Your Chance to Win a Turing Award

- It is generally believed that \( P \neq NP \), i.e. there are problems in NP that are not in P
  - But no one has been able to show even one such problem!
  - This is the fundamental open problem in theoretical computer science
  - Nearly everyone has given up trying to prove it. Instead, theoreticians prove theorems about what follows once we assume \( P \neq NP \)!

Outline: Day 2

- We’ve seen that there are a bunch of problems that seem to be hard.
- Today we’ll see how these problems relate to one another.
- Def: \( P_1 \) is reducible to \( P_2 \) if there is a conversion from an instance \( X \) of \( P_1 \) to an instance \( Y \) of \( P_2 \) such that \( P_1 \) is yes for \( X \) iff \( P_2 \) is yes for \( Y \).

Clique

- Set of vertices so that each edge is "covered"

\[ \text{Vertex cover of size 4} \]

k-Clique problem

- Is there a clique of size \( k \) in the graph?

How are they related?

- How might we reduce clique to vertex cover?
- That is, given a clique problem \((G, k)\), how can we turn it into a vertex cover problem?
- Once we do this reduction, we know we can always solve vertex cover given solution to clique!
Clique to Vertex Cover

- We can reduce Clique to Vertex Cover.
- Given an input \((G, k)\) to Clique:
  - Build graph \(G\) complement
  - Let \(k' = n - k\)
- Vertex Cover is “as hard as” Clique.

If \(G'\) has a \(k'\) cover then \(G\) has a \(k\) Clique:
- Let \(D\) be a cover in \(G'\) of size \(k'\). Then there are no edges in \(V-D\), since otherwise they wouldn’t be covered. Therefore, \(V-D\) is a clique in \(G\).
- Size of clique is \(n-k'\).

TSP

- Travelling Salesman Problem:
  - Given complete weighted graph \(G\), integer \(k\).
  - Is there a cycle that visits all vertices with cost \(\leq k\)?
- One of the canonical problems.
- Note difference from Hamiltonian cycle: graph is complete, and we care about weight.

Hamiltonian Cycle to TSP

- We can reduce Hamiltonian Cycle to TSP.
- Given graph \(G=(V, E)\):
  - Construct complete graph \(G'\) on \(N\) vertices with edge weights: 1 if \((u, v)\) in \(E\), 2 otherwise.
  - Let \(k = N\).
- TSP is “as hard as” Hamiltonian cycle.

Example

If \(G\) has a \(k\) Clique then \(G'\) has a \(k'\) cover:
- Let \(C\) be the clique of size \(k\). Let the cover be \(V-C\). Then clearly every edge outside \(C\) is covered, and in \(G'\) there are no edges in \(C\).
- Size is \(n-k\)
**Proof**

- If G has a Hamiltonian Cycle then G’ has a tour of weight N.
  - The cycle is the tour, since there are N edges of weight 1
- If G’ has a tour of weight N, then G has a Hamiltonian Cycle.
  - The tour is the cycle, since it must contain N edges, each edge must weigh 1, and thus must have been in original graph

**Ham. Cycle to Longest Path**

- Recall, Longest Path: Given directed graph G, start node s, and integer k. Is there a simple path from s of length ≥ k?
- We’ll use Directed Hamiltonian Cycle.

**The reduction**

- Given a directed graph G, want to find Ham. Cycle
  - Convert to Longest path
    - Pick any node as start vertex s.
    - Create a new node t. For every edge (u, s), add an edge (u, t). Let k = N.
- Longest Path is “as hard as” Ham. Cycle

**Proof**

- If G has a Ham. Cycle, then G’ has a path of length k from s.
  - Follow the cycle starting at s, at the last step go to t instead of s.
- If G’ has a path of length k from s, then G has a Ham. Cycle.
  - Path must have hit every node exactly once, and last step in path could have formed cycle in G.

**NP-completeness**

- We’ve seen that there are seemingly hard problems. That’s kind of interesting.
- The really interesting part: A large class of these are equivalent. Solving one would give a solution for all of them!

**More on NP-completeness**

- The pairs I picked weren’t important. There is a large class of problems, called NP-complete, such that any one can be reduced to any other.
- So given an algorithm for any NP-complete problem, all the others can be solved.
- Conversely, if we can prove there is no efficient algorithm for one, then there are no efficient algorithms for any.
NP-Complete Problems

- The “hardest” problems in NP are called NP-complete.
  - If any NP-complete problem is in P, then all of NP is in P.
- Examples:
  - Hamiltonian circuit: find the shortest path that visits all nodes in a weighted graph (okay to repeat edges & nodes).
  - Graph coloring: can the vertices of a graph be colored using \( K \) colors, such that no two adjacent vertices have the same color?
  - Crossword puzzle construction: can a given set of \( 2N \) words, each of length \( N \), be arranged in an \( NxN \) crossword puzzle?

P, NP, and Exponential Time Problems

- All currently known algorithms for NP-complete problems run in exponential worst case time.
  - Finding a polynomial time algorithm for any NPC problem would mean:
- Diagram depicts relationship between P, NP, and EXPTIME (class of problems that provably require exponential time to solve).

Coping with NP-Completeness

1. Settle for algorithms that are fast on average: Worst case still takes exponential time, but doesn’t occur very often.
   - But some NP-Complete problems are also average-time NP-complete!
2. Settle for fast algorithms that give near-optimal solutions: In traveling salesman, may not give the cheapest tour, but maybe good enough.
   - But finding even approximate solutions to some NP-Complete problems is NP-Complete!
3. Just get the exponent as low as possible! Much work on exponential algorithms for satisfiability: in practice can often solve circuits with 1,000+ inputs.
   - But even \( 2^{n/100} \) will eventual hit the exponential curve!