CSCI-570 HW5

1.

First, we will define the variables.

Let x1, x2, x3 be the type of couches that is type1, type2, type3 respectively.
Then we have the total profit made by the company as $(10x_1 + 8x_2 + 5x_3)$.
Our objective function will be

$$\mathbf{max\ (10x_1 + 8x_2 + 5x_3)}$$

For the current month, we do have a shortage of framing wood by 600 feet and additional supply of cabinet wood by 100 feet. This leads to the following system of un equalities:

$$\mathbf{x_1 + 2x_2 + 2x_3 \leq -600}$$

$$\mathbf{3x_1 + 2x_2 + x_3 \leq 100}$$

From the question, we have that the factory produces 500 couches each month of the first type, 300 of the second type, and 200 of the third type. We have:

$$\mathbf{x_1 \leq 500 \Rightarrow x_1 \geq -500}$$

$$\mathbf{x_2 \leq 300 \Rightarrow x_2 \geq -300}$$

$$\mathbf{x_3 \leq 200 \Rightarrow x_3 \geq -200}$$

So, we have following linear programming:

$$\mathbf{max\ (10x_1 + 8x_2 + 5x_3)}$$

$$\mathbf{subject\ to\ 3x1 + 2x2 + x3 \leq -600,}$$

$$\mathbf{x1 + 2x2 + 2x3 \leq 100,}$$

$$\mathbf{x1 \geq -500}$$
$$\mathbf{x2 \geq -300}$$
$$\mathbf{x3 \geq -200.}$$

2.

We have an objective function:

$$\mathbf{max\ (3x_1 + 2x_2 + x_3)}$$

equation $\mathbf{-x_1 + 2x_3 = 3}$ is not in the standard form. We will convert it to standard form,

$$\mathbf{-x_1 + 2x_3 \leq 3}$$

$$\mathbf{-x_1 + 2x_3 \geq 3 \Rightarrow x_1 - 2x_3 \leq -3}$$

We will be Multiplying each equation by a new variable $y_k \geq 0$.

$$\mathbf{y1(x1 - x2 + x3) \leq 4y1}$$

$$\mathbf{y2(2x1 + x2 + 3x3) \leq 6y2}$$

$$\mathbf{y3(x1 + x2 + x3) \leq 8y3}$$

$$\mathbf{y4\ (-x1 + 2x3) \leq 3y4}$$

$$\mathbf{y5(x1 - 2x3) \leq -3y5}$$

- Next step is to add up those equations.
  $y1(x1 - x2 + x3) + y2(2x1 + x2 + 3x3) + y3(x1 + x2 + x3) + y4(-x1 + 2x3) + y5(x1 - 2x3) \leq 4y1 + 6y2 + 8y3 + 3y4 - 3y5$
- Now, we will collect the terms with respect to x_k.
  $x1(y1 + 2y2 + y3 - y4 + y5) + x2(-y1 + y2 + y3) + x3(y1 + 3y2 + y3 + 2y4 - 2y5) \leq 4y1 + 6y2 + 8y3 + 3y4 - 3y5$

Now, the given problem can be written in a matrix form:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad c = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}, \quad b = \begin{bmatrix} 4 \\ 6 \\ 8 \\ 3 \\ -3 \end{bmatrix}, \quad A = \begin{bmatrix} 1 & -1 & 1 \\ 2 & 1 & 3 \\ 1 & 1 & 1 \\ -1 & 0 & 2 \\ 1 & 0 & -2 \end{bmatrix}$$

Now we have, $b^T = [4 \quad 6 \quad 8 \quad 3 \quad -3]$ and $A^T = \begin{bmatrix} 1 & 2 & 1 & -1 & 1 \\ -1 & 1 & 1 & 0 & 0 \\ 1 & 3 & 1 & 2 & -2 \end{bmatrix}$

- Choose y_k in a way that $A^T y \geq c$.

Dual form is, **min ($4y_1 + 6y_2 + 8y_3 + 3y_4 - 3y_5$)**

is subjected to:

$y_1 + 2y_2 + y_3 - y_4 + y_5 \geq 3$
$-y_1 + y_2 + y_3 \geq 2$
$y_1 + 3y_2 + y_3 + 2y_4 - 2y_5 \geq 1$
$y_1, y_2, y_3, y_4, y_5 \geq 0$

3.

Let x_{ij} denotes the assignment of frequency f_i to the stations s_i. As there are n stations, n frequencies would be enough to be assigned. We have, $1 \leq i,j \leq n$. Also, we will define variable t_j, $\begin{cases} t_j = 0 \\ t_j = 1 \end{cases}$ for each frequency f_i. Our ILP would be

$$\min_{x,t} \sum_{k=1}^{n} t_k$$

Subjected to

(i)	$x_{i,j}, x_{i',j} \leq 1,$	$\forall_j$ and $\forall(i, i')$, (s_i, s_j) is an edge.
(ii)	$\sum_{=1}^{n} x_{i,j} = 1,$	$i, \in [n]$
(iii)	$x_{i,j} \leq t_j ,$	$i,j \in [n]$
(iv)	$t_j, x_{i,j} \in \{0,1\}$,	$i,j \in [n]$

The first inequality means that no adjacent stations will have the same frequency where I and I' ar the adjacent stations. The second equality implies that one station will have only one frequency. Third inequality implies that only if the frequency is active, it is assigned to the corresponding station.

4.

1. This is **False**. An empty language $\emptyset$ can be transformed into NP hard problem but empty language is not NP hard.

2. This is **True.** One can construct a certifier for A by composing the certifier for B and the polynomial reduction map.

3. We know that 2-SAT is in P and 3-SAT is in NP. From the reduction, it says that 2-SAT is in NP. Therefore, we have P = NP.

4. This is **True**. As the polynomial time deterministic turing machine can check all possible certificates in exponential time, any NP problem can be solved in time $O(2^{poly(n)})$.

5. This is False. We have A $\leq_p$ B. Let A be a problem in P and B be a problem that requires exponential time to solve. If B $\leq_p$ A is true, which means that there should be a polynomial time algorithm for B. This is a contradiction.

5.

Consider an instance of 3-SAT with variables S = {s_1 , . . . , s_n } and clauses C_1,...,C_k.

We have, $\varphi(s_1,...,s_n) = C_1 \wedge \cdots \wedge C_k$ be the corresponding Boolean formula. Let's define φ_i and $\bar{\varphi} i$ as the CNF formula when we set s_i to TRUE and FALSE, respectively.

Let A(φ) be the given algorithm that decides whether a given instance has a satisfying assignment. A(φ) returns TRUE if φ is satisfiable and returns FALSE if φ is not satisfiable.

Here, we can see that φ_i is satisfiable if and only if φ has a satisfying assignment that is when s_i = TRUE. Then the following algorithm constructs satisfying assignment for φ if possible, or returns that no such assignment exists.

If A(φ) = FALSE,

return that φ not satisfiable. • For i=1,...,n

If A(φ_i) = TRUE

$\varphi = \varphi_i$

s_i = TRUE

Else

$\varphi = \bar{\varphi}_i$
s_i = FALSE

Return s = (s_1,...,s_n)

6.

To prove that the problem is NP Complete, we need to first prove that it is in NP. Then we take NP Complete problem that is Hamiltonian path problem and reduce it to minimum remoteness problem in polynomial time.

This problem is NP.

A path with remoteness at most k can be verified in polynomial time.

Given a path, the remoteness of the path can be calculated in polynomial time. For each vertex which is not in the path P, we run Dijkstra algorithm to each node on P as target and keep track of the minimum distance to the path. After that, we do the summation of all the minimum distances obtained. Dijkstra algorithm will have time complexity O(V + E log E) and our verification algorithm will take V^2((V + E) log E).

Claim : G has a Hamiltonian path if and only if G has a path with remoteness k=0.

Proof: We must show the implications on each directions

=>) If the Hamiltonian path exists, then the remoteness of path is zero.

A Hamiltonian path in G visits every vertex in the graph only once. So, all the nodes lie on that path.

$$\forall\, v \ \in V, d(v, P\,) = 0.$$

Because all the vertices will be on the path, the remoteness for each of the vertices will be 0 and $d(v, P)$ will also be 0. Since a Hamiltonian path is equivalent to remoteness 0, it means that there is a polynomial time reduction from the Hamiltonian path problems to the multilane highway problem.

<=) If the remoteness value is zero, then a Hamiltonian path exists in the graph

Remoteness value i.e.. k is zero, tells that all the nodes are on the path. This path is now constructed in such a way that it covers all the nodes and each node visited only once. This is nothing but a Hamiltonian path. Hence, Hamiltonian path exists in the graph.

Thus, we have proved that Hamiltonian problem can be reduced to the given problem and is NP-complete.