NP Completeness and Computational Intractability

Written by Gareth Antle 2019 S2

Important Definitions

Decision Problems

A decision problem is a yes or no question on a infinite set of inputs. For example "Given two positive integers x and y, does x evenly divide y?" An algorithm that correctly solves this problem will return YES if x evenly divides y and NO otherwise.

Certificates

A certificate is a potential solution to a decision problem. You can think of it as a "valid solution" that might be generated by a brute force algorithm. The size of a certificate is usually expressed in terms of the size of the input. For example a certificate to the SMP problem would be a list of pairs of men and women, and would be of size $\Theta(n)$.

A certificate can be verified in polynomial time if we can determine whether or not a particular certificate answers the question of the decision problem in polynomial time.

P

The class P is the class of all decision problems that can be solved efficiently, which is to say in polynomial time.

NP

The class NP is the class of all decision problems which can be verified in polynomial time. Since any problem that can be solved in polynomial time can be verified in polynomial time, the class NP contains the class P.

NP-Hard

Problems in the class NP-Hard are at least as hard as the hardest problem in NP. Some problems are even harder. For instance the Turing halting problem is NP-Hard but is too hard to be considered NP.

NP Complete

A decision problem p is NP-Complete if it is in NP and every problem in NP is reducible to p in polynomial time. An equivalent definition is that a problem is NP Complete if it is in NP and is in NP Hard.

In order to prove that p is in NP, we need to show that it has a polynomial length certificate which can be verified in polynomial time. We can show that a problem p is NP-Hard by reducing from a problem we know to be NP Hard to p, and showing that this reduction runs in polynomial time in terms of the size of p.

Its important to keep the direction of this reduction in mind. When proving NP Completeness we reduce from a known NP Complete problem to the problem whose NP Completeness we are trying to prove.

A note about reductions: When we say we reduce problem A to problem B that does not imply that we are making the problem easier to solve or smaller. In fact, reductions in this context are used to show that problems are very hard. We are reducing problems, but not reducing hardness.

Classical NP Complete Problems

SAT

Given a Boolean expression composed of n variables $x_1, ..., x_n$ with the Boolean connectives AND, OR, and NOT, along with parentheses, the answer to the Satisfiability decision problem is YES if the expression is satisfiable. That is; if there exists an assignment of truth values to the variables which makes the entire expression evaluate to TRUE.

Packing Problems

Independent Set

Given a Graph $G = (V, E)$ and an integer k the answer to the Independent Set decision problem is YES if there is a set of nodes with size k in V where no two nodes are adjacent to each other.

Covering Problems

Vertex Cover

Given a Graph $G = (V, E)$ and an integer k the answer to the Vertex Cover decision problem is YES if there is a set S of nodes with size k such that every edge in E is incident on one of the nodes in S.

Sequencing Problems

Hamiltonian Path/Cycle

A Halmiltonian Path is a path that visits each node in a graph exactly once. A Hamiltonian Cycle is a cycle that visits each node in the graph exactly once. Given a Graph $G = (V, E)$ the answer to the Hamiltonian Path or Cycle decision problem is yes iff G contains a Hamiltonian Path or Cycle respectively.

Partitioning Problems

Graph Coloring

The answer to the Graph Coloring decision problem is YES if given a Graph $G = (V, E)$ and an integer k a the nodes in V can be assigned on of k colors such that no adjacent nodes are assigned the same color.

Numerical Problems

Subset Sum

The answer to the Subset Sum Decision problem is YES iff, given a set of integers S and an integer k, there is a set $N \subseteq S$ that sums to k

Lets prove some NP Completeness!

1. Lets consider the Vertex Cover problem, which is as follows:

   Recall that a vertex cover is a subset of nodes $S \subseteq V$ where each edge in E is incident on at least one node in S

   INSTANCE: A Graph $G = (V, E)$
   OUTPUT: A vertex cover S and an integer $k = |S|$

   Can we prove that this Vertex Cover problem is NP Complete? If not, how would we need to alter the problem so that we could prove its membership in the classes NP and NP Hard?

   SOLUTION: The VC problem as stated above is not a decision problem, so we cannot reason about its membership in NP. A decision problem version of the VC problem would be:

   INSTANCE: A Graph $G = (V, E)$ and an integer k
   OUTPUT: YES if there is a Vertex Cover in G of size k and NO otherwise

2. Now that we've altered the problem into a form where we can better reason about its complexity, lets prove that the Vertex Cover problem is in NP. In order to do this we need to define a certificate for the Vertex Cover problem. (A certificate should contain all the information you need to decide if it represents a valid and good solution)

   SOLUTION: A certificate for the Vertex Cover problem would be a set $S \subseteq V$ where $|S| = k$

3. In order for the Vertex Cover problem to be in the set NP, it must have a polynomial length certificate and that certificate must be able to check whether or not the certificate is a good solution in polynomial time. Briefly justify that the certificate you have described above fulfills this criteria.

   SOLUTION: If we have a graph where $|V| = n$ and $|E| = m$ then the certificate is obviously of size polynomial in n since it has size at most n. Now we need to show that this certificate can be verified in polynomial time. We can verify a solution by taking each node in S and removing all edges incident on S from E. If after k iterations both S and E are empty then we return YES, otherwise we return NO. Since $|S| \leq n$ and $|E| = m$ this algorithm runs in $O(n + m)$ time.

4. Now that we have shown that the Vertex Cover problem is in NP, next we need to show that it is NP-Hard. We do this by reducing from a problem we know is NP Complete to the Vertex Cover problem. In theory we can reduce from any NP Complete problem, but in practice we want to look for a problem that is similar in some way to the Vertex Cover problem. Brainstorm some similar NP complete problems using the list of classical NP complete problems at the start of this worksheet.

I would suggest using the Independent Set problem, though if you'd like a challenge you can try reducing from a different problem.
In our reduction we need to show that the answer to the Vertex Cover problem is YES iff the answer to the Independent set problem is YES. To prove this biconditional, we need to prove two conditionals:

(a) If the answer to the instance of the Independent Set problem is yes, then the answer to the Vertex Cover produced by our reduction is guaranteed to be YES. Independent Set $\rightarrow$ Vertex Cover

(b) If the answer to the Vertex Cover problem produced by our reduction is YES then the answer to the instance of the Independent set problem must have been yes. Vertex Cover $\rightarrow$ Independent Set

Lets start with describing the actual reduction from Independent Set to Vertex Cover and showing that this can be done in polynomial time:

SOLUTION: We can reduce an instance of the Independent Set $G = (V, E), k$ problem to an instance of the Vertex Cover problem $G' = (V', E'), k'$ by setting $G' = G$ and $k' = |V| - k$

Now lets prove the correctness of the the "forward" direction of the reduction, going from Independent Set $\rightarrow$ Vertex Cover:

SOLUTION: Given a graph $G = (V, E)$ suppose $S \subseteq V$ is an Independent Set of size k, and let $e = (u, v)$ be an arbitrary edge. Only one of u, v can be in S. Hence at least one of u, v is in $V - S$. So for every edge (u, v) there will be a node in $V - S$. This is exactly the definition of a vertex cover, so $V - S$ is a vertex cover of size $n - k$.

5. And now the "reverse" direction of the reduction, from our special case of the Vertex Cover $\rightarrow$ Independent Set

   SOLUTION: Given a graph $G = (V, E)$ suppose $V - S$ is a vertex cover of size k and let $u, v \in S$. There cannot be an edge (u, v), otherwise that edge wouldnt be covered by the vertex cover $V - S$. This is exactly the defintion of an independent set, so S is an independent set and $k + |S| = n$.

This completes our proof of the NP Completeness of the Vertex Cover problem!