1 Study Group

none YES

2 NP Basics

If A reduces to B, we know B can be used to solve A, which means B is at least as hard as A.

(a) B can be anything {P,NP,NPH}

(b) A is in P

(c) B in NP-hard

(d) A can be anything {P,NP,NPH}

3 Runtime of NP

True

By definition, NPC is the intersection of NP and NPH.

Since any problem in NP can be reduced to NPC, any problem in NP is at most as hard as NPC problem.

Thus, every problem in NP has a O(n^k) time algorithm

4 Dominating Set

We can prove Dominating Set as NPC by showing that the vertex cover can be reduced by dominating set

Given a graph G(V,E) and a number k, we define Vertex Cover = {<G,k>: there exists a vertex cover of G with size at most k}

For (u,v) in E, we add a new vertex a and new edges (u,a) and (a,v) to G. Denote the new graph G'(V',E')

1)If G' has a dominating set of size k

Notice that we create a triangle for u,v and a if (u,v) is in E. By definition, at least one of the 3 vertex belongs to D. The new vertex we added only adjacent to u and v. Thus, if the dominating set contains a, we can switch that vertex with u or v. Also notice that if we have u or v in the dominating set, there is no need to add a into the set. Thus, we can assume that there is no a in the dominating set without making the set size larger. Thus, the dominating set contains u or v for (u,v) in E. So it's a vertex cover in G.

2)If G has a vertex cover of size at most k

It's clear that the vertex cover for G is the dominating set for G'. Because for every edge (u,v) in E, u or v belongs to D and a is connected to u and v.

5 More Reductions

(a)Let (L,t) be an instance of subset's sum where L holds a subset of A and B is the target sum.

Let m be the sum of elements in L. Let L' be the set L + a(n+1) + a(n+2) where a(n+1) = 2m + t, a(n+2) = 3m-t.

Then L' is a partition iff (L,t) is a subset sum.

If (L,t) is a subset sum, there exist B which is a subset of L with sum t. Then (L-B)U{a(n+1)} and BU{a(n+2)} is a partition

If L' is a partition, there exists a partition ($L1$,$L2$) . Since the total sum is 6m, a(n+1) and a(n+2) should not appear in the same set. WLOG, let a(n+1) in $L1$ and a(n+2) in $L2$. Then $L2$ - {a(n+2)} has sum k

Notice that we L' can be constructed in linear time. Thus, the reduction is linear

(b)Set w_i = v_i = a_i i in {1,2,...,n} and W = V = t which is the target sum.

Then P is the subset of the modified knapsack iff P is a subset sum with t

If P is the subset of knapsack, then we know that the sum of elements in P is less or equal then t from weight constraint and the sum of elements in P is greater or equal then t from value constraint. Thus, the sum of elements in P is t which is a subset sum with t.

If P is the subset sum with t, then P is also the subset of knapsack problem since k<=k , k>=k

6 Orthogonal Vectors

First, let 0 represents true and 1 represents false

For a 3-SAT problem with n variables and m clauses, we divide n variables into 2 parts A = {x_1,..,x_n/2}

B = {x_n/2+1,...,x_n}. There are 2^(n/2) permutations for A and B respectively.

Let's discuss about a specific permutation. Suppose we have a bit string of length n.

Then construct the dot product [ ]*[ ] (Denote the 2 bracket to be A' and B'). Each multiplier has m rows. We want the result to be a zero vector to make it satisfiable.

Suppose the i-th clause looks like (x_pVx_qVx_r).

If x_p, x_q and x_r are all in A', then the i-th entry for A' is the result of x_pVx_qVx_r while i-th entry of B' is 0

If one of them has 2 variables, WLOG, let's say x_p and x_q belongs to A' and x_r belongs to B'. The i-th entry of A' is the result of x_pVx_q while the i-th entry of B' is x_r.

Construct all i in {1,2,...,m}. Compute the product and check whether it's a zero vector

Notice that we assume all vectors in A, B are in {0,1}^m and |A| = |B| = n with O(n^c*m)-time to find the orthogonal product.

Since we assume the 3-SAT problem has n variables and m clauses, we need 2^n/2 permutations. Thus, the 3-SAT takes time O((2^n/2)^c*m)