Let us analyze the brute force algorithm

line #1 $\Rightarrow O(1)$

#2 $\Rightarrow O(2^n \cdot n)$ $\quad n \leftarrow$ #vertices

#3 $\Rightarrow$ ~~$O(2^n)$~~ $O(2^n \cdot n^2)$ | $O(n^2)$ to check validity

#4 $\Rightarrow O(2^n \cdot n)$ | $O(n)$ to count #1's in assignment

#5 $\Rightarrow O(2^n \cdot 1)$

for each line, we take $O(x \cdot y)$

where x is the no. of times the line gets execu

and

y is the time to execute the line once.

Running time of algorithm is simply the max $\Rightarrow O(2^n \cdot n^2)$

Acceptable running time : Polynomial $O(n), O(n^2), O(n^{10})$

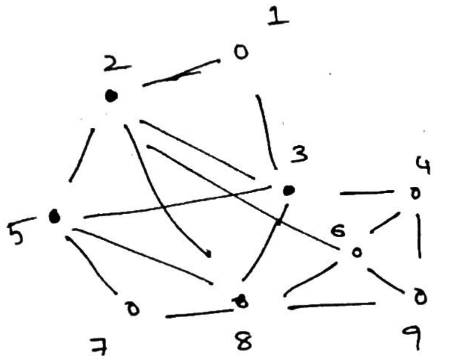
Unacceptable : Exponential $O(2^n)$ $O(1.1^n)$ $O(10^n)$ $O(3^n \cdot n^2)$

	Polynomial ?	Exponential ?
$O(2^n \cdot \log n)$		✓
$O(2^{\log n})$	✓	
$O(1.001^n)$		✓
$O(n^{1000})$	✓	
$O(2^n \cdot n^2)$		✓

polynomial, but acceptable ?!

exponential, but unacceptable ?!

Problem: Consider a graph $G(V, E)$ find a set of vertices V' such that ~~$\forall v \in$~~ V' every edge is connected to every other vertex in V' and find largest possible such set (clique)



$\Rightarrow \{2, 3, 5, 8\}$

idea: try all possible sets

---

largest_grp = 0
for each assignment of (0,1) to vertices:
  if assignment is valid:
    grp-size = # 1's in assignment
    largest-grp = max(largest-grp, grp-size)

---

running time $\Rightarrow O(2^n * n^2)$ line 3 is the bottleneck

Problems with only exponential time algorithms are called intractable problems

Vertex cover and Clique? We don't know yet!

$\Rightarrow$ SHOWING TRACTABILITY IS EASIER THAN SHOWING INTRACTABILITY

↳ show <u>one</u> polynomial time algorithm exists

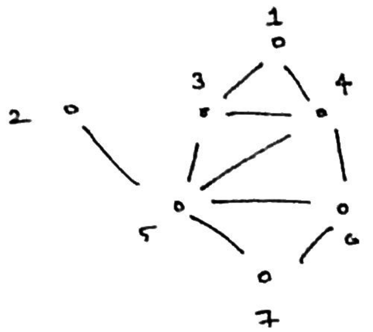
show <u>NO</u> polynomial time algorithm exists

Algorithm with $O(n^{10}) \Rightarrow$ is tractable ?

Algorithm with $O(1.01^n) \Rightarrow$ is intractable ?

Fortunately, this seldom occurs in practice...

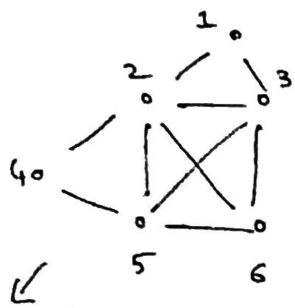
<u>New problem:</u> Consider a graph $G(V, E)$. Independent set is defined as a set of vertices where no two vertices are connected to each other. Find longest such set.



$\Rightarrow \{2, 4, 7\}$

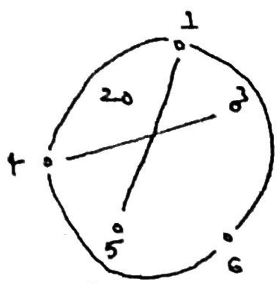
We can have a similar algorithm. $O(2^n \cdot n^2)$ like we had for clique.

Clique & Independent set are closely related. (one difference)



longest clique

$\{2, 3, 5, 6\}$



longest indepent set

$\{2, 3, 5, 6\}$

$\Longleftrightarrow$

The above 2 graphs are edge complemented if we overlap them, we get a fully connected graph.

1 represents an edge b/w two vertices, 0 represents absence.

flip 1's to 0's

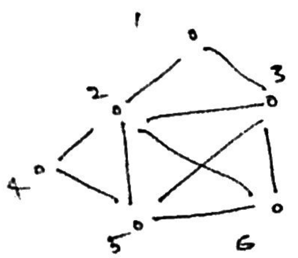
flip 0's to 1's

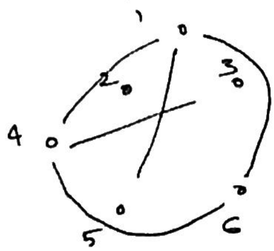
⟹ We can solve clique for a graph, if we solved independent set and vice versa.

They are equivalent in a way.

We can conclude that either both problems are tractable or intractable!

What about vertex cover? is it related to clique & i.s.?





vertex cover

{2, 3, 5 ~~▮~~}

clique {2, 3, 5, 6}

i.s {1, 4, 6}

vertex cover

{1, 4}

clique {1, 4, 6}

i.s. {2, 3, 5, 6}

vertex cover = V ~~clique~~ − i.s. (set subtraction)

↳ makes sense, since i.s. is longest possible set where no edges are present among any two vertices.

So vertex cover is tractable if one of i.s. and clique is tractable and goes both ways.

Reduction: transformation of one problem to another

RECAP:

		N = 100	N = 200	N = 300	N = 400	
Time	α	25	317,000	4.36×10^9	6.01×10^{13}	exponential
	β	6	11	16	21	linear
	γ	500	2000	4500	8000	polynomial

Best case running time for a given size is no longer than best measured ~~time~~ time for each algorithm — TRUE

Worst case running time for a given size is ~~no longer~~ at least the worst measured time for each algorithm — TRUE

β is the fastest for all inputs — FALSE (we don't know)

For small inputs, α is faster than γ — TRUE

Consider the algorithm

```
isPrime(n):
    if n < 2:
        return False
    if n == 2:
        return True
    if n % 2 == 0
        return False
    for i in (3, √n + 1):
        if n % i == 0:
            return False
    return True
```

Asymptotic analysis ⇒ $O(\sqrt{n})$

For this number of digits = d

$$d = \log_{10} n + 1 \Rightarrow 10^{d-1} = n$$

if we measure input size with d

$$\Rightarrow O\left(\sqrt{10^{d-1}}\right) \Rightarrow O\left(10^{\frac{d-1}{2}}\right) \Rightarrow O\left(10^{d/2}\right)$$

⇓

EXPONENTIAL!!!

in RAM model, arithmetic operations take constant time.

How DO WE MEASURE THE SIZE OF INPUT ??!

↳ # bits required to write the input

isPrime (int n)

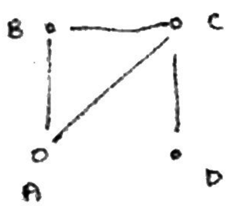
↳ proportial to $\log n$ (#digits/# bits)

isSorted (int [] arr)

↳ 32-bit ints, so $32 \cdot n$, where n is arr's length

isConnected (graph G)

↳ n+m, n ← # edges
m ← # vertices



≡

	A	B	C	D
A	0	1	1	0
B	1	0	1	0
C	1	1	0	1
D	0	0	1	0

memory ← $O(n^2)$

inverting graphs ⇒ flip 1's and 0's
(look out for loops!)