P, NP, NP-COMPLETENESS

WE HAVE THROUGHOUT THIS COURSE FOCUSED ON SPECIFIC ALGORITHMS AND DATA STRUCTURES, LOOKING AT SOLVING SPECIFIC PROBLEMS AND ANALYZING SPECIFIC RUNNING TIMES.

TODAY I'M GOING TO GIVE YOU AN INTRODUCTION TO COMPLEXITY THEORY THE THEORY UNDERLYING THE STUDY OF HOW DIFFICULT PROBLEMS ARE TO SOLVE, AND THEIR RELATIONSHIP.

COMPLEXITY THEORY IN PARTICULAR DIVIDES UP PROBLEMS INTO CLASSES BASED ON THE RUNNING TIME OF KNOWN ALGORITHMS TO SOLVE THOSE PROBLEMS.

MOREOVER, THOSE CLASSES ARE ROBUST THE CLASSIFICATION OF PROBLEMS IS MOSTLY INDEPENDENT OF THE COMPUTATIONAL MODEL USED TO ANALYZE ALGORITHMS.

└ RAM MODEL, ETC

MOST ALGORITHMS WE SAW HAVE A RUNNING TIME LIKE

$\Theta(N)$
$\Theta(N \log N)$
$\Theta(N^2)$

THESE ARE ALL POLYNOMIAL-TIME ALGORITHMS — THEIR RUNNING TIME IS $O(N^k)$ FOR SOME K.

$\Theta(N)$ is $O(N)$
$\Theta(N \log N)$ is $O(N^2)$
$\Theta(N^2)$ is $O(N^2)$

MATRIX MULTIPLICATION IS $O(N^3)$ WHERE N IS THE SIZE OF THE MATRICES BEING MULTPLIED, ETC.

DEFINITION: P IS THE CLASS OF PROBLEMS SOLVABLE BY A POLYNOMIAL-TIME ALGORITHM.

P IS GENERALLY KNOWN AS THE CLASS OF PROBLEMS THAT CAN BE SOLVED "EFFICIENTLY"

(SOMEWHAT TONGUE-IN-CHEEK)

NOT EVERY SOLVABLE PROBLEM IS IN P.

SOME PROBLEMS REQUIRE EXPONENTIAL TIME

EXP IS THE CLASS OF PROBLEMS SOLVABLE BY AN ALGORITHM THAT RUNS IN EXPONENTIAL TIME $O(2^{p(n)})$ WHERE p IS A POLYNOMIAL

IS THERE ANY PROBLEMS HERE?

IT'S TRICKY — WE NEED A PROBLEM WHERE WE CAN SHOW THERE IS NO ALGORITHM SOLVING IT THAT CAN RUN IN POLYNOMIAL TIME

SAMPLE EXPTIME PROBLEMS

— SOLVING CHESS, GO, CHECKERS

INTUITIVELY - EXPONENTIALLY MANY GAMES WRT TO WHAT BOARD OF SIZE n.

EXAMPLE: A PEBBLE GAME IS A TUPLE (X, R, S, t)

WHERE: X IS A FINITE SET OF NODES

R IS A SET OF RULES, EACH OF THE FORM (x, y, z) WHERE $x \neq y \neq z$, $x, y, z \in X$ STATING: IF THERE'S A PEBBLE ON x AND ON y, BUT NONE ON z, CAN MOVE A PEBBLE FROM x TO z.

S A START CONFIGURATION OF PEBBLES ON X

t A NODE IN X.

2-PLAYER PEBBLE GAME: EACH PLAYER MOVES A PEBBLE ACCORDING TO RULES UNTIL

1. A PEBBLE IS ON t (A WIN)
2. A PLAYER CANNOT MOVE (A LOSS)

DETERMINING IF PLAYER 1 HAS A <u>WINNING STRATEGY</u> IS IN EXP (BUT NOT IN P)

SIDE NOTE — NOT EVERY PROBLEM IS SOLVABLE!

E.G. PCP

GIVEN A SET OF "DOMINOES"

$$\binom{x_1}{y_1} \cdots \binom{x_n}{y_n}$$ WHERE $x_1, \ldots, x_n$, $y_1, \ldots, y_n$ ARE STRINGS

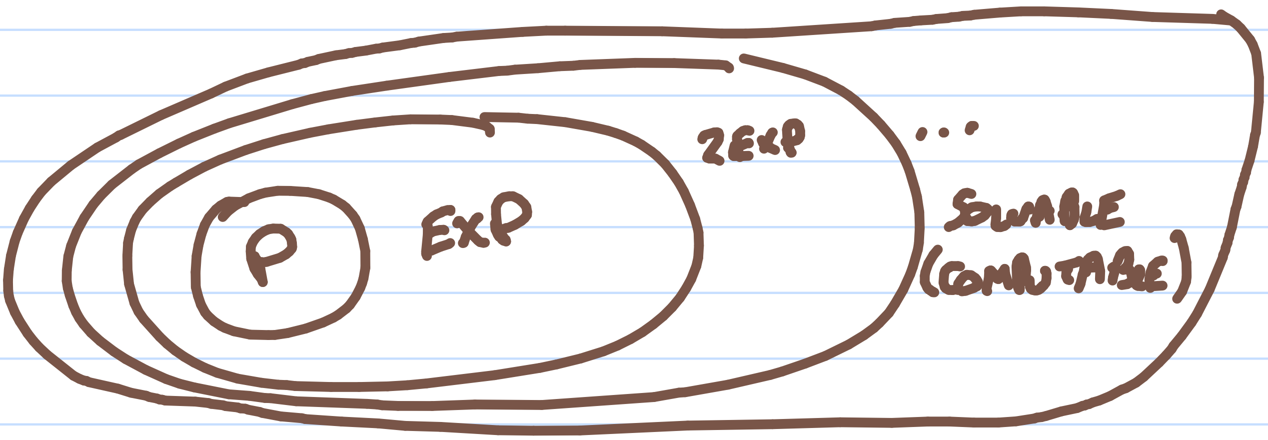
FIND A SEQUENCE OF INDICES $i_1, i_2, \ldots, i_k$ SUCH THAT (ALLOW REPETITIONS)

$$x_{i_1} x_{i_2} x_{i_3} \cdots x_{i_k} = y_{i_1} y_{i_2} y_{i_3} \cdots y_{i_k}$$

OR SAY "THERE IS NO SUCH SEQUENCE"

THERE DOES NOT EXIST <u>ANY</u> ALGORITHM TO SOLVE THIS PROBLEM!

(WHY? TAKE FOCS)



EXAMPLES:

CONSIDER THE DOMINOES

$$\underset{1}{\begin{pmatrix} b \\ aa \end{pmatrix}} \quad \underset{2}{\begin{pmatrix} a \\ ab \end{pmatrix}} \quad \underset{3}{\begin{pmatrix} ababb \\ b \end{pmatrix}}$$

HERE'S A SOLUTION TO PCP:

2 1 2 2 3

BECAUSE

$$\begin{pmatrix} a \\ ab \end{pmatrix} \begin{pmatrix} b \\ aa \end{pmatrix} \begin{pmatrix} a \\ ab \end{pmatrix} \begin{pmatrix} a \\ ab \end{pmatrix} \begin{pmatrix} ababb \\ b \end{pmatrix}$$

THE TOP ROW SPELLS <u>ab a a ababb</u>

THE BOTTOM ROW SPELLS <u>ab aa ababb</u>

BUT THE DOMINOES

$$\begin{pmatrix} a \\ ab \end{pmatrix} \begin{pmatrix} b \\ ba \end{pmatrix} \begin{pmatrix} ab \\ ba \end{pmatrix} \begin{pmatrix} ba \\ ab \end{pmatrix}$$

HAVE NO SOLUTION (WHY?)

WHY IS THIS INTERESTING?
BECAUSE IT PROVIDES A FOUNDATION FOR CLASSIFYING/ANALYZING PROBLEMS WHOSE STATUS IS UNKNOWN

E.G. SUBSET SUM

GIVEN A SET $\{n_1, \ldots, n_k\}$ OF NATURAL NUMBERS, AND A NATURAL NUMBER M, IS THERE A SUBSET $S \subseteq \{n_1, \ldots, n_k\}$ SUCH THAT $\sum S = M$?

BEST KNOWN ALGORITHM IS EXPONENTIAL — ENUMERATE ALL SUBSETS OF $\{n_1, \ldots, n_k\}$, CHECK THEIR SUM AGAINST M.

```
SUBSET_SUM(S, M) ≡
  IF |S| = 0:
    IF M = 0:
      RETURN TRUE
    ELSE:
      RETURN FALSE
  PICK N ∈ S
  RETURN SUBSET_SUM(S \ {N}, M)
      OR SUBSET_SUM(S \ {N}, M-N)
```

RUNNING TIME $\Theta(2^{|S|})$

WE DON'T KNOW OF A POLYNOMIAL TIME ALGORITHM FOR SUBSET SUM

WE ALSO DON'T KNOW THERE ISN'T ONE!

SO WE KNOW SUBSET SUM IS IN EXP. BUT WE DON'T KNOW IF IT'S IN P.

WE CAN SAY MORE THOUGH. EVEN THOUGH WE DON'T KNOW WHETHER WE CAN SOLVE SUBSET SUM "EFFICIENTLY", WE KNOW WE CAN VERIFY A CANDIDATE SOLUTION EFFICIENTLY.

FOR SIMPLICITY, FOCUS ON <u>DECISION PROBLEMS</u> — TRUE/FALSE PROBLEMS

A <u>VERIFIER</u> FOR A PROBLEM Q IS AN ALGORITHM

A S.T. FOR ALL x WHERE Q[x] IS TRUE, A(x, c) = TRUE FOR SOME c

→ c IS A "CERTIFICATE" HELPING TO "PROVE" THAT Q[x] IS TRUE

A POLYNOMIAL-TIME VERIFIER IS A VERIFIER THAT RUNS IN TIME POLYNOMIAL IN THE SIZE OF x.

DEF: NP IS THE CLASS OF ALL PROBLEMS WITH A POLYNOMIAL TIME VERIFIER.

CLEARLY, ANY PROBLEM IN P HAS A POLYNOMIAL TIME VERIFIER — JUST IGNORE THE CERTIFICATE.

SUBSET-SUM IS IN NP — PASS A SUBSET OF S SUMMING UP TO M AS A CERTIFICATE — THE VERIFIER SIMPLY VERIFIES THAT THE SUM OF THE SUBSET IS M.

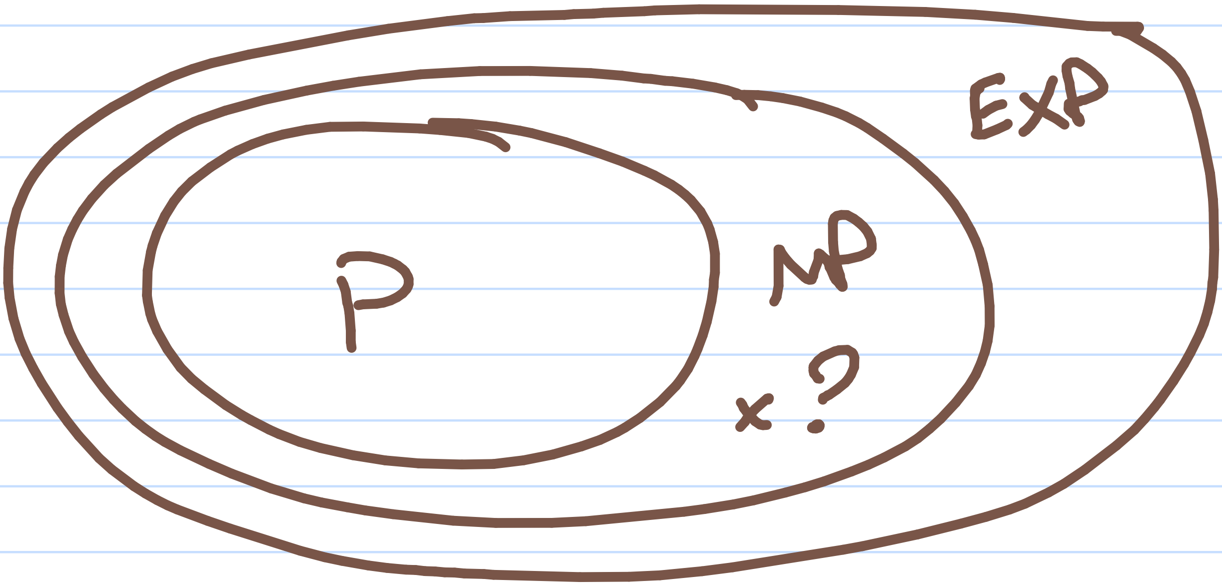
MANY MANY PROBLEMS ARE NOT KNOWN TO BE IN P BUT THEY ARE IN NP:

- SATISFIABILITY
- 3-COLORING ON GRAPHS
- VERTEX COVER
- HAMILTONIAN CYCLE (VISIT VERTICES EXACTLY ONCE)

NP PROBLEMS ARISE FREQUENTLY AND NATURALLY

- MANY PROBLEMS CALL FOR THE DESIGN OF AN ARTIFACT
- THE ARTIFACT IS OFTEN A MATHEMATICAL ABSTRACTION OF AN ACTUAL PHYSICAL OBJECT, SO THE OBJECT IS NOT TOO LARGE WITH RESPECT TO THE INPUT (I.E., POLYNOMIAL)
- THE DESIGN SPECIFICATION IS USUALLY SIMPLE (I.E., CHECKABLE IN POLYNOMIAL TIME)
- THE ARTIFACT SOUGHT IS THE CERTIFICATE

MOST PROBLEMS FIT THIS PATTERN.



WE KNOW $P \subseteq NP \subseteq EXP$

WHAT IS THE RELATIONSHIP BETWEEN P AND NP?

- $P \subset NP$
- $P = P$

THIS IS THE INFAMOUS $\underline{P \text{ VS } NP}$ PROBLEM

WE STILL DON'T KNOW WHICH RELATIONSHIP HOLDS — MOST BELIEVE $P \subset NP$ BUT NOT PROVED YET (AND I DON'T THINK IT WILL BE IN OUR LIFETIME)

WHY DO MOST BELIEVE THAT $P \subset NP$ BUT ARE NOT EQUAL?

BECAUSE OF THE PHENOMENON OF <u>COMPLETENESS</u>

TO DEFINE THE NOTION OF COMPLETENESS, WE NEED THE CONCEPT OF <u>REDUCTION</u>

↳ WHEN DOES A PROBLEM REDUCES TO ANOTHER PROBLEM?

EXAMPLE: MULTIPLICATION REDUCES TO ADDITION

IF YOU MAKE A WAY TO ADD, YOU CAN USE IT TO MULTIPLY

$$n * m \longrightarrow \underbrace{m + m + \dots + m}_{n \text{ TIMES.}}$$

PROBLEM Q (POLYNOMIAL-TIME) REDUCES TO R IF THERE IS IS A POLYNOMIAL TIME ALGORITHM F TRANSFORMING INSTANCES OF Q TO INSTANCES OF R SUCH THAT

$Q(x)$ IS TRUE

IFF

$R(F(x))$ IS TRUE.

WRITTEN $Q \leq_p R$

IDEA: IF YOU CAN SOLVE R, YOU CAN USE IT TO SOLVE Q.

A PROBLEM IN NP IS <u>NP-COMPLETE</u> IF EVERY PROBLEM IN NP REDUCES TO IT

↳ IT IS ONE OF THE "HARDEST" PROBLEM IN NP

THE PROBLEMS ABOVE ARE NP-COMPLETE

- SUBSET SUM
- 3 COLORING
- ...

IF WE FIND A POLYNOMIAL-TIME ALGORITHM FOR <u>ANY</u> NP-COMPLETE PROBLEM, IT WILL <u>MAKE</u> P=NP, AND WILL GIVE US A POLYNOMIAL TIME ALGORITHM FOR <u>ALL</u> OTHER NP-COMPLETE PROBLEMS.

E.G., A POLYNOMIAL TIME ALGORITHM FOR SUBSET SUM WILL GIVE US A POLYNOMIAL TIME ALGORITHM FOR 3-COLORING GRAPHS, OR SATISFYING BOOLEAN FORMULAS!

WHY?

IF Q IS NP-COMPLETE, AND HAS A POLYNOMIAL-TIME ALGORITHM A, THEN FOR ANY OTHER PROBLEM R IN NP:

HERE'S AN ALGORITHM FOR R — SINCE $R \leq_p Q$, LET F_R BE THE TRANFORMATION IN THE REDUCTION.

$ALG_R(x)$:

$y \leftarrow F(x)$ ← POLYNOMIAL
RETURN $A(y)$ ← POLYNOMIAL

BOOM. CAN DO THIS FOR ANY PROBLEM IN NP.

SINCE NOBODY HAS EVER COME UP WITH A POLYNOMIAL TIME ALGORITHM FOR ANY PROBLEM IN NP, LET ALONE THE NP-COMPLETE ONES, THIS STRONGLY SUGGESTS $P \neq NP$.