Announcements

Topic 10: P versus NP

- “P” and “NP” are abbreviations of names of classes of a particular type of problem:
  - Decision Problem: problems that can be answered yes/no or true/false

“P” is for “Polynomial Time”

- Except for Bogosort, Hamiltonian Cycles, and Vertex Cover, all algorithms covered this semester have been polynomial time algorithms
- O(1) – O(n^k) are polynomial time

Quick Detour: Deterministic versus Nondeterministic

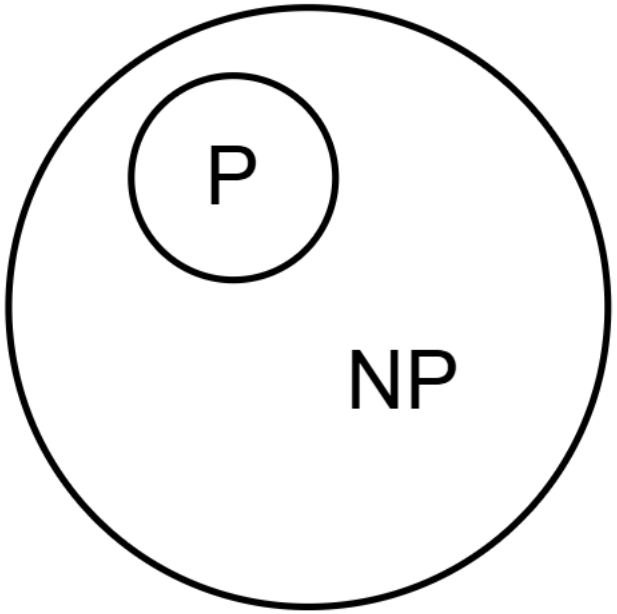
- Standard (non-quantum) computers are deterministic: Actions are determined by the given instructions
- A nondeterministic machine is allowed choices, but always makes the right choice—the one that leads to a correct solution to the given problem
- If one did exist, it could be used to solve “superpolynomial” time problems in polynomial time

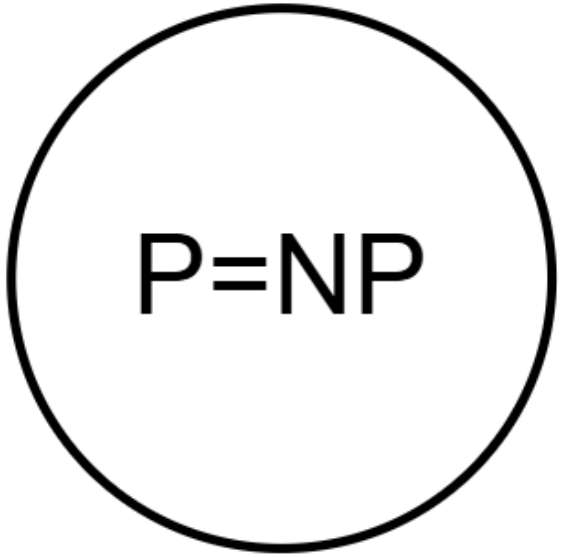
“NP” Stands for…

- Say you have a vertex cover. How hard would it be to verify the cover’s correctness? How would you verify it?
- Not very hard! Iterate through the edges, check to see if an endpoint is in the proposed cover
- If we can check the solution to the problem in polynomial time, the problem is an NP class problem
- NP stands for Non-deterministic Polynomial Time

Relationship Between P and NP

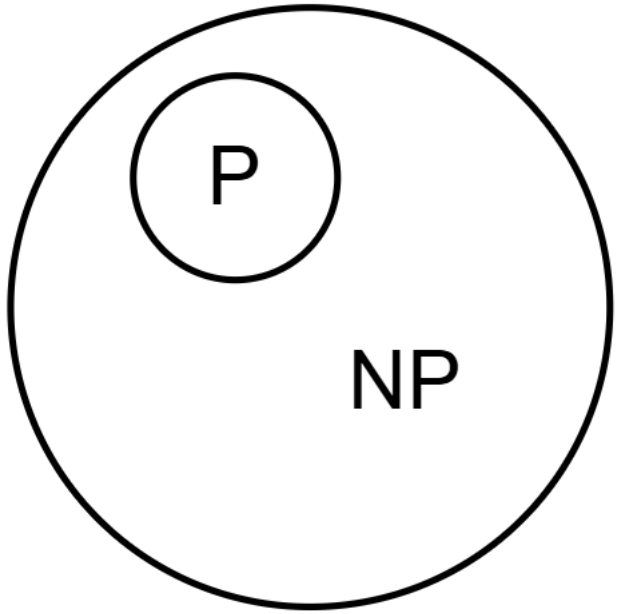
- Clearly, P ⊆ NP
- Example: Bubblesort is O(n^2) and thus is polynomial. We can verify that its output is sorted in O(n) time, showing that it is also in NP.
- But: Is P ⊂ NP?

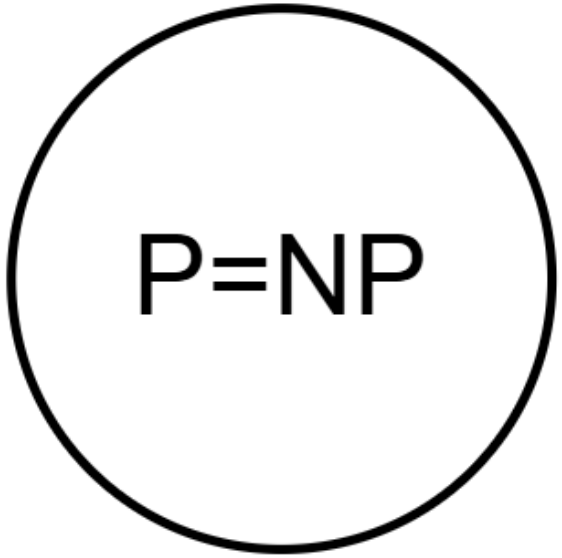




The Big Question

- Does P = NP? In other words, if a problem solution can be verified in P time *must* the problem be solvable in P time?
- This is a Millennium Problem, and considered to be the most important open question in Computer Science





Some Other Big Questions