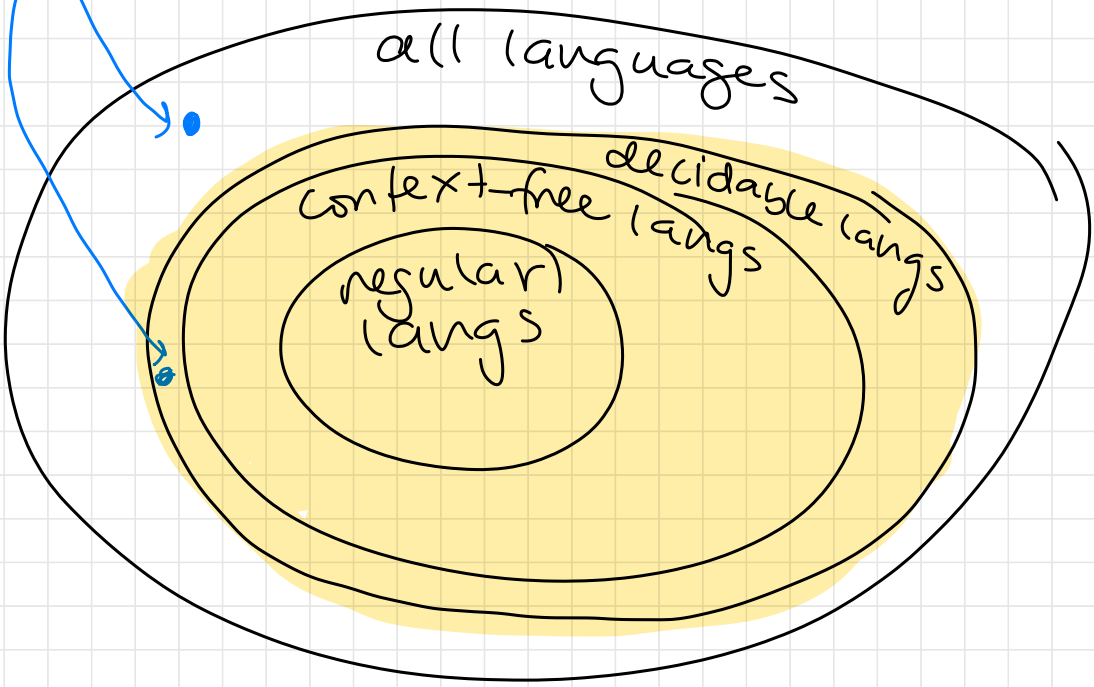
what is computable?

What is Computable?

$w \in L$ = "does object have property"

↑ computing set membership

eg. is number prime?
does this program accept the string "griz"?



Model of computation: TM

Church-Turing Thesis: TM is equivalent

to all reasonable models of computation
eg RM- can generate truly random numbers

How fast?

$n!$?

n^2 ?

$\log n$?

How fast can we solve problem A?

(1) Give an algorithm — last 6 wks
(upper bound)

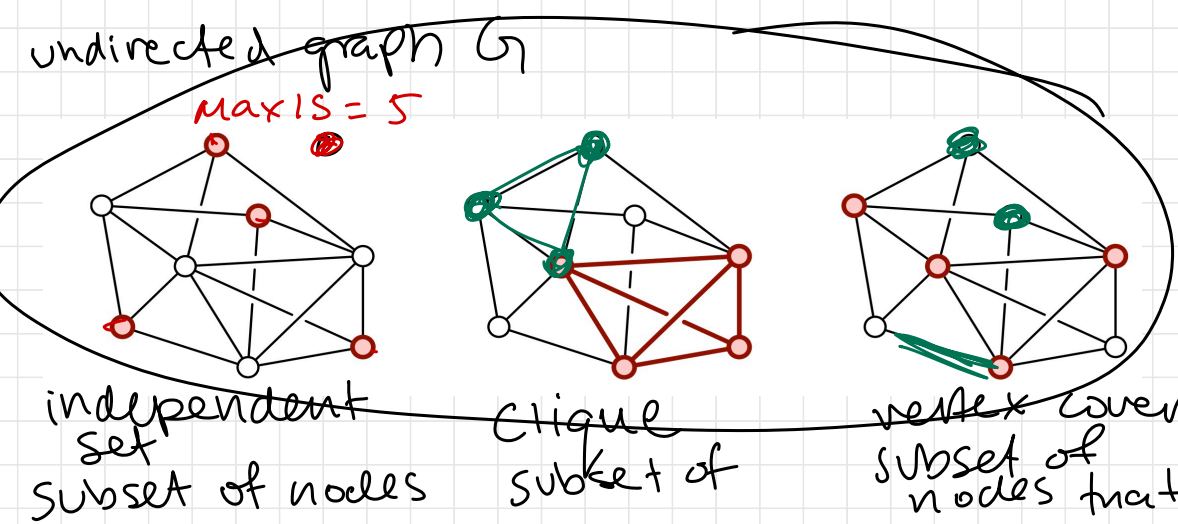
(2) Reductions (lower bound) ←

Assume we have a fast alg. for A.

Reduce a known problem B to A.

Give an algorithm to solve B uses A as a subroutine.

$\Rightarrow$ solve B fast



where no pair is connected by an edge

this G has an indep. set of size 4.

is there a bigger one?

maxIndSet(G): size of largest ind. set in G

eg 4

nodes where every node connected to every other node

maxClique(G): size of largest clique in G

4

cover all edges

minVC(G): size of smallest VC

Claim: MaxIndSet reduces to MaxClique.

PF

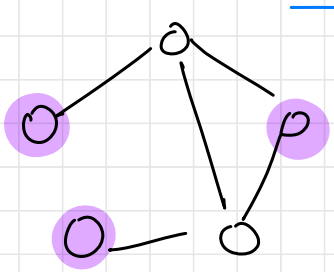
Suppose I have an (polynomial time) alg. for MaxClique (Magic black box solving MaxClique)!

MaxIndSet($G=(V,E)$):

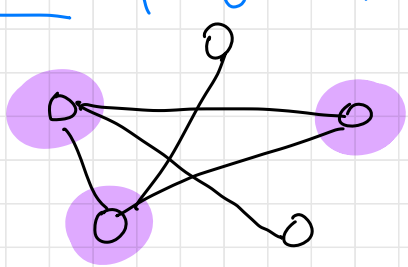
build a new graph $\bar{G}$ with $\bar{V}=V$, $\bar{E}=\{uv : uv \notin E\}$ — $O(n^2)$

Solve MaxClique on $\bar{G}$ — poly(n)

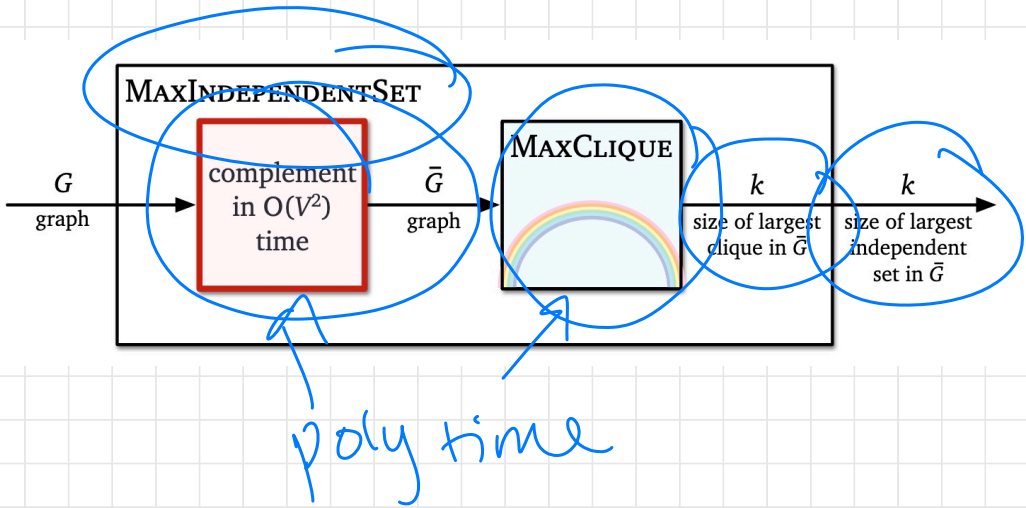
G



$\bar{G}$



runtime?
MaxIndSet can be solved in poly time.

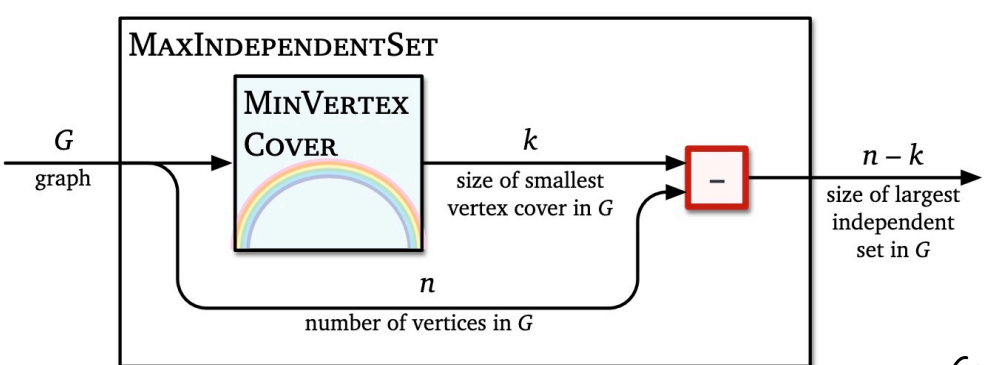


MaxIndSet reduces to MinVertex Cover

MaxIndSet(G):

let n = # verts in G

return n - MinVertexCover



Q: in our examples, MaxIndSet = MaxClique = MinVC

1. Suppose you are given a magic black box that somehow answers the following decision problem *in polynomial time*:

   - INPUT: A directed graph G and a positive integer L. (The edges of G are not weighted, and G is not necessarily a dag.)
   - OUTPUT: TRUE if G contains a (simple) path of length L, and FALSE otherwise.[1]

(a) Using this black box as a subroutine, describe algorithms that solves the following optimization problem *in polynomial time*:

   - INPUT: A directed graph G.
   - OUTPUT: The length of the longest path in G.

   [Hint: You can use the magic box more than once.]

(b) Using this black box as a subroutine, describe algorithms that solves the following search problem *in polynomial time*:

   - INPUT: A directed graph G.
   - OUTPUT: The longest path in G