CSCI 570 : HW-05 (Spring 2021)

April 16, 2021

Question 1

A furniture company produces three types of couches. The first type uses 1 foot of framing wood and 3 feet of cabinet wood. The second type uses 2 feet of framing wood and 2 feet of cabinet wood. The third type uses 2 feet of framing wood and 1 foot of cabinet wood. The profit of the three types of couches is $10, $8, and $5, respectively. The factory produces 500 couches each month of the first type, 300 of the second type, and 200 of the third type (with no wood resources leftover). However, this month there is a shortage of cabinet wood to only 600 feet, but the supply of framing wood is increased by 100 feet. How should the production of the three types of couches be adjusted to minimize the decrease in profit? Formulate this problem as a linear programming problem.

Answer

Let x_1, x_2 *and* x_3 should be the production of three types of couches this month that will minimize the decrease in profit.

Using information given,

Type	Framing Wood Required	Cabinet Wood Required	Profit
I	1	3	10
II	2	2	8
III	2	1	5

Initially,

The amount of framing wood required to produce 500, 300 and 200 units furniture of type I, II and III respectively = $500 \cdot 1 + 300 \cdot 2 + 200 \cdot 2 = 1500$

The amount of cabinet wood required to produce 500, 300 and 200 units furniture of type I, II, III respectively = $500 \cdot 3 + 300 \cdot 2 + 200 \cdot 1 = 2300$

Amount of framing wood available this month = $1700 + 100 = 1600$

Amount of cabinet wood available this month = $2300 - 600 = 1700$

So, minimize decrease in profit we have to maximize our profit this month.

Objective Function :

$$max(10 \cdot x_1 + 8 \cdot x_2 + 5 \cdot x_3)$$

Constraints :

1. Amount of cabinet wood required for x_1, x_2 and x_3 units type I, II and III should be less than 1700.

$$1 \cdot x_1 + 2 \cdot x_2 + 3 \cdot x_3 \leq 1700$$

2. Amount of framing wood required for x_1, x_2 and x_3 units type I, II and III should be less than 1600.

$$3 \cdot x_1 + 2 \cdot x_2 \cdot 1 \cdot x_3 \leq 1600$$

3. Units of each furniture type cannot be negative.

$$x_1, x_2, x_3 \geq 0$$

Question 2

Consider the following linear program:

$$max(3x_1 + 2x_2 + x_3)$$

subject to

$$x_1 - x_2 + x_3 \leq 4$$

$$2x_1 + x_2 + 3x_3 \leq 6$$

$$-x_1 + 2x_3 = 3$$

$$x_1 + x_2 + x_3 \leq 8$$

$$x_1, x_2, x_3 \geq 0$$

Write the dual problem. You do not need to demonstrate intermediate steps.

Answer

We can write the constraints in the problem in the following standard format (primal)

$max\ c^T x$

$Ax \leq b\ and\ x \geq 0$

Objective Function : $max\ (3x_1 + 2x_2 + x_3)$

Constraints :
$x_1 - x_2 + x_3 \leq 4$
$2x_1 + x_2 + 3x_3 \leq 6$
$-x_1 + 2x_3 \leq 3$
$x_1 - 2x_3 \leq -3$
$x_1 + x_2 + x_3 \leq 8$
$x_1, x_2, x_3 \geq 0$

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

We will write dual of the above problem in the following format

$min\ b^T y$

$A^T y \geq c\ and\ y \geq 0$

Here, $A^T = \begin{bmatrix} 1 & 2 & -1 & 1 & 1 \\ -1 & 1 & 0 & 0 & 1 \\ 1 & 3 & 2 & -2 & 1 \end{bmatrix}$, $y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix}$

Dual Problem

Objective Function : $min(\ 4y_1 + 6y_2 + 3y_3 - 3y_4 + 8y_5\ )$

Constraints :

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

$-y_1 + y_2 + y_5 \geq 2$

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

$y_1, y_2, y_3, y_4, y_5 \geq 0$

Question 3

Spectrum management is the process of regulating the use of radio frequencies to promote efficient use and gain a net social benefit. Given a set of broadcast emitting stations $s_1, ..., s_n$, a list of frequencies $f_1, ..., f_m$, where $m \geq n$, and the set of adjacent stations $\{(s_i, s_j)\}$ for some $i, j \in [n]$. The goal is to assign a frequency to each station so that adjacent stations use different frequencies and the number of used frequencies is minimized. Formulate this problem as an integer linear programming problem.

Answer

We formalize the problem by introducing an indicator variable w_{ij} to indicate where frequency j is assigned to station s_i, and another variable x_j denotes whether frequency j is assigned to some station or not.

$$w_{ij} = \begin{cases} 1, & \text{if frequency } f_j \text{ is assigned to stations } s_i \\ 0, & \text{otherwise} \end{cases}$$

$$x_j = \begin{cases} 1, & \text{if frequency } f_j \text{ is assigned to some station} \\ 0, & \text{otherwise} \end{cases}$$

Then, we will write the problem as follows,

Objective Function : We have to minimize number of frequency used.

$$min(\sum_{j=1}^{m} x_j) = max(-\sum_{j=1}^{m} x_j)$$

Constraints :

1. For every station i, sum of all indicator variables for all frequency (w_{ij}) should be 1.

$$\forall i \in S \sum_{j=0}^{m} w_{ij} = 1$$

In standard form,

$$\forall i \in S \sum_{j=0}^{m} w_{ij} \leq 1, \forall i \in S - \sum_{j=0}^{m} w_{ij} \leq -1$$

2. For any 2 adjacent stations, the sum of indicator variable for a particular frequency should be less than or equal to 1.

$$\forall (u, v) \in E, \forall j \in F \quad w_{uj} + w_{vj} \leq 1$$

3. The indicator variable to tell if frequency f_j is used for station s_i should upper bounded by indicator variable that tells if frequency f_j is used.

$$\forall i \in S, \ \forall j \in F \ w_{ij} \leq x_j$$

In Standard Form,

$$\forall i \in S, \ \forall j \in F \ w_{ij} - x_j \leq 0$$

4. For any 2 adjacent stations, sum of indicator variables for both station for any frequency should be less or equal to frequency selection indicator for that frequency.

$$\forall (u, v) \in E, \forall j \in F, \quad w_{uj} + w_{vj} \leq w_j$$

In Standard Form,

$$\forall (u, v) \in E, \forall j \in F, \quad w_{uj} + w_{vj} - w_j \leq 0$$

Question 4

Prove or disprove the following statements.

1. If $A \leq_p B$ and B is in NP-hard, then A is in NP-hard.

2. If $A \leq_p B$ and B is in NP , then A is in NP.

3. If $3 - SAT \leq_p 2 - SAT$, then P = NP.

4. Any NP problem can be solved in time O($2^{poly(n)}$), where n is the input size and poly(n) is a polynomial.

5. If a problem $A \leq_p B$, then it follows that $B \leq_p A$.

Answer

1. If $A \leq_p B$ and B is in NP-hard, then A is in NP-hard.

The given statement is False. Problem A is reducible to problem B in polynomial time which means that A is atleast as hard as B. From this, we cannot infer anything about A and thus the given statement is false.
Proof by Counter example: Let us have an empty language as problem A. This language is reducible to n-SAT by literals such as $x \wedge \sim x$. In this example, our problem A is reducible to n-SAT which is NP-hard but problem A is not NP-hard.

2. If $A \leq_p B$ and B is in NP, then A is in NP.
The given statement is true. Since B is in NP, it means that there exists a polynomial time algorithm that can verify the solution of B. We have reduced a given problem A to B in polynomial time by a function $f(x)$ such that $\forall x \in A, f(x) = y$ and $y \in B$. We know that we can verify y in polynomial time and that f is also a polynomial time function. This means that A can also be verified in polynomial time. Thus, A is in NP.

3. If $3 - SAT \leq_p$ 2-SAT, then P=NP.
The given statement is true. Let us prove it by contradiction.
Let us assume that P≠NP and there is a polynomial time reduction of 3-SAT to 2-SAT. We already know that 3-SAT is NP-complete and hence, we can derive that 2-SAT is also NP-complete. However, we also know that 2-SAT can be solved in polynomial time. Thus, for all decision problems $A \in NP$, we can reduce A to a 2-SAT instance in polynomial time and solve it in polynomial time, making the total time also as polynomial. Hence, $A \in P$ and thus, $NP \subseteq P$. We also already know that $P \subseteq NP$. Thus, we can conclude that P = NP which is a contradiction.

4. Any NP problem can be solved in time $O(2^{poly(n)})$, where n is the input size and poly(n) is a polynomial.
Answer: Given statement is True.

Explanation :

- Class NP consists of those problem, which can be solved in exponential time, and can be verified in polynomial time such as Decision Travelling Salesman Problem.
- There is another class of problems, EXPT which can be solved in exponential time and there does not exists any polynomoal time algorithm, that can verify the solution E.g.

Tree Quantified Boolean Formula.

- Hence, NP is the subset of EXPTIME.
- So, all problems that can be solved in exponential time are in in class EXPTIME, but the ones which can be verified in polynomial time, are the only ones among them, that are in class NP.

Eg. Let's assume we have an arbitary problem L in NP, (it's solution can be validated in polynomial time.)

To Show: **L is solvable in exponential time.**

Since $L \in NP$, there exists a deterministic machine M and p(n), q(n) polynomials such that x in L iff there is a certificate y such that $|y| \leq_p |x|$ and M accepts yx in time $q(|x|)$. Construct a nw determinsitic machine M' that iterates through all $2^{p(|x|)}$ certificates c, simulating M on cx for $q(|x|)$ steps and accepting if M does. If no certificate validates, M' rejects.
The M has total runtime $O(2^{p(n)}q(n))$ and decides L, hence L in EXP.

5. If a problem $A \leq_p B$, then it follows that $B \leq_p A$.
The given statement is false. A can be reduced to B in polynomial time using any polynomial time function, lets say f. This means that $\forall x \in A, f(x) = y$ where $y \in B$. In our set B, there could be some y_i that can be mapped back to some $x_i \in A$. However, there also exists some $f(x_j) = y_j \in B$ that doesn't always map back to any $x_i \in A$. This means that $f(x) \subset B$.
This means that not all $y_i \in B$ can be converted back to A. Hence, the given statement is false. In other words, since we already know that B is more harder than A, it is not compulsory that we can reduce every problem in B to A. Take the example of Max-Flow $\leq_p$ Bipartite Matching. Not every max-flow problem can be solved by reducing it to Bipartite Matching.

Question 5

Assume that you are given a polynomial time algorithm that given a 3-SAT instance decides in polynomial time if it has a satisfying assignment. Describe a polynomial time algorithm that finds a satisfying assignment (if it exists) to a given 3-SAT instance.

Answer

Let $\phi(x_1, x_2, x_3.., x_n)$ be the any 3-SAT instance. We are given an algorithm "A" that decides in polynomial time that if there exists a satisfying assignment for that assignment.

Algorithm :

1. If $A\big(\phi(x_1, x_2, ...x_n)\big) = 0$, then the given instance is not satisfiable, and we cannot find any valid assignment.

2. If $A\big(\phi(x_1, x_2, ..., x_i, ..., x_n)\big) = 1$, then the given instance is satisfiable and there exists a valid assignment.

3. For each x_i in $(x_1, x_2, ..., x_i, ..., x_n)$, assign $x_i = 1$, and check $A\big(\phi(x_1, x_2, ..., x_i, ..., x_n)\big)$.

    a. If it is satisfiable, then we will assign $x_i = 1$

b. If it is unsatisfiable, then we will assign $x_i = 0$.

c. In both of the above cases, ϕ has a satisfying assignment. In next iteration we will assign x_{i+1}, given we had already made choices for $(x_1, ..x_i)$.

4. In the end we will get assignment for every x_i, such that $\phi(x_1, x_2, ..., x_n)$ can be satisfied.

Time Complexity :

Since we are making n calls to algorithm "A" which we know has a polynomial runtime. Thus our algorithm also has a polynomial runtime.

Question 6

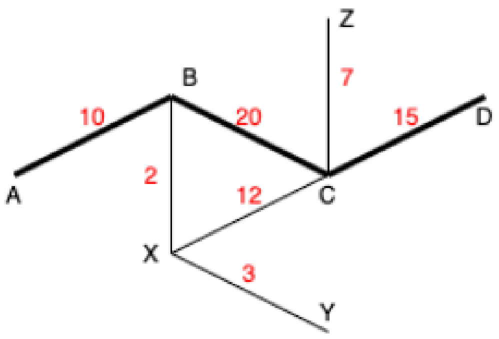
The government wants to build a multi-lane highway across the country. The plan is to choose a route and rebuild the roads along this route. We model this problem with a simple weighted undirected graph with the nodes denoting the cities and the edges capturing the existing road network. The weights of the edges denote the length of the road connecting the corresponding two cities.

Let d_{uv} denote the shortest path distance between nodes u and v.

Let $d(v, P)$ denote the shortest path distance from a node v to the closest node on a path P (i.e. $min_{u \in P} d_{uv}$).

Next, we define the aggregate remoteness of P as r(P) = $\sum_{v \in V} d(v, P)$.

In the example shown in figure below, path P = ABCD is the highway. Then, $d(A, P) = d(B, P) = d(C, P) = d(D, P) = 0$**, while** $d(X, P) = d_{XB} = 2$, $d(Y, P) = d_{YB} = 5$**, and,** $d(Z, P) = d_{ZC} = 7$**. The remoteness of path ABCD is** $0 + 0 + 0 + 0 + 2 + 5 + 7 = 14$**.**

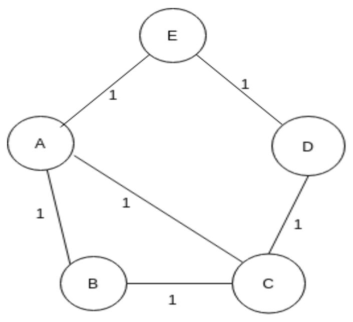


The government wants a highway with the minimum aggregate remoteness, so that all the cities are somewhat close to the highway. Formally, we state the problem as follows, “Given a graph G, and a number k, does there exist a path P in G having remoteness r(P) at most k”? Show that this problem is NP-complete by reduction from the Hamiltonian Path problem.

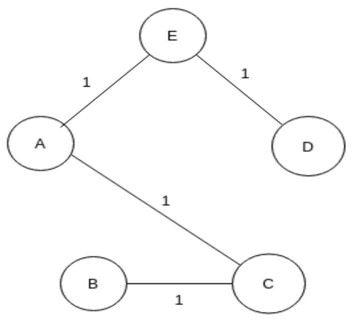
Answer:
We are given a graph in which we have to find the highway such that all cities are somewhat close to the highway with the given remoteness of each city as r(P) = $\sum_{v \in V} d(v, P)$. We call this as the highway problem.

1. The highway problem is in NP since, for the given assignment of vertices to the highway path, we can verify if all cities are connected to this highway for a given remoteness k.
2. We will now reduce the Hamiltonian Path problem to the given problem for k = 0.

   Claim: Graph G has a hamiltonian path iff G' has a remoteness value of $\sum r(P) = k = 0$ i.e all cities have a remoteness 0 from the highway and no vertex is out of the highway.



Graph G



Hamilonian Path G' of Gaph G

   - First, we will try to prove that if all vertices form the highway with k = 0, then there exists a hamiltonian path. Since, all vertices are on the highway, $\forall P, d(v, P) = 0$. Hence, $r(P) = 0 = k$. The Hamiltonian path will cover all vertices on this graph and hence, the given hamiltonian path is the assignment to the given highway problem.
   - Now, we will prove that if there exists a Hamiltonian Path in the graph, there exists a valid assignment to our problem with k = 0. Since, there exists a Hamiltonian Path in the graph, all the cities are covered and lie on the Highway. This path also has remoteness zero because all cities actually lie on the highway path.

   Thus we can say that the highway problem is NP-complete from steps 1 and 2.