NP-COMPLETENESS: a brief informal introduction

We've seen algorithms with several time complexities:

- $O(\log n)$
- $O(n)$
- $O(n \log n)$
- $O(n^2)$
- $O(n^c)$
- $O(2^n)$

Focus on decision problems.

$P$: We can verify solutions in $O(n^c)$.

$NP$: We haven't found $O(n^c)$ yet, but algorithms in $O(n^c)$ are not polynomial in $w(n^c)$.
NP: non-deterministic polynomial

(NOT "non-polynomial")

The $1,000,000$ question...

$P \overset{\text{?}}{=} NP$

Is it ever "much" harder to solve a problem than it is to verify a solution, if the verification takes poly-time?

not within a polynomial factor:

\[
T(\text{verify}) = o(n^c \cdot T(\text{solve})) \\
T(\text{solve}) = \omega(n^c \cdot T(\text{verify}))
\]
There are thousands of problems for which no known polynomial-time solution is known, yet we can verify proposed solutions in poly-time.

e.g. Hamiltonian cycle

given a graph, find a cycle that visits each vertex exactly once.

\[ \text{\LARGE \rightharpoonup \ decide \ if \ one \ exists} \]
DECISION PROBLEM

"is there a set of k independent vertices?"

OPTIMIZATION PROBLEM

"find the largest independent set" (size)

(independent: no neighbors)

binary search on k: 0...|V|
NP-COMPLETE PROBLEMS

1) in NP, & not known to be in P
   (decision problems with solutions that can be verified in poly-time, but for which no poly-time algo is known)

2) if you ever find a polynomial-time solution for any NPC problem, this implies the same for every problem in NP. \( \rightarrow P = NP \)

\( \rightarrow \) if you ever prove that an NPC problem has no poly-time algo, then no NPC problem does \( \rightarrow P \neq NP \)
Are there other problems in NP but not in P or NPC?

- If P=NP then N/A.
- If P≠NP then yes. [Theorem]

Few “natural” problems (almost everything in NP is P or NPC)

If we solved such a problem in poly-time, it would just move into P without dragging everything else along.
NP-hard problems

- as hard as any NP problem.
- NPC problems are NP-hard.
- NP-hard need not be NPC
  - might not be decision problems
  - or might not have poly-time verification.

- like NPC, solving an NP-hard problem quickly → same for all NP

NPC = NP-hard & in NP
AN IMPORTANT DETAIL just mentioned here

For NPC problems, we measure input in terms of a finite alphabet (e.g. binary $1 = 1$ bit; $k = \Theta(\log k)$ bits)