CSCI570 Fall2023 HW5

Vinit Pitamber Motwani (USC ID: 8187180925)

November 2023

1. In the network below (See Fig. 1), the demand values are shown on vertices (supply values are negative). Lower bounds on flow and edge capacities are shown as (lower bound, capacity) for each edge. Determine if there is a feasible circulation in this graph. Please complete the following steps.

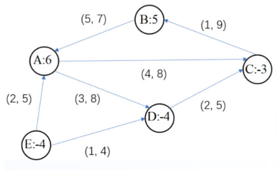


Figure 1

   (a) Remove the lower bounds on each edge. Write down the new demands on each vertex A, B, C, D, E, in this order.

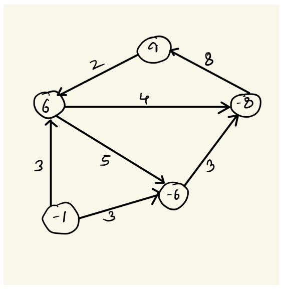
   (b) Solve the circulation problem without lower bounds. Write down the max-flow value.

   (c) Is there is a feasible circulation in the original graph? Explain your answer.

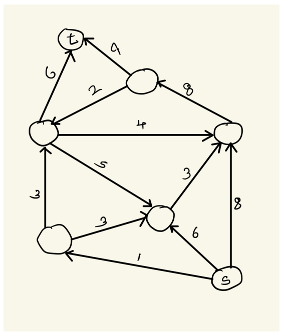
   Solution :

   (a) A:6,B:9,C:-8,D:-6,E:-1

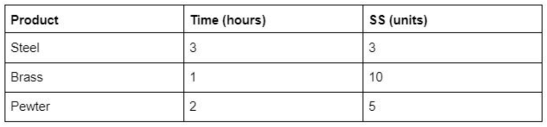
   (b) The max-flow value is: 9



(c) Check if there is a $s - t$ flow $|f| = 15$. The max-flow of this flow network is 9, so there is no feasible circulation.



2. A company is currently trying to fill a large order of steel, brass, and pewter, measured in tons. Manufacturing each ton of material requires a certain amount of time, and a certain amount of special substance (SS), both of which are limited and are given in below table. Note that it is acceptable to manufacture a fraction of a ton (e.g. 0.5t) of material. Specifically, the company currently has 8 hours of time, and 20 units of special substance (SS). Manufacturing each ton of the three products requires:



Product	**Time (hours)**	**SS (units)**
Steel	3	3
Brass	1	10
Pewter	2	5

Figure 2: Table describing the amount of time taken and special substance (SS) required for each product.

(a) Write down a linear program that determines the maximum amount of products (in tons) that the company can make.

(b) Due to the potential danger posed by the special substance (SS), the company would like to use up as much of its supply of special substance (SS) as possible, while:

   i. spending at most 8 hours

   ii. manufacturing a total of at least 2 tons of steel plus pewter.

Solution :

(a) Let:
x = amount of steel to manufacture (in tons)
y = amount of brass to manufacture (in tons)
z = amount of pewter to manufacture (in tons)
Objective function: Maximize the amount of products produced:

$$\max(x+y+z)$$

subject to

$$\begin{aligned} 3x+y+2z &\le 8 \\ 3x+10y+5z &\le 20 \\ x,y,z &\ge 0 \end{aligned}$$

(b) Objective function: Maximize the supply of Special Substance:

$$\max(3x+10y+5z)$$

subject to

$$\begin{aligned} 3x+y+2z &\le 8 \\ 3x+10y+5z &\le 20 \\ x+z &\ge 2..(\textit{steel plus pewter constraint}) \\ x,y,z &\ge 0 \end{aligned}$$

3. Suppose that a cement supplier has two warehouses, one located in city A and another in city B. The supplier receives orders from two customers, one in city C and another in city D. The customer in city C needs at least 50 tons of cement, and the customer in city D needs at least 60 tons of cement. The amount of cement at the warehouse in city A is 70 tons, and the number of units at the warehouse in city B is 80 tons. The cost of shipping each ton of cement from A to C is 1, from A to D is 2, from B to C is 3, and from B to D is 4.
Formulate the problem of deciding how many tons of cement from each warehouse should be shipped to each customer to minimize the total shipping cost as a linear programming. You can assume that the values of units to be shipped are real numbers.

Solution :

Let:
x be the amount of cement (in tons) shipped from warehouse A to customer C,
y be the amount of cement (in tons) shipped from warehouse A to customer D,
z be the amount of cement (in tons) shipped from warehouse B to customer C,
w be the amount of cement (in tons) shipped from warehouse B to customer D.
Objective function: Minimize the total shipping cost:

$$\min(1x+2y+3z+4w)$$

subject to

$$\begin{aligned} x + z &\geq 50 \\ y + w &\geq 60 \\ x + y &\leq 70 \\ z + w &\leq 80 \\ x, y, z, w &\geq 0 \end{aligned}$$

4. Write down the dual program of the following linear program. There is no need to provide intermediate steps.

$$\max(x_1 - 3x_2 + 4x_3 - x_4)$$

subject to

$$\begin{aligned} x_1 - x_2 - 3x_3 &\leq -1 \\ x_2 + 3x_3 &\leq 5 \\ x_3 &\leq 1 \\ x_1, x_2, x_3, x_4 &\geq 0 \end{aligned}$$

Solution :

$$\min(-y_1 + 5y_2 + y_3)$$

subject to

$$\begin{aligned} y_1 &\geq 1 \\ -y_1 + y_2 &\geq -3 \\ -3y_1 + 3y_2 + y_3 &\geq 4 \\ y_1, y_2, y_3 &\geq 0 \end{aligned}$$

There is also a trivial constraint $0 \geq -1$

5. Given an undirected graph $G = (V, E)$, a vertex cover is a subset of V so that every edge in E has at least one endpoint in the vertex cover. The problem of finding a minimum vertex cover is to find a vertex cover of the smallest possible size. Formulate this problem as an integer linear programming problem.

Solution :

For every vertex $v \in V$, introduce a variable x_v and consider the following integer linear program:

$$\min \textstyle\sum_{v \in V} x_v$$

subject to

$$\begin{aligned} x_u + x_v \geq 1, \forall (u,v) \in E \\ x_v \in \{0,1\}, \forall v \in V \end{aligned}$$

6. 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.

   Solution :

   Let $\phi(x_1, x_2, ..., x_n) = c_1 \wedge c_2 \wedge ... \wedge c_m$ be the boolean formula corresponding to a 3-SAT instance where $c_1, c_2, ..., c_m$ are the clauses and $x_1, x_2, ..., x_n$ are the variables and A be the polynomial-time algorithm to decide the 3-SAT. Denote $\phi(x_i = 0)$ and $\phi(x_i = 1)$ as the formulas when a variable $x_i, i \in \{1, 2, 3\}$ is assigned 0 or 1, respectively. The following procedure can find a satisfying assignment:

   (a) Apply A on ϕ to decide whether ϕ is satisfiable. If NO, return "no satisfying assignment can be found"; else continue.
   (b) Initialize $i = 1$
   (c) Let $x_i = 0$, then $\phi(x_i = 0)$ is a boolean formula with $n - i$ variables. Apply A to check the satisfiability of $\phi(x_i = 0)$. If it returns YES, record $x_i = 0$. Else record $x_i = 1$. Since ϕ is satisfiable, there must be an assignment of x_i satisfies ϕ;
   (d) Replace x_i in ϕ by the recorded value from step (3), then increase i by 1, repeat step (3) until $i = n$
   (e) Return the recorded assignments for all variables $\{x_1, x_2, ..., x_n\}$

   This will lead to n iterations with one call to A per iteration. In each iteration, finding the new formula by plugging the value of a variable takes $O(m)$ time. Hence, we can find the assignment in polynomial time.

7. The graph five-coloring problem is stated as follows: Determine if the vertices of G can be colored using 5 colors such that no two adjacent vertices share the same color. Prove that the five-coloring problem is NP-complete.
   Hint: You can assume that graph 3-coloring is NP-complete

   Solution :

   (a) **Showing that 5-coloring is in NP:**
   Certificate: Each node has a color, i.e color solution exists.
   Certifier:
      i. Check for each edge (u,v), the color of node u is different from the color of node v.

ii. Check at most 5 colors are used.

(b) **Showing that 5-coloring is NP-hard:**

Proving that 3-coloring $\leq_p$ 5-coloring.
Graph construction:
Given an arbitrary graph G. Construct G' by adding 2 new nodes u and v to G. Connect u and v to all nodes that existed in G, and to each other.
Proof-G' can be colored with 5 colors iff G can be colored with 3 colors.

(a) If there is valid 3-color solution for G, say using colors 1,2,3, we want to show there is a valid 5-coloring solution to G'. We can color G' using five colors by assigning colors to G according to the 3-color solution, and then color node u and v by additional two different colors. In this case, node u and v have different colors from all the other nodes in G', and together with the 3-coloring solution in G, we use at most 5 colors to color G'.

(b) If there is a valid 5-coloring solution for G', we want to show there is a valid 3-coloring solution in G. In G', since node u and v connect to all the other nodes in G and to each other, the 5-coloring solution must assign two different colors to node u and v, say colors 4 and 5. Then the remaining three colors 1,2,3 are used to color the remaining graph G and form a valid 3-color solution.

Solving 5-coloring to G' is as hard as solving 3-color to G, then 5-coloring problem is at least as hard as 3-coloring, i.e., 3-coloring $\leq_p$ 5-coloring. So 5-coloring problem is NP-Complete

8. Longest Path is the problem of deciding whether a graph $G = (V, E)$ has a simple path of length greater or equal to a given number k. Prove that the Longest path Problem is NP-complete by reduction from the Hamiltonian Path problem.

Solution :

(a) **Showing that Longest Path Problem is in NP:**
Given a solution path P, we just check that P consists of at least k edges, and that these edges form a path (where no vertex is used more than once). This verification can be done in polynomial time.

(b) **Showing that Longest Path Problem is NP-Complete:**
The reduction follows directly from Hamiltonian Path. Given an instance of Hamiltonian Path on a graph G = (V, E). We create an instance of the longest path problem G' as follows. We use exactly the same graph, i.e G' = G and we set k = V-1.
Proof:- Then there exists a simple path of length k in G' iff G contains a Hamiltonian path.

i. If G contains a Hamiltonian path, then there exists a simple path of length k in G':

Assume G contains a Hamiltonian path, which is a simple path that visits every vertex exactly once. As G' = G and $k = |V| - 1$, where $|V|$ is the number of vertices in G, this Hamiltonian path in G corresponds to a simple path of length $|V| - 1$ in G'. Therefore, if G contains a Hamiltonian path, it implies there exists a simple path of length k in G'.

ii. If there exists a simple path of length k in G', then G contains a Hamiltonian path:
Suppose there exists a simple path of length k in G'. As G' = G, this path visits at least $|V| - 1$ vertices since $k = |V| - 1$. If there is a simple path of length $|V| - 1$ in G', it means that this path visits every vertex exactly once, forming a Hamiltonian path in G.

Hence Longest Path problem is NP-Complete

9. There are a set of courses in USC, each of them requiring a set of disjoint time intervals. For example, a course could require the time from 9am to 11am and 2pm to 3pm and 4pm to 5pm. You want to know, given a number K, if it's possible to take at least K courses. Since you want to study hard and take courses carefully, you can only take one course at any single point in time (i.e. any two courses you choose can't overlap). Show that the problem is NP-complete, which means that choosing courses is indeed a difficult thing in our life. Use a reduction from the Independent set problem.

Solution :

(a) **Showing Problem in NP:** The solution of the problem can be verified in polynomial time (just check the number of the courses in the solution is larger or equal to K, and they don't have time overlap), thus it is in NP.

(b) **Showing Problem is NP-Complete:** Given an independent set problem, suppose the graph has n nodes and asks if it has an independent set of size at least M.We do the construction as follows

$$f(v_i, v_j) = \begin{cases} i * n + j, & \text{if } i \leq j \\ f(v_j, v_i) & \text{otherwise} \end{cases}$$

Now we construct an instance of the course choosing problem, each course corresponds to a vertex of the graph, and if there exists an edge (v_i, v_j) in the original graph, we let the i-th courses require the $f(v_i, v_j)$-th hour. The problem is to determine whether we can choose M courses. Notice that, if there exists an edge (v_i, v_j) in the original graph, then the i-th course and the j-th course will jointly require the $f(v_i, v_j)$-th hour, which means that we can't choose these two courses at the same time. You can verify that any other courses (except i-th and j-th course) will not require $f(v_i, v_j)$-th hour.

Proof:-

i. If the Independent problem is a “yes” instance (has an independent set of size at least M), then we can choose the corresponding courses, and they don’t overlap.
ii. For the other direction, if we can choose the corresponding courses, then it follows that the independent set problem is a “yes” instance.

Thus we can reduce the independent set problem to the course choosing problem in polynomial time.

Since independent set problem is NP-Complete, so the course choosing problem is in NP-Complete.