Logistics

- Quiz and exercise 7 released tomorrow
- Quiz due Tuesday at 11:59pm
- Exercise due Friday 11:59pm
  - Just a few reductions

CS3102 Theory of Computation

www.cs.virginia.edu/~njb2b/cstheory/s2020

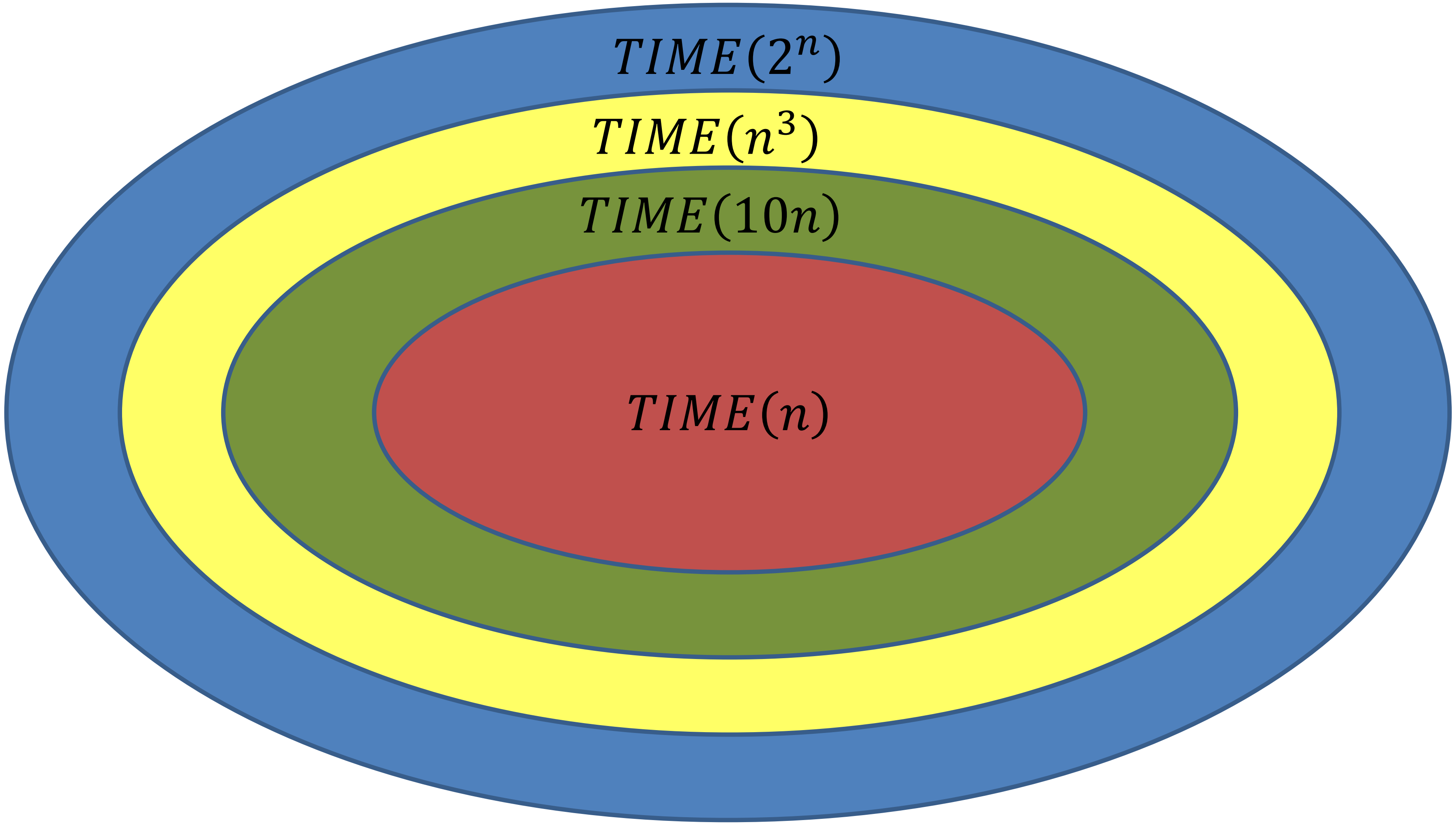
Warm up:

To measure the “cost” of computing something, what would units should we use?

Larger inputs = More time

- Run time is not measured by a number, but a function that maps input size to number of transitions
- **Running time:** $T(n)$ is a function mapping naturals to naturals. We say $F: \{0,1\}^* \rightarrow \{0,1\}^*$ is computable in $T(n)$ time if there exists a TM M s.t. for every large n and ever input $x \in \{0,1\}^n$, M halts after at most $T(n)$ steps and outputs $F(x)$.
- $TIME\big(T(n)\big)$ represents the set of boolean functions computable within $T(n)$ time
  - $TIME\big(T(n)\big)$ is a complexity class (a set of problems that all can be computed by a turing that uses no more than $T(n)$ steps)

More time gives more functions



RAM Machine

- We can go directly to a certain index in the tape

To transition:

1. Have a second tape to keep track of current location (increment each time we move right, decrement for left)
2. Have a third tape to record the target location
3. Move until the two tapes match
    1. Maybe we need another tape to do this?

(details not important, but if you want them, see 7.2 in text)

Important observation: Tape-machine takes more steps than a RAM machine (if a RAM-TM computes f in $T(n)$ time, a tape TM can compute f in $\left(T(n)\right)^4$ time, see theorem 12.5 for details)

Finding Running Times

- We will find running times for the following:
  - Shortest Path in a graph
  - Longest Path in a graph
  - 3SAT
  - 2SAT

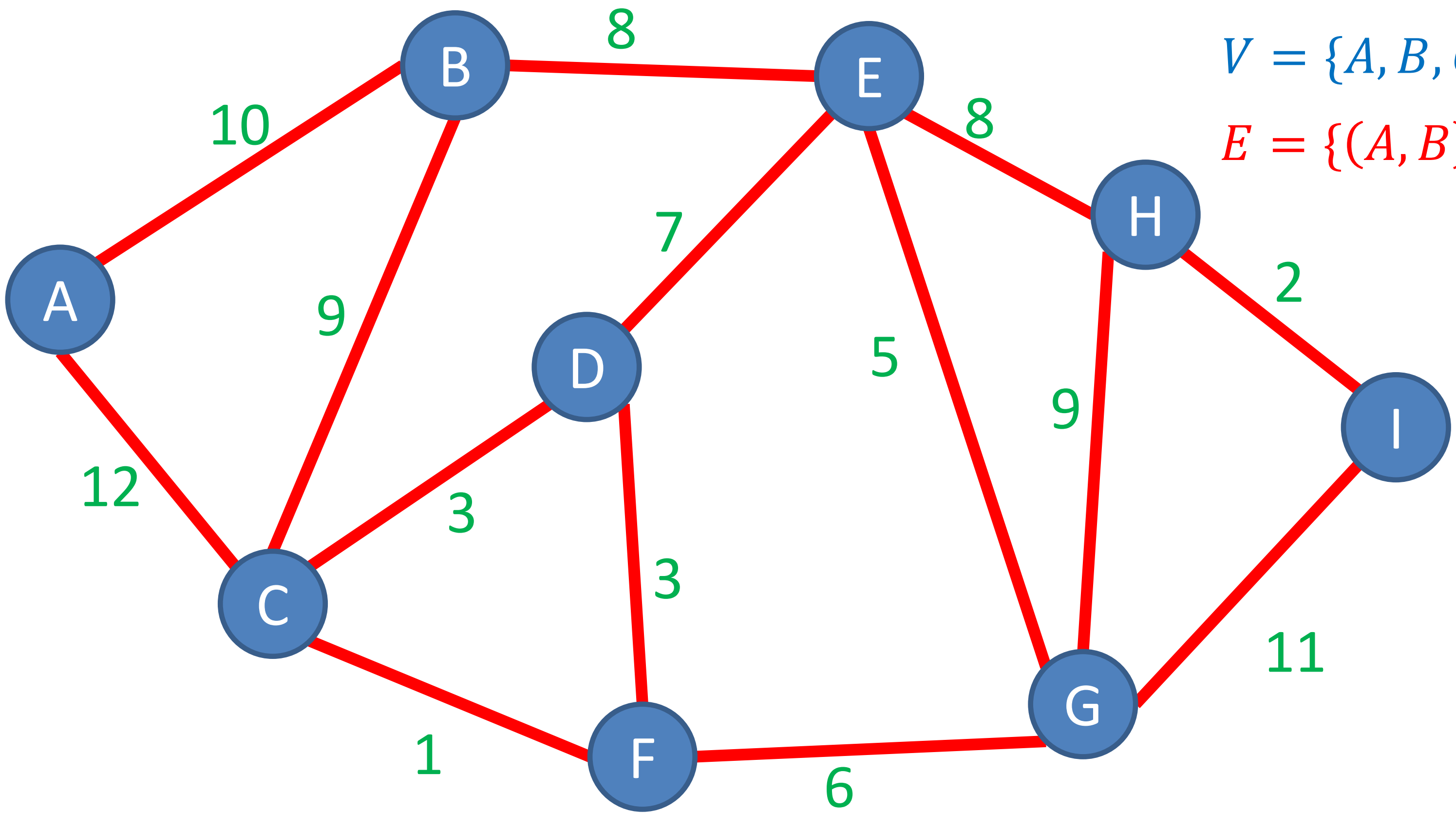
Graphs

Definition: $G = (V, E)$

Vertices/Nodes

Edges

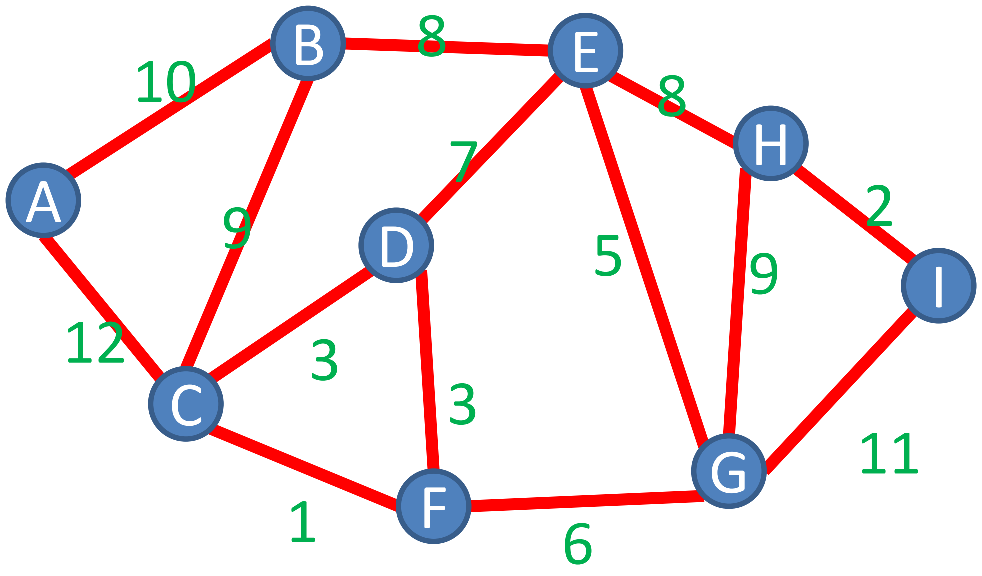
$w(e)$ = weight of edge e



$V = \{A, B, C, D, E, F, G, H, I\}$

$E = \{(A, B), (A, C), (B, C), \dots\}$

Adjacency List Representation



A	B	C		
B	A	C	E	
C	A	B	D	F
D	C	E	F	
E	B	D	G	H
F	C	D	G	
G	E	F	H	I
H	E	G	I	
I	G	H		

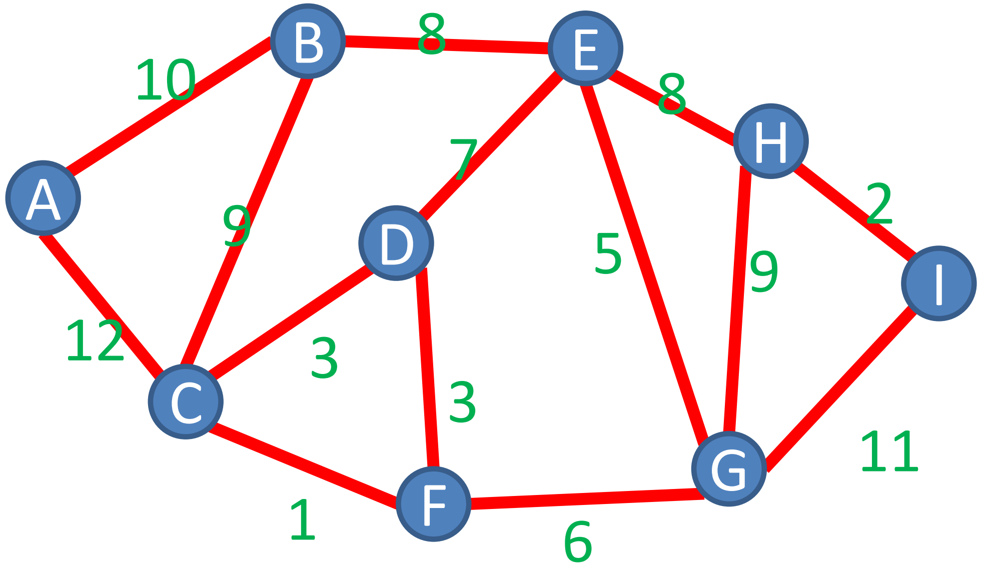
<u>Tradeoffs</u>

Space: $|V| + |E|$

Time to list neighbors: $Degree(A)$

Time to check edge (A, B): $Degree(A)$

Adjacency Matrix Representation



	A	B	C	D	E	F	G	H	I
A			1						
B	1		1		1				
C	1	1		1		1			
D			1		1	1			
E		1		1			1	1	
F			1	1			1		
G					1	1		1	1
H					1		1		1
I							1	1	

<u>Tradeoffs</u>

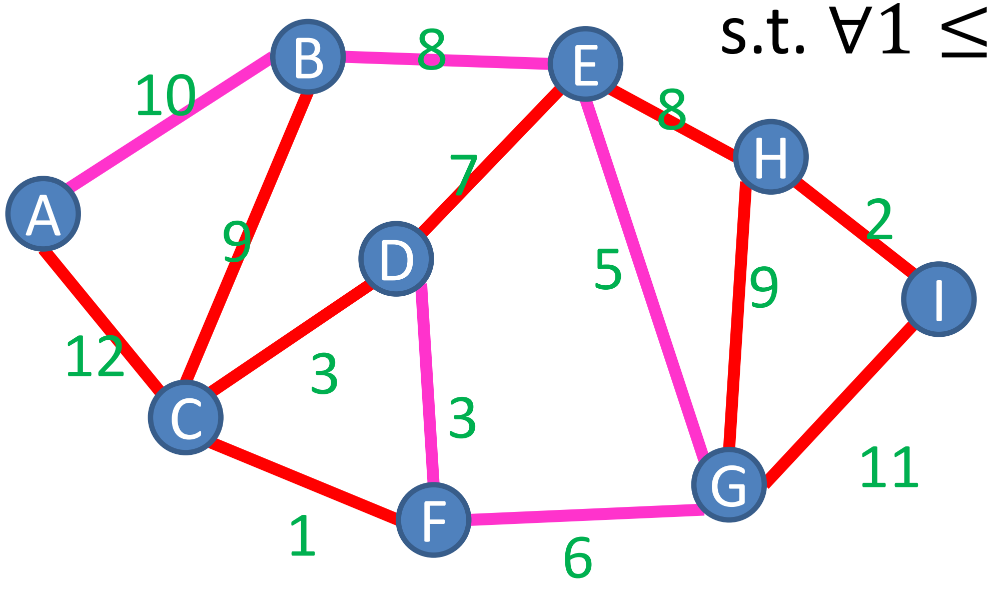
Space: V^2

Time to list neighbors: V

Time to check edge (A, B):$O(1)$

Definition: Path

A sequence of nodes $(v_1, v_2, \dots, v_k)$
s.t. $\forall 1 \le i \le k-1$, $(v_i, v_{i+1}) \in E$



<u>Simple Path:</u>

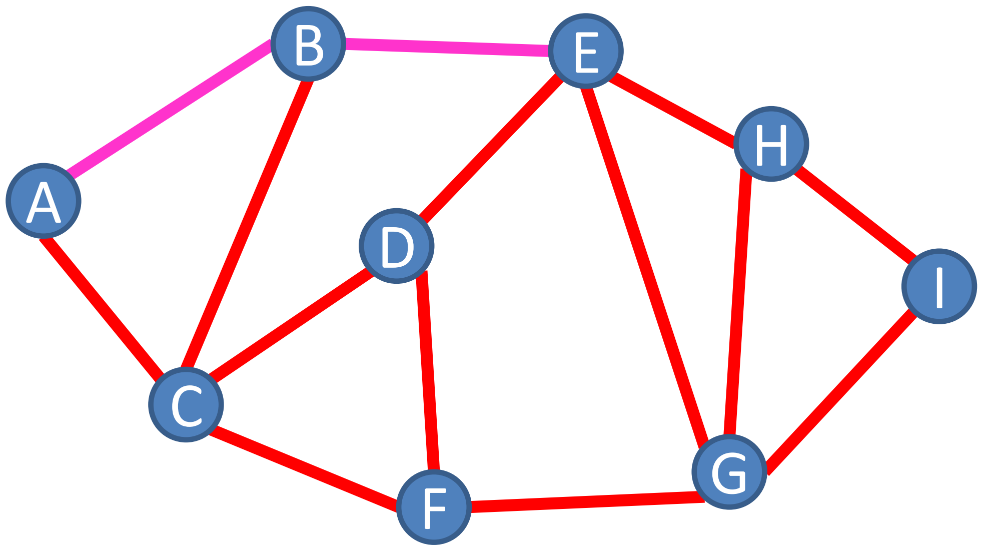
A path in which each node appears at most once

<u>Cycle:</u>

A path of > 2 nodes in which $v_1 = v_k$

Shortest Path

- Given an unweighted graph, start node s and an end node t, how long is shortest path from s to t?



Shortest path from A to E
has length 2

Breadth First Search

Find a path from s to t

Keep a queue Q of nodes

$hops = 0$

$Q.enqueue((s, hops))$

While Q is not empty and $v \mathrel{!}= t$:

    $v, hops = Q.dequeue()$

    for each "unvisited" $u \in V$ s.t. $(v, u) \in E$:

        $Q.enqueue((u, hops + 1))$

Running time: $O(|V| + |E|)$

Longest Path

- Given a start node s and an end node t, how long is longest path from s to t?

Longest Path

Enumerate all possible sequences of nodes
check if it’s a path
print the length of the longest one

Running time: $n!$

(3-)CNF

- Conjunctive Normal Form (CNF) formula:
  - Logical AND of clauses
  - Each clause being an OR of variables
- 3-CNF: Each clause has 3 variables

$$\underbrace{(x \vee y \vee z)}_{\text{Clause}} \wedge (x \vee \bar{y} \vee y) \wedge (u \vee y \vee \bar{z}) \wedge (z \vee \bar{x} \vee u) \wedge (\bar{x} \vee \bar{y} \vee \bar{z})$$

Variables

3-SAT

- Given a 3-CNF formula (logical AND of clauses, each an OR of 3 variables), Is there an assignment of true/false to each variable to make the formula true?

$$\underbrace{(x \lor y \lor z)}_{\text{Clause}} \land (x \lor \bar{y} \lor y) \land (u \lor y \lor \bar{z}) \land (z \lor \bar{x} \lor u) \land (\bar{x} \lor \bar{y} \lor \bar{z})$$

Variables

$$\boldsymbol{x = true}$$
$$\boldsymbol{y = false}$$
$$\boldsymbol{z = false}$$
$$\boldsymbol{u = true}$$

3-SAT algorithm

- Given a 3-CNF formula with n variables and m clauses, try all combinations of True/False, check to see if any combinations evaluate to True.

Running Time: 2^n

2-SAT

- Given a 2-CNF formula (logical AND of clauses, each an OR of 2 variables), Is there an assignment of true/false to each variable to make the formula true?

$$\underbrace{(x \vee y)}_{\text{Clause}} \wedge (x \vee \bar{y}) \wedge (y \vee \bar{z}) \wedge (z \vee u) \wedge (\bar{y} \vee \bar{z})$$

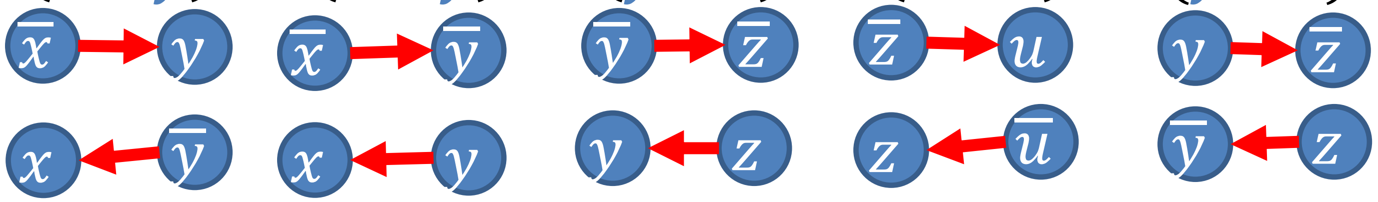
Variables

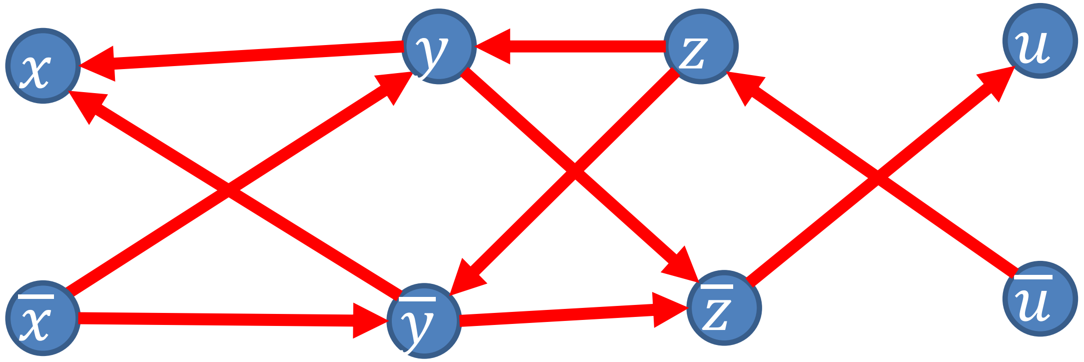
$$\boldsymbol{x = true}$$
$$\boldsymbol{y = false}$$
$$\boldsymbol{z = false}$$
$$\boldsymbol{u = true}$$

2-SAT in Polynomial Time

- Convert formula to an "implication graph"

$$(x \vee y) \wedge (x \vee \bar{y}) \wedge (y \vee \bar{z}) \wedge (z \vee u) \wedge (\bar{y} \vee \bar{z})$$





Are there any cycles with a variable and its negation?

2-SAT in Polynomial Time

- Convert formula to an "implication graph"

$(x \vee y) \wedge (\bar{x} \vee y) \wedge (x \vee \overline{y}) \wedge (\overline{x} \vee \overline{y})$

Are there any cycles with a variable and its negation?

3-SAT algorithm

- Given a 3-CNF formula with n variables and m clauses, try all combinations of True/False, check to see if any combinations evaluate to True.

Polynomial Time vs Exponential Time

- Polynomial Time: $P = \bigcup_{c\in\{1,2,3,\dots\}} n^c$
- Exponential Time: $EXP = \bigcup_{c\in\{1,2,3,\dots\}} 2^{n^c}$
- A strange pattern:
  - Most "natural" problems are either done in small-degree polynomial (e.g. n^2) or exponential time

Tractability

- Tractable:
  - Feasible to solve in the "real world"
- Intractable:
  - Infeasible to solve in the "real world"
- Whether a problem is considered "tractable" or "intractable" depends on the use case
  - For theory: Tractable = polynomial time, Intractable = Exponential time