CS 600 HW 9
SHUCHI PARAGBHAI MEHTA
CWID: 20009083

- 17.8.8

Given the CNF formula $B = (x_1) \cdot (\overline{x_2} + x_3 + x_5 + \overline{x_6}) \cdot (x_1 + x_4) \cdot (x_3 + \overline{x_5})$, show the reduction of B into an equivalent input for the 3SAT problem.

(x1+b+c)· (x1+¬ b +c)· (x1+b+¬ c)· (x1+¬ b+¬ c)·(x2+ x3+c1)· (¬c1+ x5+c2)· (¬c2+ x6+c3)· (¬c3+ x5+ x6) · (x1+x4+d)· (x1+x4+¬ d)· (x3+x5+e)· (x3+ x5+ ¬ e)

- 17.8.12

Professor Amongus has just designed an algorithm that can take any graph G with n vertices and determine in $O(n^k)$ time whether G contains a clique of size k. Does Professor Amongus deserve the Turing Award for having just shown that $\boldsymbol{P} = \boldsymbol{NP}$? Why or why not?

The clique problem is NP complete. Reduction from vertex-cover.

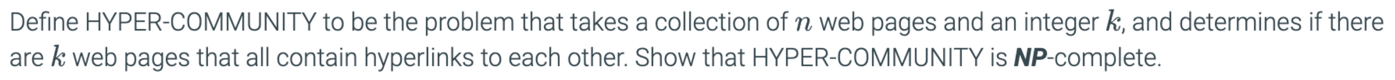
Input: G= (V, E) and integer k>0

a. Non-deterministically select a subset C of nodes of G.
b. Test whether all nodes in c are connected and whether G contains all edges connecting nodes in C.
c. If yes, accept
d. Else reject

Every NP-related task must be solvable in polynomial time in order to prove P = NP. It must be solvable in polynomial time for any value of k. K does not have a defined value.

Because there are infinitely many problems in NP and it is realistically impossible to justify polynomial time for each problem, Professor Amongus does not deserve the Turing Award for having just demonstrated that P = NP.

- 17.8.28


Define HYPER-COMMUNITY to be the problem that takes a collection of n web pages and an integer k, and determines if there are k web pages that all contain hyperlinks to each other. Show that HYPER-COMMUNITY is ***NP***-complete.

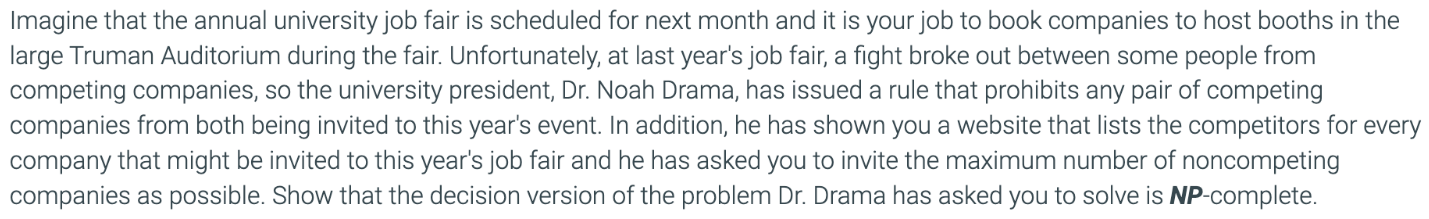
Keeping in mind that HYPER-COMMUNITY is in NP Because a non-deterministic machine might easily predict k web pages and determine whether or not they are all connected.

HYPER-COMMUNITY is now being subtracted from INDEPENDENT-SET. Find an independent set f of size k for a graph G with n vertices, for example. Now imagine another graph, G', with the same number of nodes as G.
If and only if this is not present in G, graph G' contains the edge (u, v). Therefore, iterating through all the pairs of vertices will require polynomial time to complete.

All k vertices are connected in G' if there is an independent set of size k in G, and k vertices also make up an independent set in G if there is a set of k mutually connected vertices in G'.

Since HYPER-COMMUNITY can be reduced to INDEPENDENT-SET. Thus, it is NP-complete.

- 17.8.35


Imagine that the annual university job fair is scheduled for next month and it is your job to book companies to host booths in the large Truman Auditorium during the fair. Unfortunately, at last year's job fair, a fight broke out between some people from competing companies, so the university president, Dr. Noah Drama, has issued a rule that prohibits any pair of competing companies from both being invited to this year's event. In addition, he has shown you a website that lists the competitors for every company that might be invited to this year's job fair and he has asked you to invite the maximum number of noncompeting companies as possible. Show that the decision version of the problem Dr. Drama has asked you to solve is ***NP***-complete.

Vertex cover is the problem mentioned above. A graph G(V, E) and a positive integer k serve as instance of vertex cover problems. Now, to check to see if it is NP-complete or not.

For a problem to be NP-complete, it must satisfy the two condition:
1. Proof that the problem is in NP.
2. Proof that the problem is NP-Hard

Proof that the problem is in NP.
1. Pick a subset W of size k non-deterministically.
2. Remove all edges from set X that are adjacent to each vertex v in W in order to test this. If the set X gets empty and the edges removed equal k.
3. If yes, accept
4. Else reject

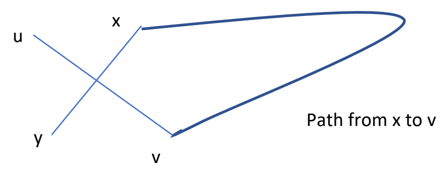
As above algorithm can be done in polynomial time thus it is in NP.

Proof that the problem is NP-Hard
1. Reduce 3SAT to VERTEX-COVER for NP-Hard.
2. Let S represent the CNF's Boolean formula, with 3 literals in each clause.
3. Create a graph G and a number k such that G has a vertex cover if and only if S is satisfiable.
4. Vertex cover must include one of the two vertices in accordance with the "truth setting" component and the "satisfaction setting" component.

According to theorem 17.9 given in the textbook Vertex-Cover is NP complete.

- 18.6.19

In the ***Euclidean*** traveling salesperson problem, cities are points in the plane and the distance between two cities is the Euclidean distance between the points for these cities, that is, the length of the straight line joining these points. Show that an optimal solution to the Euclidean TSP is a simple polygon, that is, a connected sequence of line segments such that no two ever cross.



Assume that two edges cross each other at (x, y) (u, v) as indicated in the figure. The best route is now determined by the path P, which went from y to x, then from x to v, and finally from v to u.

Taking into account a different route P' from y to v (in a straight line), v to x, and x to u (straight line). By proving that path P' will lead to the Euclidean TSP's optimal solution.

Let's assume that there is a location d at which the two edges (x, y) cross (u, v). Consider the distance between w and z to be d(w,z). Currently, using the Euclidean formula,

$$d(u, v) + d(x, y) = d(y, d) + d(d, x) + d(u, d) + d(d, v)$$

By triangle inequality,

$$d(y, d) + d(d, v) >= d(y, v) \text{ and}$$

$$d(u, d) + d(d, x) >= d(u, x)$$

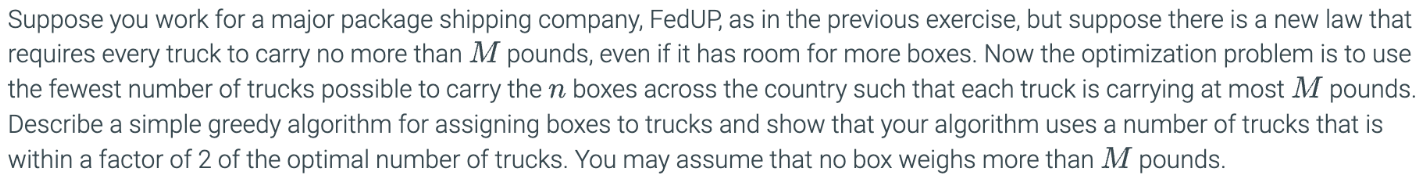
Now, above equality gives,

$$d(x, y) + d(u, v) >= d(y, v) + d(u, x)$$

The overall distance traveled by Path P' is less than that of the original Path P. Our presumption is incompatible with the condition described above, hence no optimal path can have crossing edges.

P' is therefore the optimal solution to the Euclidean TSP is a simple polygon, that is, a connected sequence of line segments such that no two ever cross.

- 18.6.26



Suppose you work for a major package shipping company, FedUP, as in the previous exercise, but suppose there is a new law that requires every truck to carry no more than M pounds, even if it has room for more boxes. Now the optimization problem is to use the fewest number of trucks possible to carry the n boxes across the country such that each truck is carrying at most M pounds. Describe a simple greedy algorithm for assigning boxes to trucks and show that your algorithm uses a number of trucks that is within a factor of 2 of the optimal number of trucks. You may assume that no box weighs more than M pounds.

Given that there are n boxes, each weighing $w_1, w_2, \ldots, w_n$, let N be the optimal number of vehicles. Maximum weight per truck is M pounds.

We can say that

$$NM \geq \sum_{i=1}^{n} w_i$$

$$N \geq \frac{1}{M} \sum_{i=1}^{n} w_i = X$$

Here, X lists the bare minimum of trucks needed. Start placing weights in the truck one at a time while making sure the weight does not exceed M pounds using the next fit greedy technique. If M ponds are in one truck, choose the other in the next step. By following this process, every second truck contains weights that total up to more than 2M.

The minimal number of trucks needed to solve the issue will now be lower than 2M. A better approach is to order the weights such that they can fit into the truck most efficiently by sorting them from largest to smallest. Checking all of the currently filled vehicles to see whether any weight can be added; if not, finding a new truck to handle the weight.

The given algorithm will run in $O(n^2)$ time, where n is the number of boxes.