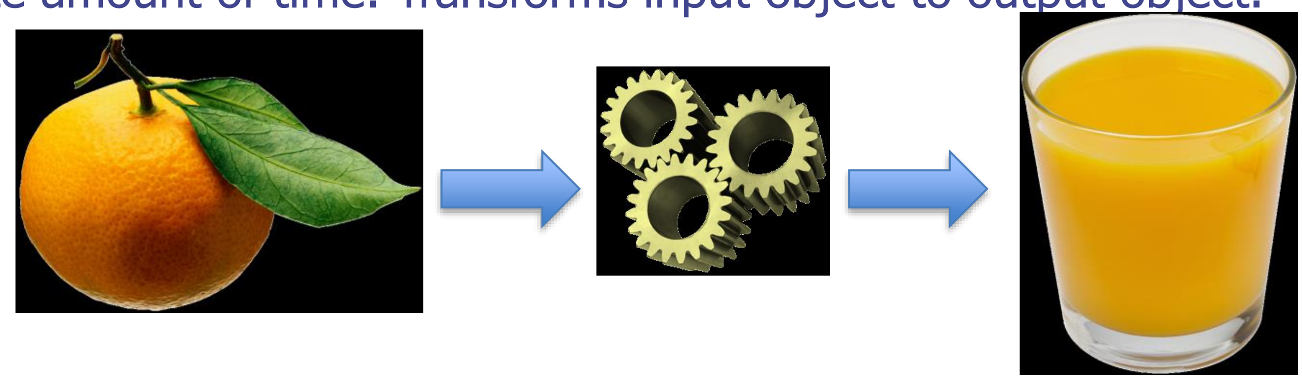
Lecture 2
Overview of concepts

CS 161 Design and Analysis of Algorithms

Ioannis Panageas

Design and Analysis of *Algorithms*

- This is a theoretical/of mathematical nature class. Ideas the primarily focus, not implementation.
- An **algorithm** is a step-by-step procedure for performing some task in a finite amount of time. Transforms input object to output object.



Input **Algorithm** **Output**

Design and Analysis of *Algorithms*

- Design: Come up with a procedure.
- Analysis: Running time.

Design and Analysis of *Algorithms*

- Design: Come up with a procedure.
- Analysis: Running time.

Running time is denoted by $T(n)$

- Number of "operations" for algorithm to terminate.
- We actually care about how it scales with **input size n**.
- Main focus is on worst-case analysis (vs average case analysis or best case analysis).

Example on worst-case vs average/best case

Given different numbers $x_1, x_2, \ldots, x_n$, find the position of x (assume exists).

Example on worst-case vs average/best case

Given different numbers $x_1, x_2, \ldots, x_n$, find the position of x (assume exists).

Algorithm:

For $i = 1$ to n **do**
  If $x_i == x$ **then**
    Print i;
    break;

Example on worst-case vs average/best case

Given different numbers $x_1, x_2, \ldots, x_n$, find the position of x (assume exists).

Algorithm:

For $i = 1$ to n **do**
  If $x_i == x$ **then**
    Print i;
    break;

Best case: 1 iterate

Example on worst-case vs average/best case

Given different numbers $x_1, x_2, \ldots, x_n$, find the position of x (assume exists).

Algorithm:

For $i = 1$ to n **do**
  If $x_i == x$ **then**
    Print i;
    break;

Best case: 1 iterate

Worst case: n iterates

Example on worst-case vs average/best case

Given different numbers $x_1, x_2, \ldots, x_n$, find the position of x (assume exists).

Algorithm:

For $i = 1$ to n **do**
  If $x_i == x$ **then**
    Print i;
    break;

Best case: 1 iterate

Worst case: n iterates

Average case: Challenging

Example on worst-case vs average/best case

Given different numbers $x_1, x_2, \ldots, x_n$, find the position of x (assume exists).

Algorithm:

For $i = 1$ to n **do**
If $x_i == x$ **then**
Print i;
break;

Best case: 1 iterate

Worst case: n iterates

Average case: $\frac{n+1}{2}$ iterates

Solution: $\frac{1}{n}\sum_{i=1}^{n} i = \frac{n(n+1)}{2n} = \frac{n+1}{2}$ iterates.

Example on worst-case vs average/best case

Given different numbers $x_1, x_2, \dots, x_n$, find the position of x (assume exists).

Explanation:

- If the order is random, x will be in any position with probability $\frac{1}{n}$.

Example on worst-case vs average/best case

Given different numbers $x_1, x_2, \ldots, x_n$, find the position of x (assume exists).

Explanation:

- If the order is random, x will be in any position with probability $\frac{1}{n}$.
- If $x = x_i$, Algorithm will run for i steps.

Example on worst-case vs average/best case

Given different numbers $x_1, x_2, \ldots, x_n$, find the position of x (assume exists).

Explanation:

- If the order is random, x will be in any position with probability $\frac{1}{n}$.
- If $x = x_i$, Algorithm will run for i steps.

⇒ Average number of steps is $1 \cdot \frac{1}{n} + 2 \cdot \frac{1}{n} + \ldots + n \cdot \frac{1}{n} = \frac{1}{n} \sum_{i=1}^{n} i$.

Example on worst-case vs average/best case

Given different numbers $x_1, x_2, \ldots, x_n$, find the position of x (assume exists).

Explanation:

- If the order is random, x will be in any position with probability $\frac{1}{n}$.
- If $x = x_i$, Algorithm will run for i steps.

⇒ Average number of steps is $1 \cdot \frac{1}{n} + 2 \cdot \frac{1}{n} + \ldots + n \cdot \frac{1}{n} = \frac{1}{n} \sum_{i=1}^{n} i$.

Since $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$, we have $\frac{n(n+1)}{2n}$.

Recap on proofs

Exercise 1: Show that $1 + 2 + ... + n = \frac{n(n+1)}{2}$ using induction.

Recap on proofs

Exercise 1: Show that $1 + 2 + ... + n = \frac{n(n+1)}{2}$ using induction.

Skeleton of Induction (2 steps):

- We prove the base case, typically $n = 1$.
- Assuming the statement holds for n,

  we prove it for $n + 1$.

Recap on proofs

Exercise 1: Show that $1 + 2 + ... + n = \frac{n(n+1)}{2}$ using induction.

Skeleton of Induction (2 steps):

- We prove the base case, typically $n = 1$.
- Assuming the statement holds for n,

  we prove it for $n + 1$.

Solution:

Base case $n = 1$

$$1 = \frac{1 \cdot 2}{2}$$

Assume $1 + 2 + ... + n = \frac{n(n+1)}{2}$

Show $1 + 2 + ... + n + (n + 1) = \frac{(n+1)(n+2)}{2}$

Recap on proofs

by Induction hypothesis

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

Recap on proofs

by Induction hypothesis

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

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

Recap on proofs

by Induction hypothesis

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

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

Therefore $1 + 2 + \ldots + n + (n + 1) = \frac{(n+1)(n+2)}{2}$

Recap on proofs

Exercise 2: Show that $1^3 + 2^3 + \ldots + n^3 = \left(\frac{n(n+1)}{2}\right)^2$ using induction.
Solution:

Recap on proofs

Exercise 2: Show that $1^3 + 2^3 + ... + n^3 = \left(\frac{n(n+1)}{2}\right)^2$ using induction.

Solution:

Base case $n = 1$

$$1^3 = \left(\frac{1 \cdot 2}{2}\right)^2$$

Recap on proofs

Exercise 2: Show that $1^3 + 2^3 + ... + n^3 = \left(\frac{n(n+1)}{2}\right)^2$ using induction.

Solution:

Base case $n = 1$

$$1^3 = \left(\frac{1 \cdot 2}{2}\right)^2$$

Assuming that $\sum_{i=1}^{n} i^3 = \left(\frac{n(n+1)}{2}\right)^2$,

we need to prove $\sum_{i=1}^{n+1} i^3 = \left(\frac{(n+1)(n+2)}{2}\right)^2$.

Recap on proofs

Exercise 2: Show that $1^3 + 2^3 + \ldots + n^3 = \left(\frac{n(n+1)}{2}\right)^2$ using induction.

Solution:

Base case $n = 1$

$$1^3 = \left(\frac{1 \cdot 2}{2}\right)^2$$

Assuming that $\sum_{i=1}^{n} i^3 = \left(\frac{n(n+1)}{2}\right)^2$,

we need to prove $\sum_{i=1}^{n+1} i^3 = \left(\frac{(n+1)(n+2)}{2}\right)^2$.

$$\sum_{i=1}^{n+1} i^3 = \sum_{i=1}^{n} i^3 + (n+1)^3$$

Recap on proofs

Exercise 2: Show that $1^3 + 2^3 + \dots + n^3 = \left(\frac{n(n+1)}{2}\right)^2$ using induction.

Solution:

Base case $n = 1$

$$1^3 = \left(\frac{1 \cdot 2}{2}\right)^2$$

Assuming that $\sum_{i=1}^{n} i^3 = \left(\frac{n(n+1)}{2}\right)^2$,

we need to prove $\sum_{i=1}^{n+1} i^3 = \left(\frac{(n+1)(n+2)}{2}\right)^2$.

$$\sum_{i=1}^{n+1} i^3 = \sum_{i=1}^{n} i^3 + (n+1)^3$$

$$= \left(\frac{n(n+1)}{2}\right)^2 + (n+1)^3 \quad \text{by Induction hypothesis}$$

Recap on proofs

Exercise 2: Show that $1^3 + 2^3 + \ldots + n^3 = \left(\frac{n(n+1)}{2}\right)^2$ using induction.

Solution:

Base case $n = 1$

$$1^3 = \left(\frac{1 \cdot 2}{2}\right)^2$$

Assuming that $\sum_{i=1}^{n} i^3 = \left(\frac{n(n+1)}{2}\right)^2$,

we need to prove $\sum_{i=1}^{n+1} i^3 = \left(\frac{(n+1)(n+2)}{2}\right)^2$.

$$\sum_{i=1}^{n+1} i^3 = \sum_{i=1}^{n} i^3 + (n+1)^3$$

$$= \left(\frac{n(n+1)}{2}\right)^2 + (n+1)^3 \quad \text{by Induction hypothesis}$$

$$= (n+1)^2(n+1+\frac{n^2}{4})$$

Recap on proofs

Exercise 2: Show that $1^3 + 2^3 + \ldots + n^3 = \left(\frac{n(n+1)}{2}\right)^2$ using induction.

Solution:

Base case $n = 1$

$$1^3 = \left(\frac{1 \cdot 2}{2}\right)^2$$

Assuming that $\sum_{i=1}^{n} i^3 = \left(\frac{n(n+1)}{2}\right)^2$,

we need to prove $\sum_{i=1}^{n+1} i^3 = \left(\frac{(n+1)(n+2)}{2}\right)^2$.

$$\sum_{i=1}^{n+1} i^3 = \sum_{i=1}^{n} i^3 + (n+1)^3$$

$$= \left(\frac{n(n+1)}{2}\right)^2 + (n+1)^3 \quad \text{by Induction hypothesis}$$

$$= (n+1)^2\left(\frac{n^2+4n+4}{4}\right)$$

Recap on proofs

Exercise 2: Show that $1^3 + 2^3 + \ldots + n^3 = \left(\frac{n(n+1)}{2}\right)^2$ using induction.

Solution:

Base case $n = 1$

$$1^3 = \left(\frac{1 \cdot 2}{2}\right)^2$$

Assuming that $\sum_{i=1}^{n} i^3 = \left(\frac{n(n+1)}{2}\right)^2$,

we need to prove $\sum_{i=1}^{n+1} i^3 = \left(\frac{(n+1)(n+2)}{2}\right)^2$.

$$\sum_{i=1}^{n+1} i^3 = \sum_{i=1}^{n} i^3 + (n+1)^3$$

$$= \left(\frac{n(n+1)}{2}\right)^2 + (n+1)^3 \text{ by Induction hypothesis}$$

$$= (n+1)^2\left(\frac{n^2+4n+4}{4}\right) = \frac{(n+1)^2(n+2)^2}{4}$$

Recap on proofs

Exercise 2: Show that $1^3 + 2^3 + \ldots + n^3 = \left(\frac{n(n+1)}{2}\right)^2$ using induction.

Solution:

Base case $n = 1$

$$1^3 = \left(\frac{1 \cdot 2}{2}\right)^2$$

Assuming that $\sum_{i=1}^{n} i^3 = \left(\frac{n(n+1)}{2}\right)^2$,

we need to prove $\sum_{i=1}^{n+1} i^3 = \left(\frac{(n+1)(n+2)}{2}\right)^2$.

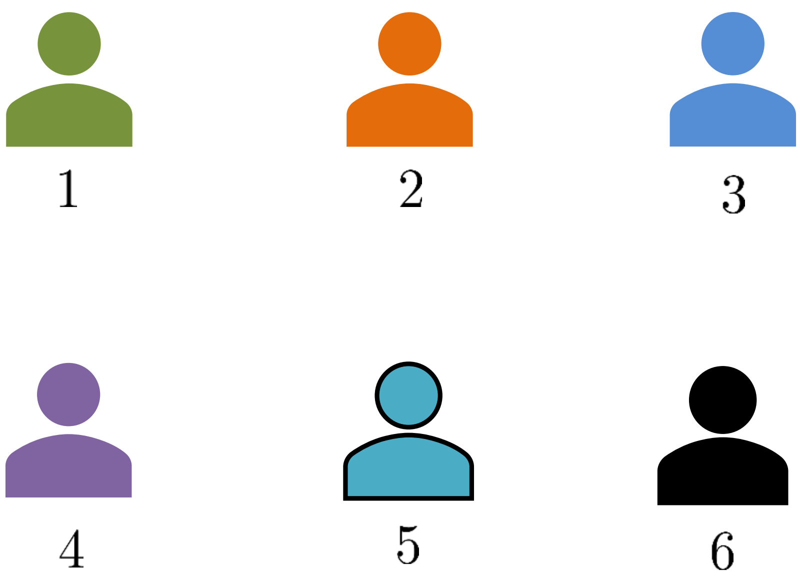
$$\sum_{i=1}^{n+1} i^3 = \sum_{i=1}^{n} i^3 + (n+1)^3$$

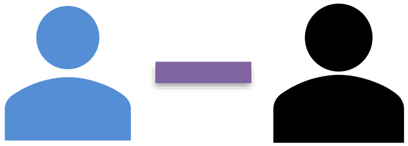
$$= \left(\frac{n(n+1)}{2}\right)^2 + (n+1)^3 \text{ by Induction hypothesis}$$

$$= (n+1)^2\left(\frac{n^2+4n+4}{4}\right) = \left(\frac{(n+1)(n+2)}{2}\right)^2 \quad \blacksquare$$

Recap on proofs

Exercise 3: We consider a group of 6 classmates. Each pair either has exchanged phone numbers or not. We need to show that there is a group of 3 classmates among them who have all shared their contact details with each other or not.



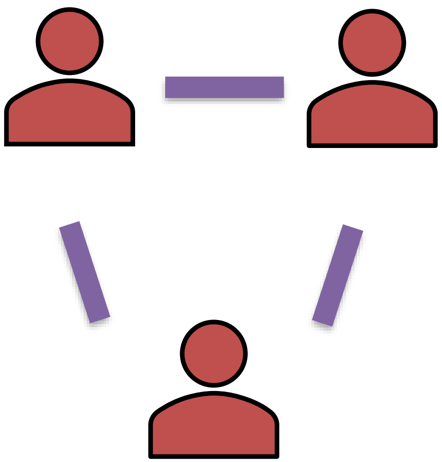


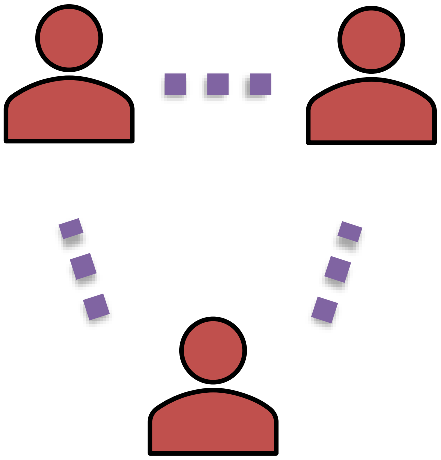
3 and 6 have exchanged phone numbers.

2 and 5 have not exchanged Phone numbers.

Recap on proofs

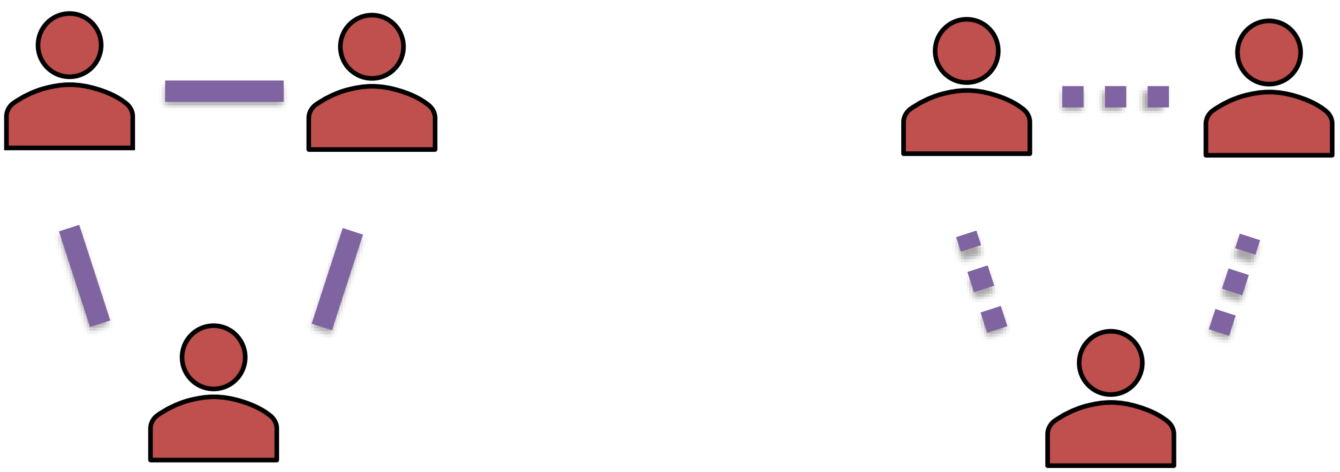
Need to show no matter the configuration, there are always 3 people so that





Recap on proofs

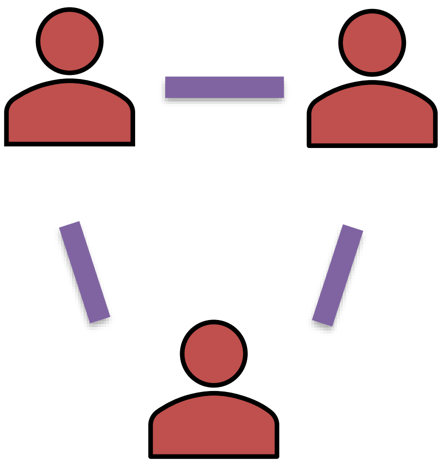
Need to show no matter the configuration, there are always 3 people so that

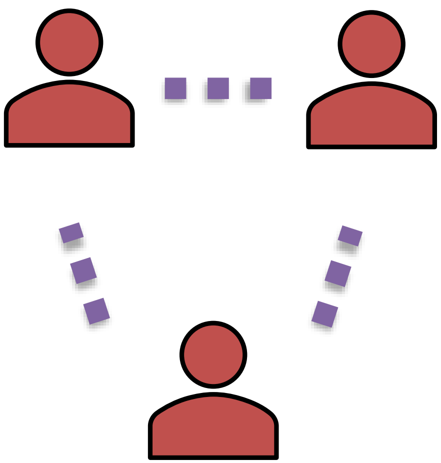


Let's consider all possible scenarios ... 2^{15}

Recap on proofs

Need to show no matter the configuration, there are always 3 people so that



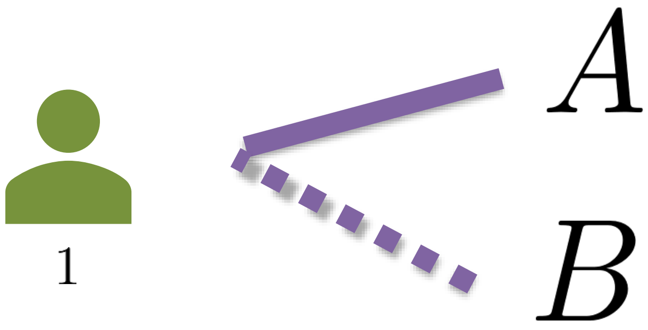


~~Let's consider all possible scenarios ... 2^{15}~~

Recap on proofs

Solution:

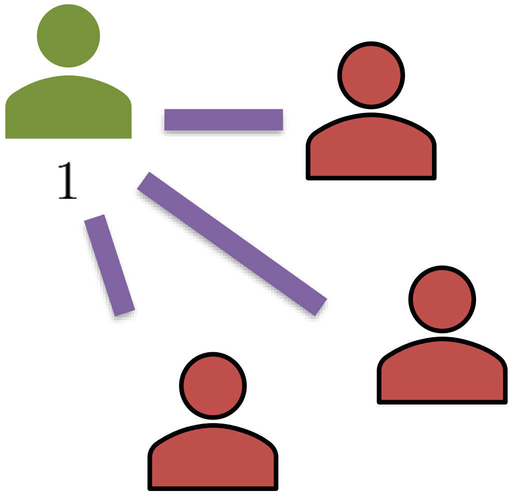
Consider the classmate with name 1 (green).



Either A or B has size at least three.

Recap on proofs

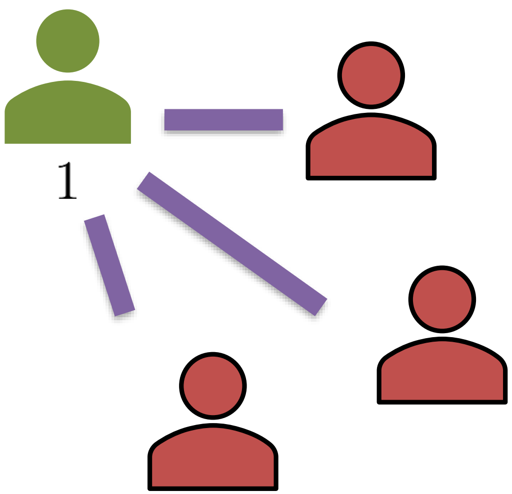
Solution:



Case 1: A is at least of size three

Recap on proofs

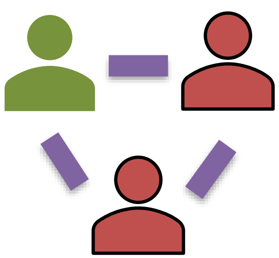
Solution:



Case 1: A is at least of size three

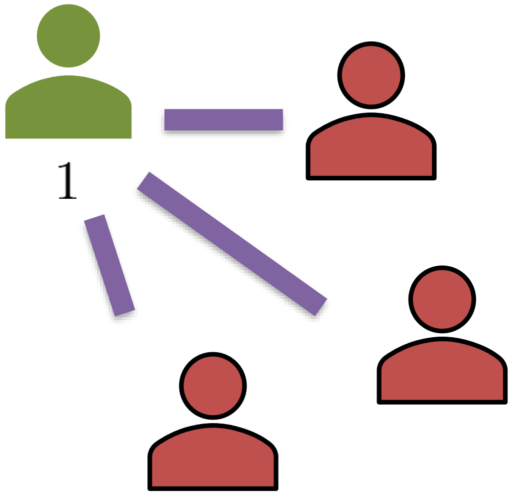
Subcase 1:

If at least two of the people in A have exchanged contacts then we found three people (two+ the green)



Recap on proofs

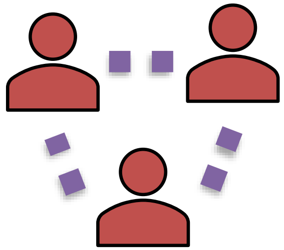
Solution:



Case 1: A is at least of size three

Subcase 2:

If all people in A have not exchanged contacts then we found three people

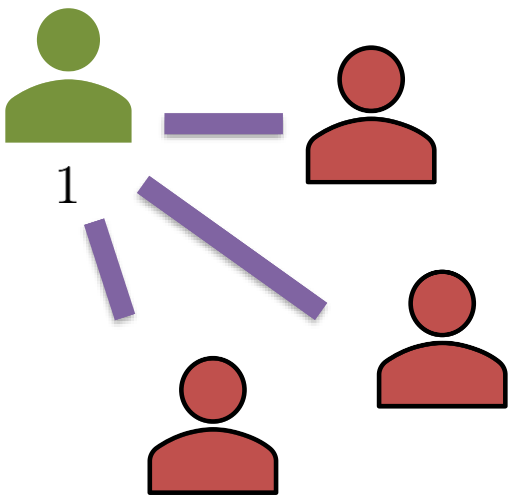


Recap on proofs

Case 2: B is at least of size 3 (similar)

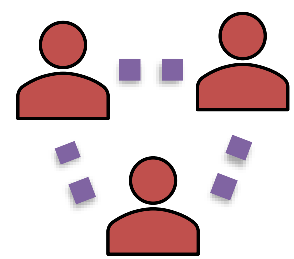
Solution:

Case 1: A is at least of size three



Subcase 2:

If all people in A have not exchanged contacts then we found three people



Recap on Asymptotics

- The asymptotic complexity describes $T(n)$, as n grows to **infinity**

- Focus on 3 types of Asymptotic complexity
  - Θ (Big Theta)
  - O (Big O)
  - Ω (Big Omega)

Recap on Asymptotics

- Θ (Big Theta) means "grows asymptotically $=$ "
- O (Big O) means "grows asymptotically $\leq$ "
- Ω (Big Omega) means "grows asymptotically $\geq$ "

Θ	O	Ω
$g \in \Theta(f)$ or $g = \Theta(f)$	$g \in O(f)$ or $g = O(f)$	$g \in \Omega(f)$ or $g = \Omega(f)$
$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \mathbf{C}$	$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \mathbf{C}$	$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \mathbf{C}$
$0 < \mathbf{C} < \infty$	$0 \leq \mathbf{C} < \infty$	$0 < \mathbf{C} \leq \infty$

Recap on Asymptotics

Big Θ examples:

$$n^3 \in \Theta(n^3)$$

$$n \log n + 0.001n^3 + 10^{17}n^2 \in \Theta(n^3)$$

$$2^{4n} \in \Theta(16^n) \ \text{ but } \ 2^{4n} \notin \Theta(2^n)$$

$$5^{n+2} \in \Theta(5^n)$$

- $g(n) \in \Theta(f(n))$ means "g grows as f, when n goes to infinity".

Recap on Asymptotics

Big O examples:

$$n^2 \in O(n^{100})$$

$$2n^3 + 1000n^2 + 10^{17} \in O(n^{299})$$

$$\log_2(2^n) \in O(n)$$

$$2^{n+1} \in O(2^{4n})$$

- $g(n) \in O(f(n))$ means "g grows at most as fast as f, when n goes to infinity".

Recap on Asymptotics

Big Ω examples:

$$n^{200} \in \Omega(n^{100})$$

$$2n^3 + 1000n^{350} + 10! \in \Omega(n^{298})$$

$$\log_2(4^n) \in \Omega(n)$$

- $g(n) \in O(f(n))$ means "g grows at least as fast as f, when n goes to infinity".

Recap on Asymptotics

Exercise 4: Show that $n \notin O(\ln n)$ but $n \in \Omega(\ln n)$

Recap on Asymptotics

Exercise 4: Show that $n \notin O(\ln n)$ but $n \in \Omega(\ln n)$

Solution:

Consider $\frac{n}{\ln n}$ and compute the limit $\lim_{n\to\infty} \frac{n}{\ln n}$.

Recap on Asymptotics

Exercise 4: Show that $n \notin O(\ln n)$ but $n \in \Omega(\ln n)$

Solution:

Consider $\frac{n}{\ln n}$ and compute the limit $\lim_{n\to\infty} \frac{n}{\ln n}$.

Challenge $\lim_{n\to\infty} \ln n = +\infty$ and $\lim_{n\to\infty} n = +\infty$

Recap on Asymptotics

Exercise 4: Show that $n \notin O(\ln n)$ but $n \in \Omega(\ln n)$

Solution:

Consider $\frac{n}{\ln n}$ and compute the limit $\lim_{n\to\infty} \frac{n}{\ln n}$.

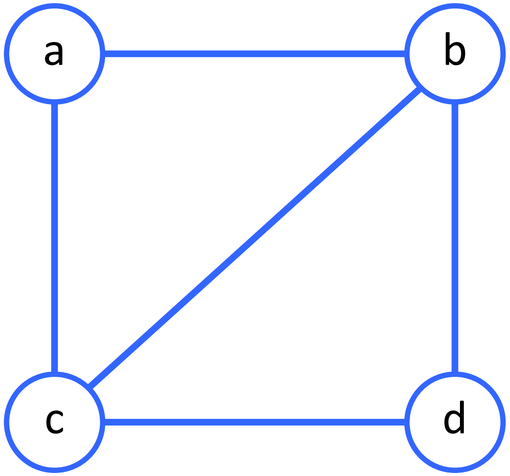
Challenge $\lim_{n\to\infty} \ln n = +\infty$ and $\lim_{n\to\infty} n = +\infty$

L'Hopital's rule: $\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \frac{g'(n)}{f'(n)}$

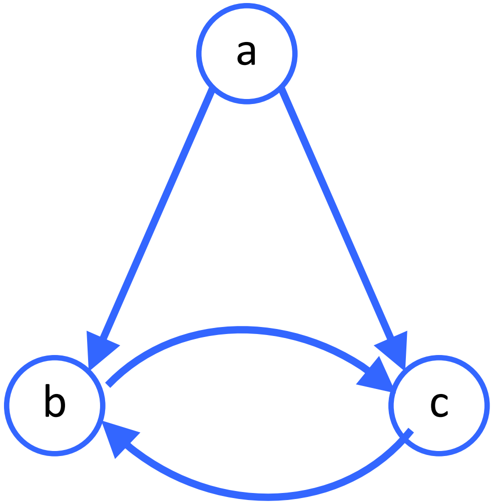
Therefore $\lim_{n\to\infty} \frac{n}{\ln n} = \lim_{n\to\infty} \frac{1}{1/n} = +\infty$ ∎

Recap on Graphs

- Undirected
- V={a,b,c,d}
- E={{a,b}, {a,c}, {b,c}, {b,d}, {c,d}}



- Directed
- V = {a,b,c}
- E = {(a,c), (a,b) (b,c), (c,b)}



- Representation

  Adjacency matrix/list, incidence list.

Recap on Graphs

Exercise 5: Given an undirected graph G with $\{1, 2, ..., n\}$ vertices and m edges, show that $\sum_{i=1}^{n} d(i) = 2m$.

Recap on Graphs

Exercise 5: Given an undirected graph G with $\{1, 2, ..., n\}$ vertices and m edges, show that $\sum_{i=1}^{n} d(i) = 2m$.

Solution:

The degree $d(i)$ of vertex i is the number of edges terminating in vertex i.

Now if you consider a particular edge (i, j), it will be counted once in $d(i)$ and once in $d(j)$, so exactly two times.

Recap on Graphs

Exercise 5: Given an undirected graph G with $\{1, 2, ..., n\}$ vertices and m edges, show that $\sum_{i=1}^{n} d(i) = 2m$.

Solution:

The degree $d(i)$ of vertex i is the number of edges terminating in vertex i.

Now if you consider a particular edge (i, j), it will be counted once in $d(i)$ and once in $d(j)$, so exactly two times.

Induction?

on the number of vertices n:

Recap on binary search

1	2	3	5	8	10	13

Canonical problem: Given a sorted array, find position of x.

Idea: Pick median (middle element). If we $x =$ median we are done.

Case 1: If x is greater than median,
repeat the process on the right half of the array.

Case 2: If x is smaller than median,
repeat the process on the left half of the array.

Example: above for $x = 10$.

Recap on binary search

- Consider

10	13	5	8	3	2	1

- An element $A[i]$ is a *peak* if it is not smaller than all its neighbor(s)
  - if $i \neq 1, n$: $A[i] \geq A[i-1]$ and $A[i] \geq A[i+1]$
  - If $i = 1$: $A[1] \geq A[2]$
  - If $i = n$: $A[n] \geq A[n-1]$

Exercise 6: find ***any*** peak.

Recap on binary search

Algorithm 1:

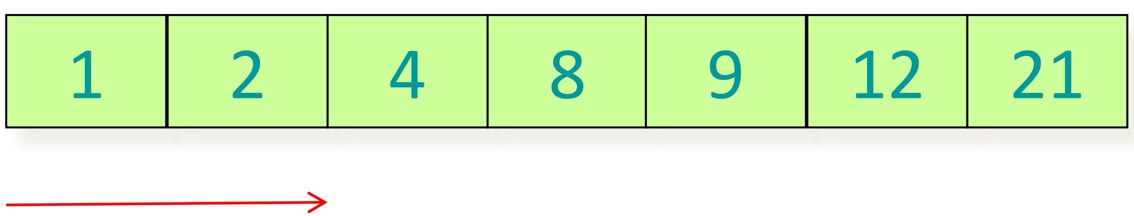
– Scan the array from left to right

– Compare each $A[i]$ with its neighbors

– Exit when found a peak

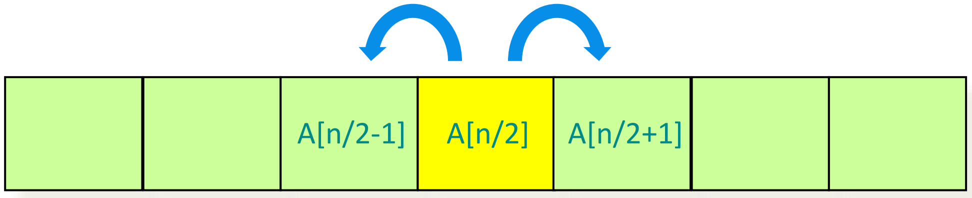
Worse-case Complexity:

– Might need to scan all elements, so $T(n)$ is $\Theta(n)$



Recap on binary search

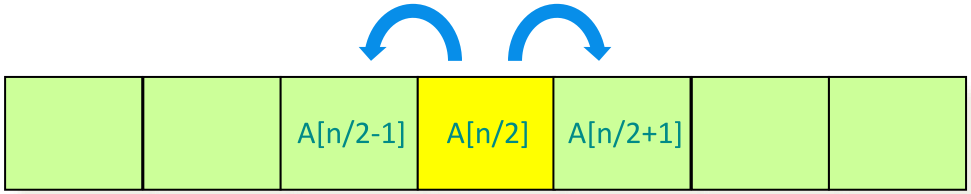
Algorithm 2:



- Consider the middle element of the array and compare with neighbors
  - If $A[n/2-1] > A[n/2]$
    then search for a peak among $A[1] \dots A[n/2-1]$
  - Else, if $A[n/2] < A[n/2+1]$
    then search for a peak among $A[n/2+1] \dots A[n]$
  - Else $A[n/2]$ is a peak!

Recap on binary search

Algorithm 2:



- Consider the middle element of the array and compare with neighbors
  - If $A[n/2-1] > A[n/2]$ then search for a peak among $A[1] \ldots A[n/2-1]$
  - Else, if $A[n/2] < A[n/2+1]$ then search for a peak among $A[n/2+1] \ldots A[n]$
  - Else $A[n/2]$ is a peak!

Running time $T(n) = T(n/2) + O(1)$ which gives $O(\log n)$.

Pseudocode

- High-level description of an algorithm
- Less detailed than a program
- Preferred notation for describing algorithms
- Hides program design issues

Pseudocode

Control flow:

```
if expr then
    body
else
    body
```

```
for expr do
    body
while expr do
    body
```

Expressions

- Equality testing
- Assignment ←
- Addition, subtraction, etc

Define methods/functions

Pseudocode

Example (running time $T(n)$ is $\Theta(n)$, linear time)

Algorithm Max(A, n)
Input: An array A storing n integers.
Output: Max element in A.

```
currentmax ← A[1]
For i = 2 to n do
    If currentmax < A[i] then
        currentmax ← A[i]
return currentmax
```

Need to Review (Reading)

- Sums, summations, Logarithms
- Asymptotics
- Data structures: Queues, stacks, lists, binary search trees
- Binary search
- Insertion and Selection sort
- Graph representation and DFS, BFS

We are here to help, please ask questions!

Next week Divide and Conquer Method