NP-Completeness

Albert Ye

April 24, 2023

1 P vs. NP

So far, we have seen problems such as polynomial multiplication ($O(n \log n)$), minimum spanning trees ($O((n+m)\log n)$).

These are all **efficiently solvable** problems, meaning they are solvable in at most polynomial time.

> **Definition 1.1** (P). P, which stands for "polynomial", is the **complexity class** of all problems that are efficiently solvable.

> **Definition 1.2** (NP). NP, which stands for "nondeterministic polynomial", is the class of problems whose solutions can be verified efficiently.

The motivation for calling NP NP was not a very good definition and involved something about random algorithms.

> **Example 1.1** (3-coloring). Input: Graph $G = (V, E)(n = |V|)$.
>
> Solution: A coloring $C : V \to \{R, G, B\}$ s.t. each edge receives two different colors of vertex.

Trivial alg: Try all colorings, and there are 3^n total colorings, which is exponential time, which is really bad.

Best alg: 1.3289^n time, after throwing all of our algorithmic toolkit at this time. So this problem is currently exponential time.

Therefore, we don't know if 3-colorings are in P yet. However, we do know that it is in NP. So we'd verify an input and a solution in $O(n^2)$ time by checking every edge and seeing if all edges (u, v) have $color(u) \neq color(v)$.

> **Example 1.2** (Hamiltonian Cycle). Input: Graph $G = (V, E)$
>
> Solution: tour = cycle visiting each vertex exactly once

Trivial alg: Try all $n!$ ways of ordering vartices.

Best known alg: $O(1.657^n)$, which is still exponential.

Yet another problem not known to be in P. However, it is in NP. Just need to show that given a graph, and a supposed ordering of vertices, we can check that it corresponds to a tour.

A slight modification of this problem, the Eulerian Cycle, is in P.

Example 1.3 (Traveling Salesman Problem)**.** Input: Graph $G = (V, E)$ with edge weights

Solution: a tour with minimum total weight.

Best-known alg: Time $O(n^2 2^n)$, the DP that we found in this class.

However, this is probably not in NP because we would need to compute every other tour to see which one actually has the best weight.

We have some modifications that are indeed in NP, such as search-TSP. We want to find a tour with total weight $\leq B$, where the "budget" budget is part of the input.

Things not believed to be in NP:

1. Some optimization versions of problems
2. Counting versions of problems (how many 3-colorings?)
3. Really, really hard problems (halting)

More formally, NP is typically defined for decision problems and search problems. (More in CS172 for decision problems).

Theorem 1.3. P ⊆ NP

The biggest open problem in theoretical CS: is P = NP? Obviously, this is true if $N = 1$ because then we would just have $P = P$.

2 Approximation Algorithms

There is usually a trade-off between efficiency and accuracy, which is where approximation comes in.

Some problems, such as TSP, cannot be approximated in polynomial time. Problems that can have different degrees of accuracy: for example, set cover is $O(\log n)$-approximation.

2.1 Sampling

As the num of samples t increases, the error ϵ and confidence range δ both decrease.

Theorem 2.1 (Chernoff Bound)**.** Suppose $x_1, \ldots, x_t$ are i.i.d. random variables taking values in $\{0, 1\}$. Let $p = E[x_i] \cdot i$. Then, $\forall \epsilon \geq 0$:

$$P[\left|\frac{1}{t}\sum x_i - p\right| \geq \epsilon] \leq 2e^{-2\epsilon^2 t}.$$

We see that this decays exponentially in t, so sampling is powerful.

If we want error $\leq \epsilon$ with probability $> 1 - \delta$ then $t = \lceil \frac{1}{2\epsilon^2} \log\left(\frac{2}{\delta}\right) \rceil$. There is no n, so the input size n is completely independent of the sample size.

So why is it so difficult to get an accurate result? Like why can't we predict the election? Because it's difficult to get a **uniformly random sample**.

2.2 (s)Treaming

INPUT: A sequence $\{z_1, z_2, \dots, z_n\}$ of numbers $\{1, \dots, M\}$ that ”just streams by”. The entire input is not stored, and input is seen exactly once.

Memory is polynomial in terms of $\log M, \log N$.