CSCI 8221 – Advanced Algorithm

Homework Solutions

1. **3-Colorability $\leq_p$ One-In-3SAT problem**.

   Idea: A common theme in reducing a graph problem into SAT problem is to find a way to represent an instance of graph in Boolean expression. In order to do that, we need to find the properties and constraints of the graph and then express them using Boolean expressions.

   Transformation:

   (1) Take an arbitrary instance of a 3-colorability problem. It's a graph G. If G is 3-colorable, using three different colors, say Red, Green and Blue, we can color vertices of G so that not any two adjacent vertices are the same color.

   (2) For any vertex V_i, it can be colored in either R, B or G. In the language of Boolean expression we have: $V_{iR} + V_{iB} + V_{iG}$

   For any edge $E(V_i, V_j)$ between V_i and V_j, the constraint is if V_i is colored in any of the three colors given, then V_j cannot be colored using that same color. Hence, we have the following One-In-3SAT expression for each edge:

   $$(V_{iR} + V_{jR} + a)\ (V_{iB} + V_{jB} + b)\ (V_{iG} + V_{jG} + c)(a + b + c)$$

   Let W be the 3CNF consisting of all the clauses above. We need |V| clauses for all vertices, one for each vertex, and 4*|E| clauses for edges, 4 clauses for each edge. Hence, this transformation is polynomial.

   Claim:

   G is in 3-colorability iff W is in One-In-3SAT

   Proof:

   = >

   Assume each vertex of the G can be assigned one of the 3 colors such that not any two adjacent vertices are the same color. If vertex V_i is colored red, we assigned V_{iR} = true, $V_{iB} = V_{iG}$ = false. It's easy to see that it holds true for the clause corresponding to each vertex. Now, let's look at the clauses corresponding to each edge $e(V_i, V_j)$. Say for

example, V_i is colored in blue, V_j is colored in green. We can pick a= true, and b=c=false, then the clauses corresponding to e(V_i, V_j) will be satisfied.

< =

Assume there is a true-false assignment for W that makes it satisfied and exactly only one variable in each clause of W is true. If V_{iR} (or V_{iG}/ V_{iB}) is true we color vertex V_i of G in red (or green/blue correspondingly).

Look at each clause of W corresponding to each vertex in G. Since there is only 1 variable is assigned true in each of these clauses, there is only 1 color assignment for each vertex in G.

Look at any clauses of W corresponding to any edge in G.

$(V_{iR} + V_{jR} + a)\ (V_{iB} + V_{jB} + b)\ (V_{iG} + V_{jG} + c)(a + b + c)$

We have just proved that exactly 1 value out of V_{iR}, V_{iG}, V_{iB} can be true, and exactly 1 value out of V_{iR}, V_{iG}, V_{iB} can be true. Without lost of generality, assume, V_{iR} = Vj_B = true. (Note that V_{jR} cannot be true if V_{iR} is already true because they are in the same clause.)

That means, a and b must be false. Hence c must be true. Consequently, V_{iG} and V_{jG} must be false. Therefore, we can color V_i in red, V_j in blue.

2. 3SAT $\leq_p$ X3C

Idea: When we transform a problem into another we need to preserve (in another word, mimic) the characteristics of the original problem. So, to transform a 3SAT problem, let's first see what its characteristics are.

Any instance of 3SAT problem is a set of clauses. In 3SAT, there are literals, clauses and consistency constraints.

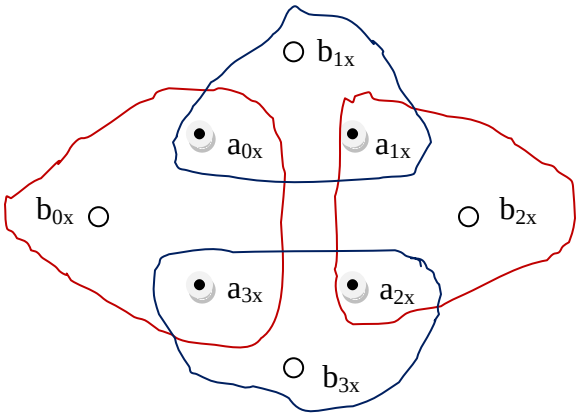
Each literal can have the positive form (x) and its negation form ($\bar{x}$). The constraint for a literal is that its positive form and negation form cannot be true at the same time. That means, when transforming, if we have a representation for x, and another representation for $\bar{x}$, then they cannot be selected at the same time. A common technique is to have a "flip-flop" gadget representing each literal.

Besides, we need to maintain some constraint in order to ensure that clauses are satisfied.

Transformation:

(1) Take an arbitrary instance of the 3SAT problem. It's a 3CNF W consisting of n variables and m clauses. W is satisfiable if there is a true-false assignment for each variable in W so that every clause of W is true.

(2) For each variable (eg, x) of the 3CNF, we make a gadget as following:



Later if x is assigned true, a_{0x} is matched with a_{3x} and b_{0x}, a_{1x} is matched with a_{2x} and b_{2x}. This selection is marked with red boundary on the picture. In contrast, if x is assigned false, then we select as marked blue on the picture.
Note that this gadget together with the selection rule above guarantee that each literal can be assigned either true or false but not both value. Because if it does, then there will be an item repeated between sets in X3C problem.

In each selection above there will be 2 extra b values in each gadget. We take care of 1 of them by introducing 2 variables c_i and c_i' for each clause i. We'll match c_i and c_i' with one of the value of b_1 or b_3 of either one of three variables from that clause if that value of b was not selected.

If the instance of 3SAT problem has n variables and m clauses, then with the selection as above, we will have (2n – m) values of b left over that are not matched with anything. To handle these values of b, we introduce (2n – m) pair of w and w' that we call cleaning clauses to match with the remaining b values.

With this transformation, with each instance of 3SAT problem with n variables and m clauses, we have an instance W' of X3C problem with:

- a set S of 12n points (8n + 2m + (2n-m)*2 = 12n)
- a set C of triples: 4 triples per variable (corresponding to x and $\bar{x}$), 6 triples per clauses (each of these 6 triples consists of c_i, c_i' and either b_1 or b_3 of one of three variables of that clause), 4n clean-up triples, each consisting of a pair of w and w' and value of b

Attention: You should ask yourself a question "Does our gadget work if a variable appear more than twice in the 3CNF?". It won't. However, we can easily transfer any 3SAT to ensure that any variable cannot appear more than twice. The trick is that you can rename any variable that appear k > 2 times as x_1, x_2,….,x_k. Then you can add the following clauses:

$$(\overline{x_1} + x_2)\ (\overline{x_2} + x_3)\ldots..\ (\overline{x_k} + x_1)$$

This 2SAT then can be easily transformed to 3SAT.

Hence, we can always use the gadget above.

Claim:

W is in 3SAT iff W'={S,C} is in X3C.

Proof:

= >

We need to prove that if there is an assignment that makes W satisfied, then we must be able to find a subset C' of C of size 4n to cover S.

Since, for each assignment of variable x, we choose 2 triples correspondingly, we have 2n triples for n variables.
For each clause, we choose a triple of c_i , c_i' and a value of b. For m clauses, we have m triples.
Then we select 2n – m cleaning triples.
This ensures that we cover all 12n points and the size of set C' is exactly 2n + m + 2n –m = 4n.

<=

We must prove that if we have a solution for W', then we have a solution for W. This is rather trivial because we can just look at the combination of **a** and **b** and decide if a variable is assigned true or false.

Also, because of the way we construct the sets that contain c_i and c_i', it guarantees that there is always at least one true variable in each clause. Otherwise, there will be conflict between this set of c_i, c_i' and a set corresponding to a false assignment.

3. **3SAT $\leq_p$Partition into Triangles**
   This problem is similar to problem 2 except that we should add edges in each triple.

4. **3SAT $\leq_p$NAE 3SAT**
   Solution can be found in lecture note.

5. **3SAT $\leq_p$One-In-3SAT**
   Solution can be found in lecture note.

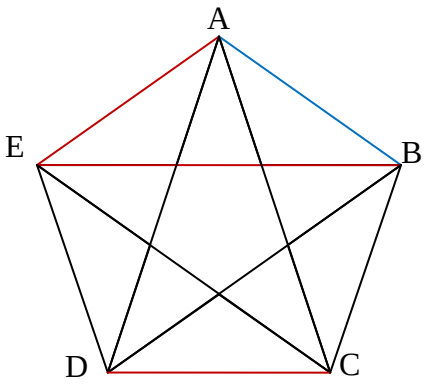
6. NAE 3SAT $\leq_p$ Monochromatic Triangle

Idea:

Definition: In a triangle, if 2 edges are color in the same color then that color is the color of that triangle.

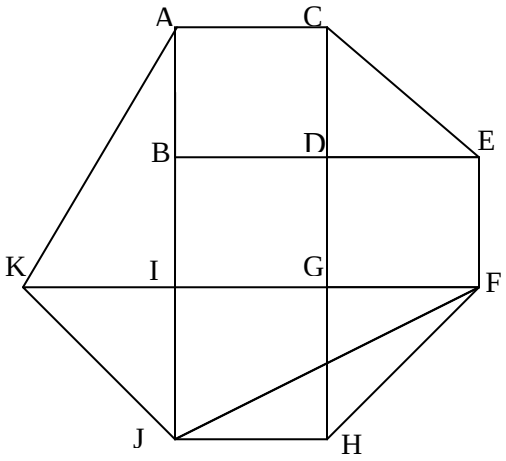
We are going to use the following lemmas:

Lemma 1: In a clique of size 5 (K_5), a triangle and its opposite edge must have the same color



For example, in the graph above, ABE and CD have the same color

Lemma 2: Given a component of 6 K_5 ACEDB, CEFGD, EFHGD, FHJIG, HJKIG, and JKABI as in the figure below, If there is no monochromatic triangle, then triangle CDE and triangle JFH must have different color.
For convenience, let's call CDE the upper triangle and JFH the lower triangle of the component.



Note: for simplicity, we don't connect all the vertices of cliques together.

Transformation:

(1) Take an arbitrary instance of NAE 3SATF problem. It's a 3CNF W. W is satisfied if there is a truth-false assignment for all variables in W so that every clause in W is true and there must be at least one variable in each clause is false.

(2) Let W have n variables and m clauses.

Construct a graph G of n sub-graphs $G_1, \ldots, G_n$. Each sub-graph G_i consists of 6 K_5 as in lemma 2.

It's clear that with the result of the lemma above, we can use these sub-graphs to represent each variable and ensure the constraint of true-false assignment.

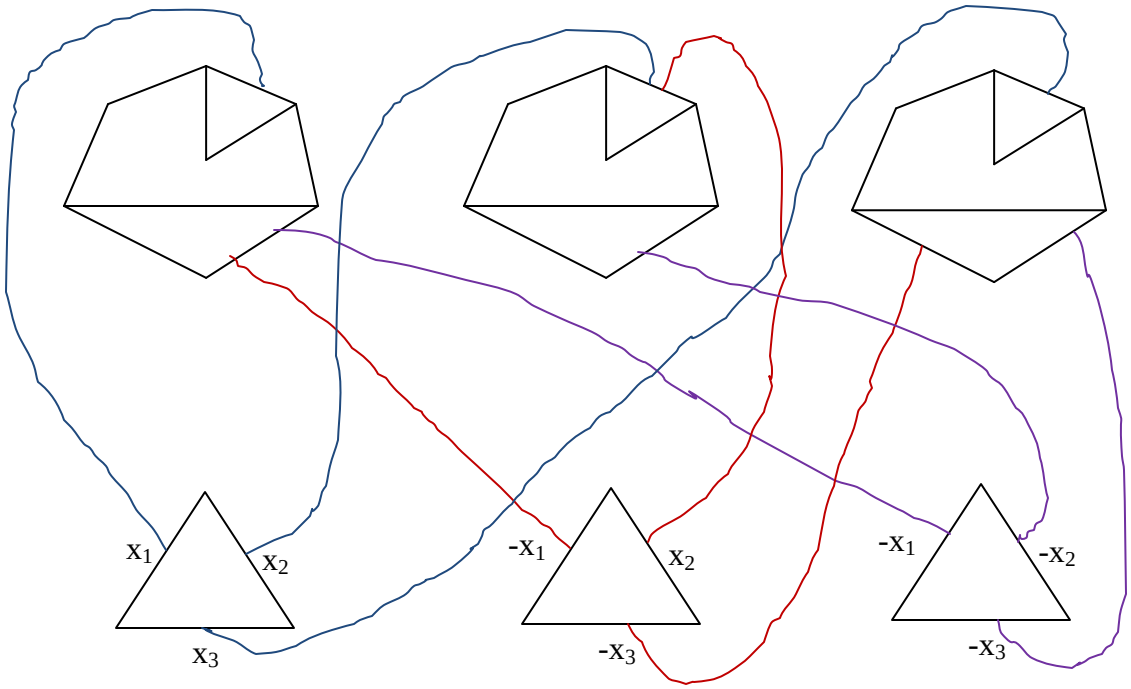
For each clause, construct a new triangle, each edge corresponding to a variable in the clause.

If a variable in a clause is in the positive form (x), connect the edge representing this variable in the clause with the upper triangle in the sub-graph corresponding to this variable. Otherwise, if a variable in the clause is in its negation form ($\bar{x}$), connect its edge to the lower triangle of its corresponding sub-graph. By doing so, we create a K_5 and ensure that this edge must be colored the same as the triangle it's connected to.

For example, given an instance of NAE 3SAT as following:

$$(x_1 + x_2 + x_3)(\overline{x_1} + x_2 + \overline{x_3})(\overline{x_1} + \overline{x_2} + \overline{x_3})$$

We will have a graph G as in the picture below:



If a variable is assigned true, then we color the corresponding edge in the clause triangle in blue; consequently, the corresponding triangle in the sub-graph G_i must be colored in blue as well. In contrast, if a variable is assigned false, we color its

corresponding edge in the clause triangle and its corresponding triangle in the sub-graph G_i in red.

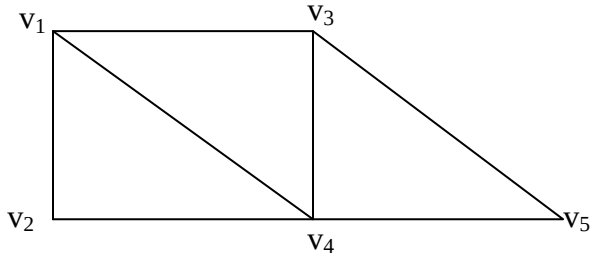
Claim:
W is in NAE 3SAT iff G is in monochromatic triangle.

Proof:
= >
We prove that if there is a solution for NAE 3SAT problem then there will be a solution for the Partition into Triangle problem.
If there is a truth assignment for the NAE 3SAT problem then in each clause, there must be at least 1 true and at least 1 false literal. So, in each triangle, there is at least 1 blue edge and 1 red edge. Hence, it's not a monochromatic triangle.
Also, we can always color any sub-graph G_i so that it does not have any monochromatic triangles.
Besides, for each K_5 constructed by connecting an edge of clause triangle with a triangle in the sub-graph G_i has no monochromatic triangle either.
That means the whole graph G has no monochromatic triangle.

< =
We prove that if there is a solution for graph G, then there will be a truth assignment for NAE 3SAT
If G has no monochromatic triangle, then each clause triangle must have at least 1 red and at least 1 blue edge. Hence, each clause of the 3SAT instance must have 1 true and 1 false assignment.

7. Hamiltonian Cycle $\leq_p$ SAT

Idea:
Let's assume we have a Hamiltonian cycle for a graph G as following:



Let's say the path we found is $\{v_1, v_3, v_5, v_4, v_2, v_1\}$.
If we use x_{ij} to denote that vertex **i** is at position **j** in the cycle, then the cycle can be written as: $\{x_{11}, x_{32}, x_{53}, x_{44}, x_{25}, x_{16}\}$.

There are some constraints that we must ensure in order to guarantee that a graph G has a hamiltonian cycle:

- Exactly one vertex must be in each position p_1 to p_n
- Each vertex, except the first vertex, must be assigned to exactly 1 position.
- Position 1 and (n+1) in the cycle must be the same vertex.

Transformation:

Take an arbitrary instance of Hamiltonian cycle problem. It's an undirected graph G(V, E). |V| = n

1) A vertex can only be at one position in the cycle, excluding the first node

Firstly, to state that a node *i* can be in any position: $(v_{i1} + v_{i2} + ... + v_{in})$

Then, to state that a node i can only be in one position at the time:

$v_{i1} \to \bar{v}_{i2},\ v_{i1} \to \bar{v}_{i3} \dots v_{i1} \to \bar{v}_{in}$

$v_{i2} \to \bar{v}_{i3},\ v_{i2} \to \bar{v}_{i4} \dots v_{i2} \to \bar{v}_{in}$

…

$v_{i(n-1)} \to \bar{v}_{in}$

Since in general $p \to q$ is equivalent to $\bar{p} + q$, the condition can be formulated in general as follows:

$$\forall i\ C_i^1 = (v_{i1} + v_{i2} + ... + v_{in})$$
$$(\bar{v}_{i1} + \bar{v}_{i2})(\bar{v}_{i1} + \bar{v}_{i3})...(\bar{v}_{i1} + \bar{v}_{in})$$
$$(\bar{v}_{i2} + \bar{v}_{i3})(\bar{v}_{i2} + \bar{v}_{i4})...(\bar{v}_{i2} + \bar{v}_{in})$$
$$...$$
$$(\bar{v}_{i(n-1)} + \bar{v}_{in})$$

2) Two vertexes of G cannot be in the same position in the cycle.

For every position *i*, one vertex from G need to be there: : $(v_{1i} + v_{2i} + ... + v_{ni})$

Then, to state that if a node is in position i, the other nodes should be elsewhere:

$v_{1i} \to \bar{v}_{2i},\ v_{1i} \to \bar{v}_{3i} \dots v_{1i} \to \bar{v}_{ni}$

$v_{2i} \to \bar{v}_{3i}$, $v_{2i} \to \bar{v}_{4i} \ldots v_{2i} \to \bar{v}_{ni}$

…

$v_{(n-1)i} \to \bar{v}_{ni}$

Again, since $p \to q$ is equivalent to $\bar{p} + q$, the condition can be formulated in general as follows:

$\forall$ *position* i

$$C_i^2 = (v_{1i} + v_{2i} + \ldots + v_{ni})$$
$$(\bar{v}_{1i} + \bar{v}_{2i})(\bar{v}_{1i} + \bar{v}_{3i}) \ldots (\bar{v}_{1i} + \bar{v}_{ni})$$
$$(\bar{v}_{2i} + \bar{v}_{3i})(\bar{v}_{2i} + \bar{v}_{4i}) \ldots (\bar{v}_{2i} + \bar{v}_{ni})$$
$$\ldots$$
$$(\bar{v}_{(n-1)i} + \bar{v}_{ni})$$

3) Position 1 and n+1 in the cycle must be the same.

$v_{11} \to v_{1(n+1)}$, $v_{21} \to v_{2(n+1)}$, …, $v_{n1} \to v_{n(n+1)}$

Again, since $p \to q$ is equivalent to $\bar{p} + q$, the condition can be formulated in general as follows:

$$C^3 = (\bar{v}_{11} + v_{1(n+1)})(\bar{v}_{21} + v_{2(n+1)}) \ldots (\bar{v}_{n1} + v_{n(n+1)})$$

Finally the SAT formula will be:

$$W = C_i^1 C_i^2 C^3$$

Claim:

G has Hamiltonian path iff W is satisfiable.

Proof:

$\Rightarrow$

Suppose there is a Hamiltonian Cycle in graph G, then, for the way it was constructed, all $C_i^1 C_i^2 C^3$ will evaluate to true. Vice versa, if there is not a Hamiltonian Cycle, at least one condition will fail.

$\Leftarrow$

Suppose W evaluates to true. Then all $C_i^1 C_i^2 C^3$ need to be true. C_i^1 implies that a vertex cannot be repeated. C_i^3 implies that there should between the first node traversed and the last node traversed. Finally, C_i^2 implies that nodes need to be traversed one by one. All together the conditions satisfy the definition of Hamiltonian Cycle. Conversely, if W evaluates to false, it is easy to see that at least one of the three C will not be satisfied.

8. X3C $\leq_p$VC

9. Planar Cubic Hamiltonian Path $\leq_p$Planar Steiner Tree

Idea:

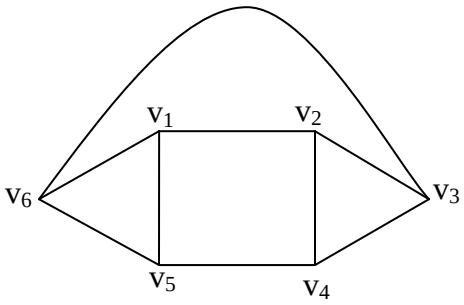
Transformation:

(1) Take an arbitrary instance of the planar cubic hamiltonian path. It's an undirected graph G(V,E) whose every vertex has degree equal three. V={v_1, v_2, …, v_n}

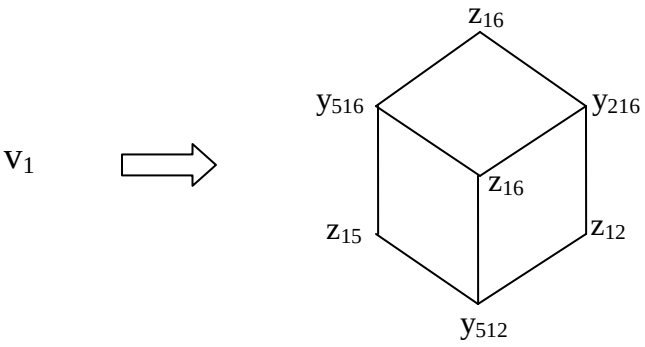
(2) We construct a new undirected graph G'(V', E') as following:

- w(e) = 1 $\forall e \in E'$
- V' = X $\cup$ Y $\cup$ Z
- X = {x_1, x_2, …., x_n}
- Y = {y_{kij} | k,i,j $\in$ {1,2,…,n}, e(v_k, v_i) $\in$ E, and e(v_i, v_j) $\in$ E}
- Z = {z_{pq}, z_{qp} | p,q $\in$ {1,2,…,n}, and e(v_p, v_q) $\in$ E}
- E' = $E_1 \cup E_2 \cup E_3$
- E_1 = {e(x_i, y_{kij})}
- E_2 = {e(y_{kij}, z_{jk}), e(e_{kij}, z_{ij})}
- E_3 = {e(z_{pq}, z_{qp})}

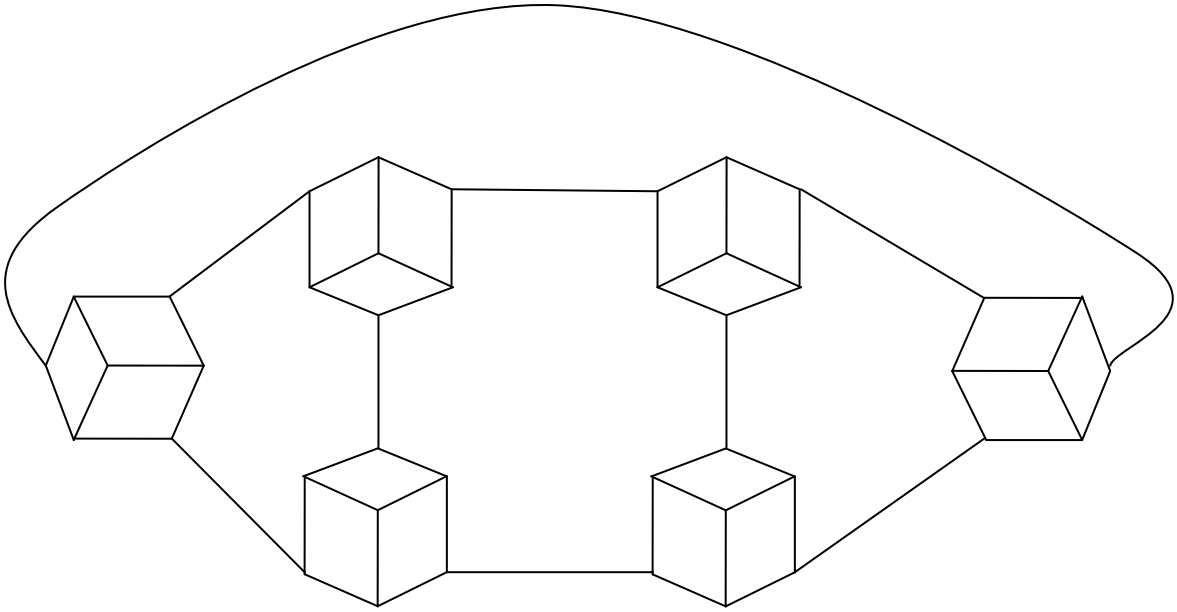
(3) For example, given a graph G as following



Each vertex in G is transformed into a component tin G' as following



If two vertices in G are connected to each other, then their components in G' are also connected to each other through **z** vertices.
Hence, graph G' is as following:



Claim:
Graph G has a Hamiltonian path iff graph G' has a steiner tree that covers all vertices of set X of size 4n -3.

Proof:
= >
If there is a Hamiltonian path in graph G, then each vertex in G is visited exactly once by one incoming edge and one outgoing edge, except the source and destination vertices. There are three possible paths for each vertex but only one can be chosen.

Correspondingly, in graph G', for each component, only one of three y vertices can be visited, and its two adjacent edges connecting to the z vertices will also be visited. Except for the source and destination components, we only need one adjacent edge. Thus, at least (**2n -2**) edges will be added into the tree.
For each component, each vertex x must be visited. Therefore, an edge v(y,x) must be added to the steiner tree. There are **n** edges like that.
At least (**n-1**) edges, connecting components together, must be visited in order to visit all x vertices.
Hence, at least (4n-3) edges must be added to the tree. Thus, the Steiner Tree's weight is (4n-3)

< =
We'll prove that if there is no Hamiltonian path in G, then we cannot find a Steiner Tree whose weight is (4n-3) in G'.
Suppose there's no Hamiltonian Path in G, then at least one vertex will be visited twice. Thus, in graph G', at least one component will be visited twice. That means, two of the three y vertices in that component will be visited, which will cover four (y,z) edges instead of two.
Furthermore, there are three "leaf components" instead of two.
So, to construct a Steiner Tree which covers all the X vertices, we need n edges to connect the X vertices, (n-1) edges to connect the n components, another (2n-3+2) edges to connect the y vertices and z vertices.
Thus, there are at least (4n-2) edges in the tree, which is larger than (4n-3).