**NP Theory Notes – Office Hours (10/25/2023)**

For a given input size n:

* Polynomial Time: O(n^k) for some constant k (efficient)
* Exponent Time: O(k^n) for some k > 1
* P == Polynomial Time
* NP == Nondeterministic Polynomial Time

The Class of Problems

* The class P – a solution may be found in polynomial time (easy)
* The class NP – a solution may be verified in polynomial time
* NP-Hard – we are yet to find a polynomial-time solution
* NP-Complete – both in NP and at least as hard as NP-Hard problems
* A problem is either known or not yet known to be in P or NP

A black background with a black square

Description automatically generated with medium confidence

NP-Complete Proofs

* These are forms of reductions
* Prove that unknown Problem B is NP-Complete by showing that it could be used to solve an instance I of A, a known NP-Complete problem
* You define f and h in your proof – these are the input and output transformations
* We reduce A to B to show that B is at least as hard as A
  + If B can solve A, then A is as easy as B
  + Reducing B to A just shows that we can make an easy problem hard

NP Reductions – The Steps

*We have a Problem B and we want to show that it’s NP-Complete*

1. Demonstrate that problem B is in the class of NP problems
2. Demonstrate that problem B is at least as hard as a problem known to be NP-Complete. This is done using reduction from a known problem
   1. Show how an instance of A is converted to B in polynomial time
   2. Show how a solution to B can be converted to a solution for A in polynomial time
   3. Show that a solution for B exists if-and-only-if (IFF) a solution to A exists

Goal vs Optimization Problems

With NP Proofs, we generally work with goal versions of problems

* Find an Independent Set of at least k vertices
* Find a Vertex Cover of at most k vertices
* Find a subset of items of weight less than or qual to B and total value greater than or equal to g

Rather than optimization versions

* Find the largest Independent Set
* Find the smallest Vertex Cover
* Find the subset of objects with weight less than or equal to B which maximizes the total value

Common Mistakes

* Reducing in the wrong direction (B to A)
* Designing a solution to the unknown problem
* Failing to transform the output from B to a solution for A
* Failing to prove both direction in IFF
* Failing to provide runtimes in Big O notation (any polynomial runtime is acceptable)
  + Using runtimes based on the target or goal variables
* Using online sources without citation; copying online sources