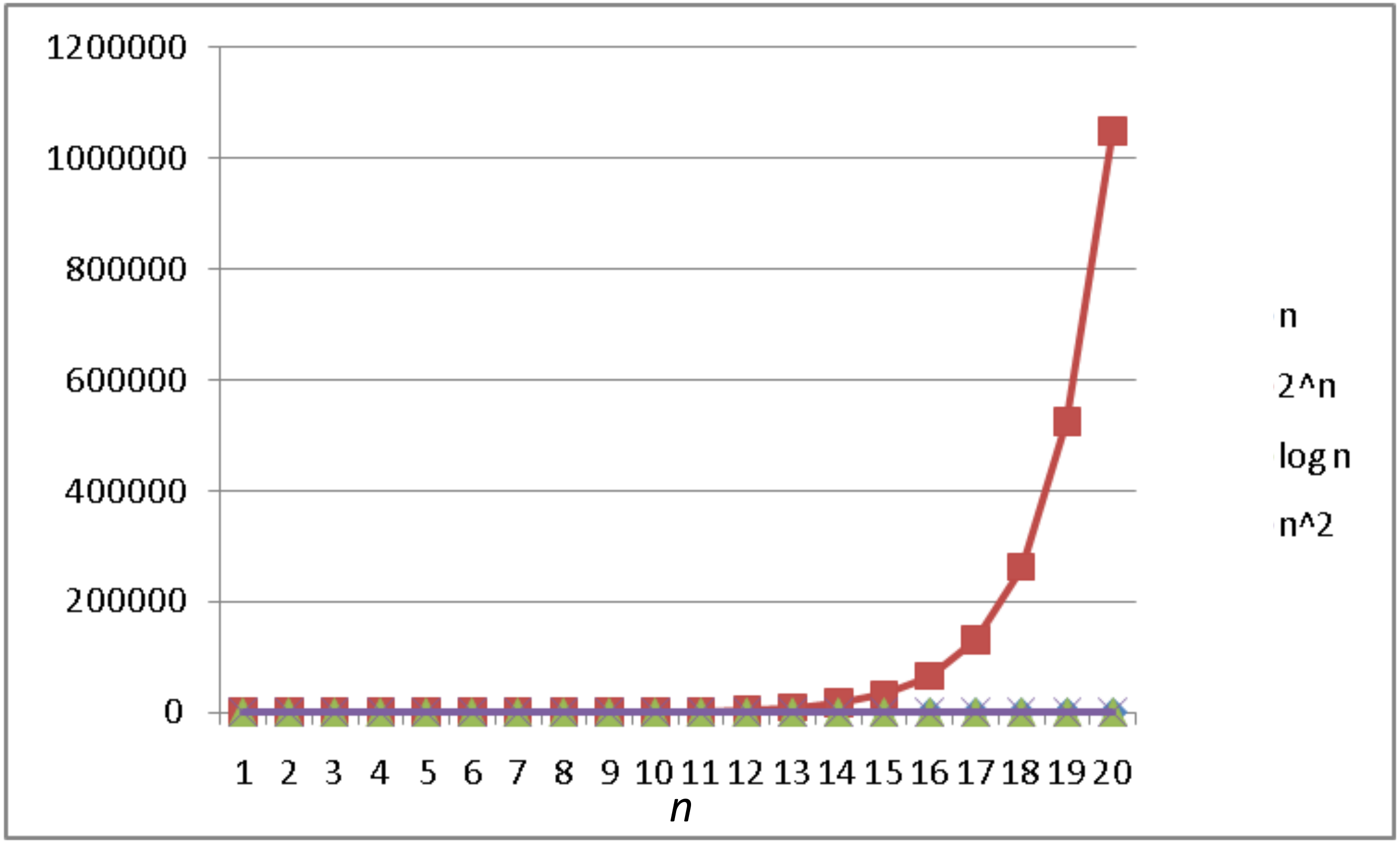
- $\mathbf{\log_2} n$ tells you how many bits needed to represent n combinations.

- So, $\mathbf{\log_2} 1{,}000{,}000$ = "a little under 20"

- Logarithms and exponents are inverse functions. Just as exponents grow *very* quickly, logarithms grow *very* slowly.

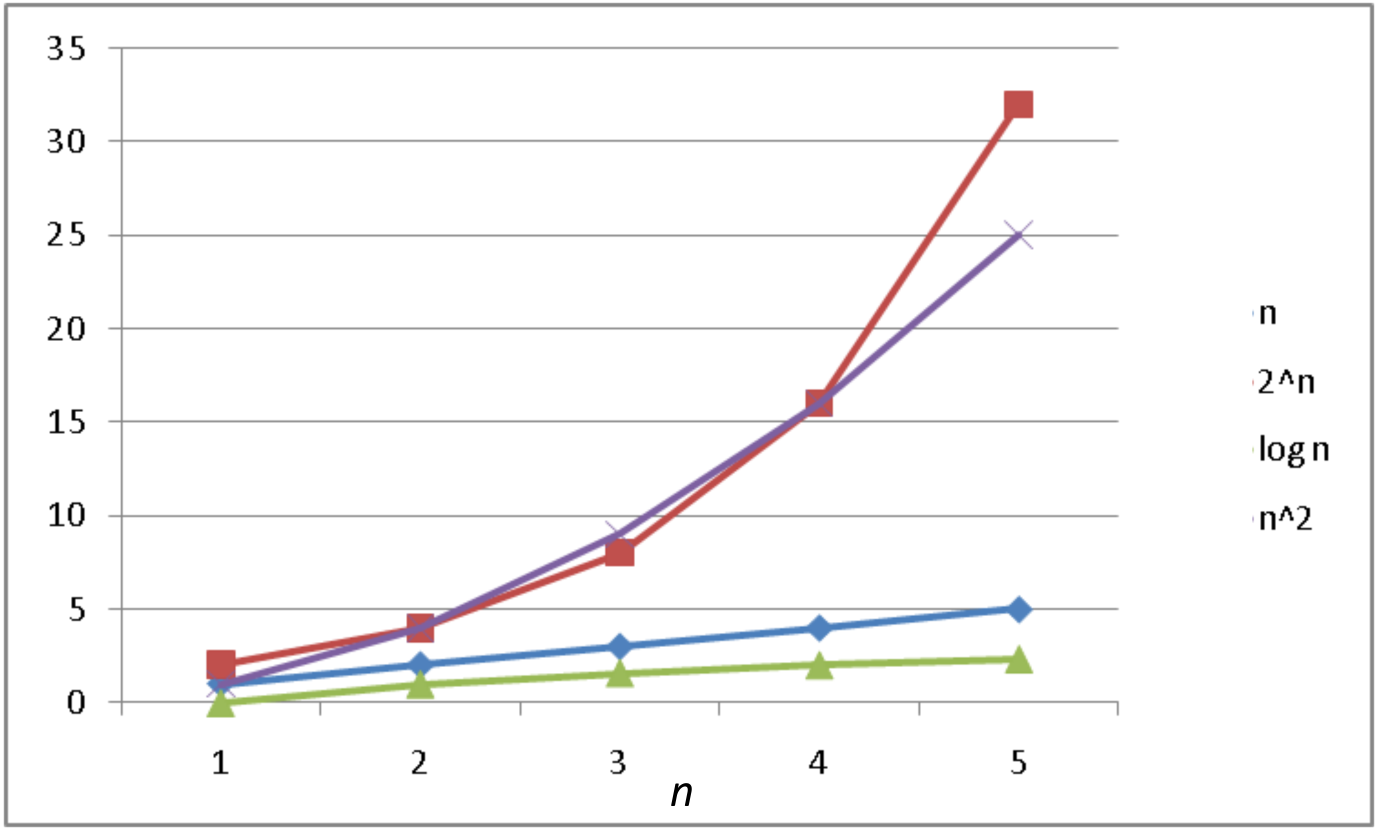
Logarithms and Exponents

See Excel file for plot data – play with it!



Logarithms and Exponents

See Excel file
for plot data –
play with it!



Logarithms and Exponents

See Excel file
for plot data –
play with it!