NP-complete Reductions

1. Prove that 3SAT $\leq_P$ DOUBLE-SAT, i.e., show DOUBLE-SAT is NP-complete by reduction from 3SAT.

   The 3-SAT problem consists of a conjunction of clauses over *n* Boolean variables, where each clause is a disjunction of 3 literals, e.g., $(x_1 \vee \neg x_3 \vee \neg x_5) \wedge (x_2 \vee x_4 \vee \neg x_6) \wedge (\neg x_3 \vee \neg x_5 \vee x_6)$. The DOUBLE-SAT problem takes as input a Boolean formula *f*, and asks if there are two satisfying assignments for *f*.

   Solution:
   DOUBLE -SAT, like any variant of SAT, is in NP since the truth assignment is the certificate; we can check every clause in polynomial time to see if it is satisfied.

   We now show a reduction from 3-SAT. Given a 3-SAT formula *f*, we construct a DOUBLE-SAT formula $f' = f \wedge (x_k \vee \neg x_k)$, where x_k is a variable not used in *f*. Clearly this reduction is poly-time.

   We now show that *f* has a satisfying assignment iff *f'* has two satisfying assignments.

   $\Rightarrow$ If *f* has a satisfying assignment σ, then *f'* has two satisfying assignments, corresponding to $(\sigma, x_k = \text{true})$ and to $(\sigma, x_k = \text{false})$.

   $\Leftarrow$ The clause $(x_k \vee \neg x_k)$ can be satisfied with either x_k =true or x_k =false. In either case, if we have a satisfying assignment σ for *f*, this ensures DOUBLE-SAT is validated. In this case, then 3-SAT is also validated.

2. Prove that Hamilton Cycle $\leq_P$ Directed-HC, i.e., show Directed-HC is NP-complete by reduction from Hamilton Cycle . The Hamilton Cycle (HC) problem asks if there is a simple cycle that visits every node once in an undirected graph G(V,E).

   INSTANCE: For Directed-HC we are given as input a directed graph G'(V,E),

   QUESTION: is a simple directed cycle that visits every node once?

   Solution:
   We first must show that DHC is in NP: Given a certificate (path in G) we can check in poly-time if this is Hamiltonian.

   We now demonstrate how to reduce HC to DHC. Let G=(V,E) be the input for HC. We create a DHC graph G'=(V,E') by a reduction operator T that takes each edge in G and replaces it by two edges in E' (one going in each direction). (This is just the standard way in which an undirected graph is converted into a directed graph.) Clearly this reduction takes poly-time.

   We now show that HC has a yes answer iff DHC has a yes answer.
   $\Rightarrow$: Assume G has a Hamilton circuit. By our construction, we must have a directed Hamilton circuit in *G'*.

⇐ Assume *G'* has a directed Hamilton circuit. By our construction, we must have a Hamilton circuit in G.

3. Prove that SubsetSum $\leq_P$ Subset-Equality.

SubSet Sum: given a set S of n non-negative integers, does there exist a subset of *k* of the integers that sums to a value *c?*

SubSet Equality: given a set S of n non-negative integers, does there exist a partition of S into X and Y such that the sum of the integers in X equals the sum of the integers in Y?

Solution:

SubSet Equality is a restriction of SubSet Sum to the case where c $= \frac{1}{2}\sum_{x \in S} x$, leading to a partition of S in X and Y, each with sum of c.

4. [HARD] Prove that SAT$\leq_P$ NAE-SAT.

Not-All-Equal Satisfiability (NAE-SAT) is like SAT, but must be such that every clause contains at least one true literal and at least one false one. NAE-3-SAT is the special case where each clause has exactly 3 literals. Show that NAE-3-SAT is NP-complete by reducing 3-SAT to it.

Solution:

NAE-3-SAT, like any variant of SAT, is in NP since the truth assignment is the certificate; we can check every clause in polynomial time to see if it is satisfied.

We now show a reduction from 3-SAT. We define a single "reference variable" *z* for the entire NAE-SAT formula. Then we represent each variable x_i in the 3-SAT formula *f* with a variable y_i in the NAE-SAT formula *f'*, where x_i is true or false if $y_i \neq z$ or $y_i = z$ respectively. Then we can represent a 3-SAT clause $(x_i \vee x_j \vee x_k)$ with a NAE-4-SAT clause (y_i , y_j , y_k , z), since this clause is satisfied if and only if at least one of y's is different from z, i.e., if at least one of the x's is true. We can similarly represent $\neg x_i$ by $\neg y_i$.

This shows that NAE-4-SAT is NP-complete. To reduce this to NAE-3-SAT, we use variables w to break each 4-variable clause into two 3-variable ones, just as for 3-SAT:

$(y_i , y_j , y_k , z) = (y_i , y_j , w) \wedge (\neg w, y_k , z)$
It is easy to check that this works, i.e., by setting w appropriately we can satisfy these two NAE-3-SAT clauses unless $y_i = y_j = y_k = z$.
□

Alternate solution. Represent each x_i with a pair of variables y_i , z_i , where x_i is true if $y_i \neq z_i$; that is, $x_i = y_i \oplus z_i$. Then the 3-SAT clause $(x_i \vee x_j \vee x_k)$ becomes a set of four NAE-6-SAT clauses,
$(y_i , z_i , y_j , z_j , y_k , z_k)$

$(y_i , z_i , y_j , z_j , \neg y_k , \neg z_k)$
$(y_i , z_i , \neg y_j , \neg z_j, y_k , z_k)$
$(y_i , z_i , \neg y_j , \neg z_j, y_k , z_k)$
and we break these down into NAE-3-SAT clauses as above. If x_i , say, is negated, we flip z_i in each of these clauses.

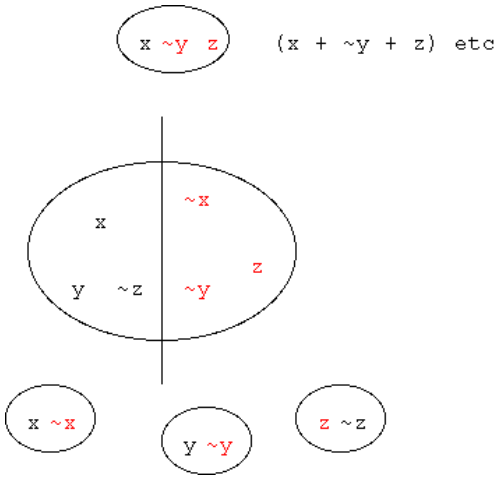
5. Show NAE-3-SAT$\leq_p$ Set-Splitting, i.e., that Set-Splitting is NP-complete by reducing NAE-3-SAT to it. Consider the Set-Splitting problem: given a family *F* of subsets of a finite set *S*, decide whether there exists a partition of *S* into two subsets S_1, S_2 such that all elements of *F* are split by this partition, i.e., none of the elements of *F* is completely in S_1 or S_2.

<u>Solution:</u>
Set-Splitting is in NP since the certificate can be checked in polynomial time to see if it is satisfied.

We now show a reduction from NAE-3-SAT. The input to NAE-3-SAT is a Boolean function μ of *m* clauses $\{C_1,\ldots,C_m\}$ over *n* variables $\{x_1,\ldots,x_n\}$. We create a Set-Splitting instance (F, S, S_1, S_2) as follows.

Make the set S contain all the variables in the NAE-3-SAT formula and their negations. The family *F* of subsets of a finite set *S* consists of two sub-collections: (1) for each variable-negation pair ((x, ¬x), (y, ¬y) etc) make a subset in collection 1; (2) for each each clause (e.g. $(x_i \vee x_j \vee x_k)$) make a subset in collection 2.

We now show that the set S can be split in the required way if and only if the formula μ has a truth-value assignment that makes it true.



We must show that there exists a partition of *S* into two subsets S_1, S_2 such that all elements of *F* are split by this partition One part of the split set corresponds to variables in μ that are"true", and the other "false". Each subset must contain at least one "true" variable and one "false" variable, which is exactly what the NAE-3-SAT problem requires.

6. Prove that *Dense Subgraph* is NP-complete.

Dense Subgraph: Given a graph G and two integers *a* and *b*, does G have a set of *a* vertices with at least *b* edges between them?

Solution:

Dense SubGraph can be restricted to the CLIQUE problem by specifying that b=1/2 a(a-1), i.e., that the subgraph be a clique (fully-connected).

7. Prove that **Vertex Cover $\leq_P$ Set Cover**

The Set Cover Problem:

Instance: A set $X = \{x_1, x_2, \ldots, x_n\}$ and a collection $S = \{S_1, S_2, \ldots, S_m\}$ of subsets of X such that S covers X, i.e., $S_1 \cup S_2 \cup \ldots \cup S_m = X$.

A **cover set** is any sub-collection $C \subseteq S$ that also covers X.

Question: Given $\langle X, S, K \rangle$, is there a cover set C of size $\leq K$?

Solution:

First, we argue that Set Cover is in NP, since given a collection of sets C, a certifier can efficiently check that C indeed contains at most k elements, and that the union of all sets listed in C does include all elements from the ground set X.

We will now show that Vertex Cover $\leq_P$ Set Cover. Given an instance of Vertex Cover (i.e. a graph G = (V,E) and an integer j), we will construct an instance of the Set Cover problem. Let X = E. We will define n subsets of X as follows: label the vertices of G from 1 to n, and let Si be the set of edges that incident to vertex i (the edges for which vertex i is an end-point). Note that $Si \subseteq X$ for all i. Finally let k = j. This construction can be done in time that is polynomial in the size of the Vertex Cover instance. We now run our black box for the Set Cover problem and return the same result it gives.

To prove that this answer is correct, we simply need to show that the original instance of Vertex Cover is a yes instance *if and only if* the Set Cover instance we created is also a yes instance.

$\Rightarrow$ Suppose G has a vertex cover of size at most j. Let S be such a set of nodes. By our construction, S corresponds to a collection C of subsets of X. Since we defined k = j, C clearly has at most k subsets. Furthermore, we claim that the sets listed in C cover X. To see this, consider any element of X. Such an element is an edge e in G, and since S is a vertex cover for G, at least one of es endpoints is in S. Therefore C contains at least one of the sets associated with the endpoints of e, and by definition, these both contain e.

$\Leftarrow$ Now suppose there is a set cover C of size k in our constructed instance. Since each set in C is naturally associated with a vertex in G, let S be the set of these vertices. |S| = |C| and thus S contains at most j nodes. Furthermore, consider any edge e. Since e is in the ground set X and C is a set cover, C must contain at least one set that includes e. But by our construction, the only sets that include e correspond to nodes that are endpoints of e. Thus, S must contain at least one of the endpoints of e.

We have now shown that our algorithm solves the Vertex Cover problem using a black box for the Set Cover problem. Since our construction takes polynomial time, and we have shown that Set Cover is in NP, we can conclude that Set Cover is NP-Complete.

8. Prove that **Hamiltonian Cycle $\leq_P$ TSP**

The Traveling Salesman Problem:

Instance: An n×n weighted adjacency matrix D = (d_{ij}) of a **complete** weighted graph on n vertices with integer edge lengths d_{ij} , and an integer K. **Feasible Sol:** A TSP tour is any Hamiltonian cycle in this complete graph.

Question: Is there a TSP tour of length (or distance) ≤ K?

Solution:

(i) It is simple to show that TSP∈ NP. Given a certificate for TSP, we can show it is correct or not in time $O(E^2)$.

(ii) The reduction is as follows. We assume we are input a Hamilton Cycle instance of a graph G(V,E), and must construct a TSP instance G'(V',E') together with a weight function w(E').

Given a graph *G* with *n* vertices construct a new weighted complete graph *G'* with the same vertices (V=V'), and we add edges E* such that E'=E∪E*. We create the weight function such that the weight of the edge (*v,u*) has weight 1 if the edge (*v,u*) is in the original graph *G*, and the edge has weight *2* otherwise. Clearly, this reduction can be done in time polynomial in G(V,E).

We claim that *G* has a Hamilton Cycle iff there is a Hamilton Cycle in *G'* with a total weight of *n*.

⇒ Suppose G has a Hamilton Cycle; by our construction this must mean that G' has a TSP tour of cost n.

⇐ Now suppose that G' has a TSP tour of cost n. By our construction the must mean that G has a Hamilton Cycle, since only edges in G have cost 1 in G'.

The Hamilton Cycle problem reduces to the Traveling Salesman Problem, therefore the Traveling Salesman Problem is NP-hard. Since TSP is also in NP, we have thus shown it is NP-complete.