the power of \( \exists \)
non-deterministic polynomial

- non-determinism allows a choice of next step
- used for yes/no problems
- if there is a choice of next steps that makes it say yes, then it “accepts”
- runs in polynomial time

- \text{ACCEPTS} \text{ iff } \exists \text{poly-length computation leading to “yes”}
some of our fave NP problems

3SAT: given 3cnf formula F on variables $x_1, x_2, ..., x_n$, is there an assignment of true/false to the $x_i$ which makes $F[x_1, ..., x_n]$ true?

ex: $(x_3 \lor \neg x_5 \lor \neg x_2) \land (x_1 \lor x_2 \lor \neg x_3) \land (x_3 \lor \neg x_5 \lor \neg x_2) \land (x_3 \lor \neg x_5 \lor \neg x_2)$

3cnf instance:
- a literal is an $x_i$ or an $\neg x_i$
- a clause is the OR of up to 3 literals
- a formula is the AND (conjunction) of clauses

nondeterministic algorithm

```
for i = 1 to n
    set $x[i] = 0 \text{ or } 1$ (nondet step)

if $F[x[1], ..., x[n]]$ is true
    then ACCEPT
else REJECT
```
3COL: Given a graph $G$, is there an assignment of 3 colors to the nodes of $G$ so that no two adjacent nodes have the same color?

HP: (Hamilton path) Does $G$ have a path that starts at one node, ends at another, and visits all the other nodes exactly once?

TSP: (travelling salesman problem) Given a weighted graph $G$ and a bound $B$, is there a cycle that traverses all nodes of $G$ and has total weight at most $B$?

LP: Given a graph $G$ and integer $B$, is there a simple path in $G$ of length at least $B$?
**Vertex Cover:** Given a graph G and integer k, is there a set C of at most k vertices such that each edge of G has an endpoint in C?

**Independent Set:** Given a graph G, is there a set I of at least k vertices such that no two vertices in I are connected by an edge?

*note:* C is a vertex cover iff V-C is an independent set

**Clique:** Given a graph G and integer k, is there a set C of at least k vertices such that all pairs of nodes in C are connected by an edge?

**Subset Sum:** Given a set S of integers and an integer W, is there a subset of S that sums to exactly W?
Sequencing with release times and deadlines: Given a set of tasks, where each task $t$ has a release time $r(t)$, length $l(t)$, and deadline, $d(t)$, is there a schedule on a single processor so that each $t$ is processed after $r(t)$ and finishes before $d(t)$?

Knapsack:

Bin Packing:
many characterizations of NP

• problems with short proofs that can be verified in poly-time

• a set $A$ is in NP if there is a poly-time checkable relation $R$ such that
  $$A=\{ x \mid \exists y \ (|y| \leq |x|^k) \ R(x,y) \}$$

• existential second-order logic (Fagin’s Theorem)

• problems with proofs easy to check, hard to find
NP-Complete

the hardest problems in NP

if one NP-complete problem can be solved in poly-time, then all of NP can be solved in poly-time

main point:
no one knows if any NP complete problem can be efficiently solved – it is one of the big open problems of computer science and/or mathematics
handled by reductions

for example, the problem of 3SAT can be reduced to 3COL, which we write $3\text{SAT} \leq_p 3\text{COL}$

to be done in class, and is a “proof by widget”, but uses the graph below

\textbf{note}: this graph cannot be colored with three colors if the corners have the same color
reductions

• we say $A \leq_p B$ if a poly-time solution to $B$ gives a poly time solution to $A$

• $K$ is $NP$-hard if for all $A$ in $NP$, $A \leq_p K$

• $K$ is $NP$-complete if $K$ is NP-hard and $K$ is in $NP$

• **Cook-Levin Theorem**: SAT is NP-complete
counting classes

• #P
• called “sharp-P”
• a set of functions which count the number of accepting computations of NP machines
• example: #SAT, the number of satisfying formulas of a boolean formula F
• #P is much stronger than NP
quantum computers

• very interesting, very hyped
• possibly can be built (20-50 years??)
• good at solving some periodic problems: factoring
• BQP: bounded quantum probabilistic time
• almost certainly will not solve all of NP
• https://www.scottaaronson.com/qclec.pdf (lecture notes)
Figure 24.1: Where BQP fits in with classical computational complexity classes, as best as we know it today.
SAT is NP-Complete (from scratch)

• Let $A$ be any set in NP. Then it is accepted by a NDTM $M$ in time $p(n)$ (a polynomial)
• that is, if $x \in A$, there is an accepting computation of $M$ on input $x$ of length at most $p(|x|)$
• so for any string $x$, we will construct a formula $\varphi_{M,x}$ such that

$$\varphi_{M,x} \text{ is satisfiable iff } M \text{ accepts } x \text{ in time } p(|x|)$$
variables of $\varphi_{M,x}$

1. $C<i,j,t>$ is true iff the $i^{th}$ cell of M’s tape contains the $j^{th}$ symbol at time $t$
2. $S<k,t>$ is true iff M is in state $q_k$ at time $t$
3. $H<i,t>$ is true iff at time $t$ the tape head of M is scanning the $i^{th}$ cell