HW4 - Theoretical Part

Xinyuan Cao
Uni: xc2461

December 1, 2018

1. For an edge $e = (x, y)$, denote the capacity of e as $C((i,j))$, and denote the weight of e as $W((i,j))$.

   The reduction transformation: Given a directed capacitated weighted graph $G = (V, E, C, W)$, we construct the following flow network $G' = (V', E', C', W')$:

   (a) Set $G' = (V, E, C, W)$

   (b) Find all pairs of nodes (i, j) that have multiple directed edges from node i to node j. For each edge $e_t \in E$ from i to j,

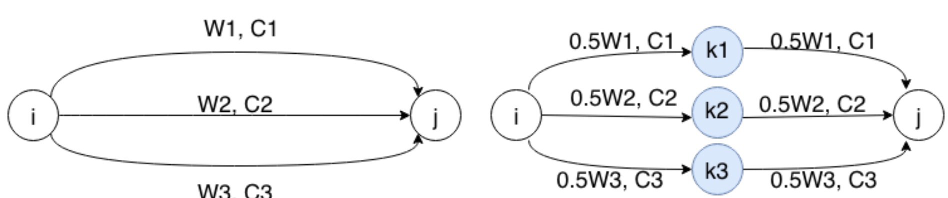
      i. remove edge e_t

      ii. add a dummy node k_t

      iii. add edge (i, k_t), (k_t, j)

      iv. set $C'((i, k_t)) = C'((k_t, j)) = C((i, j))$

      v. set $W'((i, k_t)) = W'((k_t, j)) = \frac{1}{2}W((i, j))$



(a) edges from i to j in G (b) edges from i to j in G'

Figure 1: How to construct G'

This transformation takes $O(m)$ time, so it takes polynomial time.

First I'll prove that the constructed graph G' is a standard network without multi-edges. Consider any edge (i, j) in G'. If $i, j \in G$ and there is only one edge between i, j, then in G', there is still one edge between i, j. If $i, j \in G$ and there are multiply edges between i, j in G, then by our construction, we remove all these edges. So there is no edge between i, j in G'. Otherwise, either i or j or both i and j are the newly added node in G' . By our construction, each node we add only has one in edge and one out edge. So there is at most one edge between i, j in G'. Furthermore G' has all the qualities of a network since it inherits from G. So we know that G' is a standard one without multi-edges.

Then I'll show the equivalence of the instances in min cost flow problem.

Suppose f is the optimal solution to the min-cost flow problem in multi-edges network. Set f' as follows:

$$f'((i,j)) = \begin{cases} f((i,j)) & \text{if } i, j \in G \\ f((i,k)) & \text{if } i \in G,\, j \notin G,\, \exists k \in G \text{ s.t. } (i,k) \in E,\, (i,j),(j,k) \in E' \\ f((k,j)) & \text{if } i \notin G,\, j \in G,\, \exists k \in G \text{ s.t. } (k,j) \in E,\, (k,i),(i,k) \in E' \end{cases}$$

Obviously there is no edge between two newly added nodes.
For flow conservation constraints. We know that for the original node $v \in G$,

$$\begin{aligned} f'^{out}(v) &= \sum_{j \in G, (v,j) \in E'} f'((v,j)) + \sum_{j \notin G, (v,j) \in E'} f'((v,j)) \\ &= \sum_{j:(i,j) \text{has one edge in G}} f((i,j)) + \sum_{j:(i,j) \text{has multiple edges in G}} \sum_{e: e \in E \text{ from } i \text{ to } j} f(e) \\ &= f^{out}(v) \end{aligned}$$

Similarly we can get $f'^{in}(v) = f^{in}(v)$. So we have $f'^{in}(v) = f'^{out}(v)$ for $v \in G$.
For $v \notin G$, it only has one predecessor i and one neighbor j where $i, j \in G$. So here $f'^{in}(v) - f'^{out}(v) = f'((i,v)) - f'((v,j)) = f((i,j)) - f((i,j)) = 0$. So we know f' satisfies flow constraints.

For capacity constraints, it naturally holds for original edges. For new added edges (i,j), where $i \in G$, $j \notin G$, $\exists k \in G$ s.t. $(i,k) \in E$, $(j,k) \in E'$. We know $f'((i,j)) = f((i,k)) \leq C((i,k)) = C'((i,j))$. Similarly it holds for the situation if $i \notin G$ and $j \in G$. So we know that f' satisfies capacity constraints.

Finally I'll prove that f' achieves the optimal.

Claim 1 *Given the construction of flow as above, the total cost of the flow f' in G' is the same as the total cost of f in G.*

Proof

$$
\begin{aligned}
TotalCost(f') &= \sum_{e\in E'} W'(e)f'(e) \\
&= \sum_{e\in E \text{ has one edge}} W(e)f(e) + \sum_{(i,j)\in E \text{ has multiple edges } e_t} \sum_t W'((i,k_t))f'((i,k_t)) \\
&\quad + W'((k_t,j))f'((k_t,j)) \\
&= \sum_{e\in E \text{ has one edge}} W(e)f(e) + \sum_{(i,j)\in E \text{ has multiple edges } e_t} \sum_t \frac{1}{2}W((i,j))f(i,j)) \\
&\quad + \frac{1}{2}W((i,j))f(i,j)) \\
&= \sum_{e\in E \text{ has one edge}} W(e)f(e) + \sum_{(i,j)\in E \text{ has multiple edges } e_t} \sum_t W((i,j))f((i,j)) \\
&= \sum_{e\in E} W(e)f(e) \\
&= TotalCost(f)
\end{aligned}
$$

From claim 1 and the fact that f is the optimal solution. We can conclude that f' is an optimal solution. (Otherwise, if f'_1 has smaller cost in G', we can construct another flow f_1 in G with smaller total cost, which causes a contradiction to the fact that f is optimal.)

On the other hand, suppose f' is the optimal solution of G. Then construct flow f as follows: For $e = (i,j) \in E'$, let $f(e) = f'(e)$. For $e = (i,j)_t \notin E'$, then exists $k_t \in G'$ s.t. $(i,k_t),(k_t,j) \in E'$. Then let $f((i,j)_t) = f'((i,k_t)$. Here t means the t_{th} edge from i to j. It's straightforward to show that f satisfies capacity and supply constraints in G.

Claim 2 *Given the construction of flow as above, the total cost of the flow f in G is the same as the total cost of f' in G'.*

Proof

$$
\begin{aligned}
TotalCost(f) =& \sum_{e \in E} W(e)f(e) \\
=& \sum_{e \in E \text{ has one edge}} W(e)f(e) + \sum_{(i,j) \in E \text{ has multiple edges } e_t} \sum_t W((i,j))f((i,j)) \\
=& \sum_{e \in E \text{ has one edge}} W(e)f(e) + \sum_{(i,j) \in E \text{ has multiple edges } e_t} \sum_t \frac{1}{2} W((i,j))f(i,j)) \\
& + \frac{1}{2} W((i,j))f(i,j)) \\
=& \sum_{e \in E \text{ has one edge}} W(e)f(e) + \sum_{(i,j) \in E \text{ has multiple edges } e_t} \sum_t W'((i,k_t))f'((i,k_t)) \\
& + W'((k_t,j))f'((k_t,j)) \\
=& \sum_{e \in E'} W'(e)f'(e) \\
=& TotalCost(f')
\end{aligned}
$$

From claim2, and the fact that f' is optimal, we have f is also optimal in the multi-edge min-cost flow problem. (Otherwise, if f_1 has smaller cost in G, we can construct another flow f_1' in G' with smaller total cost, which causes a contradiction to the fact that f' is optimal.)

So we know that we can reduce the multi-edge min-cost problem to a standard one. #

2. The decision version of this problem: Given a monotone instance ϕ, and an integer k, can we set k variables to 1 so that ϕ evaluates to 1?

   Next I'll prove that $VC(D) \leq_P$ The Decision Version of this Problem.

   The reduction transformation: Consider an arbitrary instance of the decision version of vertex cover: (G, k). $G = (V, E)$, $|V| = n$, $|E| = m$. Denote $V = \{v_1, \cdots, v_n\}$, $E = \{e_1, \cdots, e_m\}$. Introduce variables $x_1, \cdots, x_n$ to ϕ. For every edge $e = (v_i, v_j) \in E$, introduce the clause $(x_i \vee x_j)$ to ϕ. So we can get a monotone instance with n variables and m clauses. (Because it does not contain negation, and it is in CNF.) Clearly the transformation takes $O(m + n)$ time, which is polynomial time.

   Equivalence of the instances: Then I'll show that the instance (G, k) is a yes instance of VC(D) $\Leftrightarrow$ (ϕ, k) is a yes instance of the decision version of this problem.

   [$\Rightarrow$] Suppose that G has a vertex cover C of size k. Set the literal corresponding to the nodes in C to 1, and the remains to 0. That is, denote $C = \{v_{t_1}, \cdots, v_{t_k}\}$. Then set $x_{t_1}, \cdots, x_{t_k}$ to be 1, and all other variables to be 0.

   This assignment is *valid* because the number of the variables set to 1 is equal to the size of vertex cover, which is k. It also *satisfies* ϕ because for each clause in ϕ, denote as (x_i, x_j). From transformation we know that $e = (x_i, x_j) \in E$. By definition of the vertex cover, either v_i or v_j or both v_i and v_j are in C. So either x_i or x_j or both x_i and x_j are set to 1. So we have $x_i \vee x_j = 1$. Hence each clause in ϕ is satisfiable. And therefore we know that ϕ evaluates to 1.

   [$\Leftarrow$] Now suppose that there is a satisfying truth assignment of ϕ with k variables set to 1. Denote these k variables as $x_{t_1}, \cdots, x_{t_k}$. Construct a set $C = \{v_{t_1}, \cdots, v_{t_k}\}$. Clearly $|C| = k$. Then I'll prove C is a vertex cover of G. Because $x_1, \cdots, x_n$ are a satisfying truth assignment of ϕ, so each clause of ϕ is satisfiable. $\forall e = (v_i, v_j) \in E$, consider the clause $(x_i \vee x_j)$ in ϕ. (This clause exists because of our reduction transformation.) So we know that this clause evaluates 1, so either x_i or x_j or both x_i and x_j are set to 1. So either v_i or v_j or both v_i and v_j are in C. Therefore we have proved that every edge in E has at least one end point in C, so C is a vertex cover of G in size k.

   Since $VC(D)$ is NP-complete, we know the decision version of this problem is NP-complete. Because of the rough equivalence of decision and original problems, we conclude that this problem is NP-complete. #

3. The decision version of this problem: Given a large store with m customers and n products and maintains an $m \times n$ matrix A such that $A_{ij} = 1$ if customer i has purchased product j; otherwise, $A_{ij} = 0$. Is there a subset of orthogonal customers of size k?

   Then I'll show that $IS(D) \leq_P$ The Decision Version of this Problem.

   The reduction transformation: Consider an arbitrary instance of the decision version of independent set (G, k). $G = (V, E)$, $|V| = m$, $|E| = n$. Denote $V = \{v_1, \cdots, v_m\}$, $E = \{e_1, \cdots, e_n\}$. Denote $e_j = (v_{j_1}, v_{j_2})$. Then for this problem, denote the customers as $\{x_1, \cdots, x_m\}$. Denote the products as $\{p_1, \cdots, p_n\}$. Initialize the matrix $A = 0$. Then for each edge $e_j = (v_{j_1}, v_{j_2}) \in G$, let $A_{j_1,j} = A_{j_2,j} = 1$.
   Clearly the transformation takes $O(m + n)$ time, which is polynomial time.

   Equivalence of the instances: Then I'll show that the instance (G, k) is a yes instance of IS(D) $\Leftrightarrow$ (ϕ, k) is a yes instance of the decision version of this problem.

   [$\Rightarrow$] Suppose that G has an independent set S of size k. $S = \{v_{t_1}, \cdots, v_{t_k}\}$. Then I'll show that the subset of customers $Q = \{x_{t_1}, \cdots, x_{t_k}\}$ are orthogonal customers by contradiction. For $\forall 1 \leq i < j \leq k$, assume that x_{t_i} and x_{t_j} purchased the product p_q in common. Then by definition of the matrix A we will have $A_{t_i,q} = A_{t_j,q} = 1$. From our reduction transformation we know that $A_{i,j} = 1$ if and only if the edge e_j connects v_i. So we get the edge e_q connects v_{t_i} and v_{t_j}. This leads to a contradiction to v_{t_i} and v_{t_j} are from an independent set S. Therefore we know that customers x_{t_i} and x_{t_j} did not buy any products in common, which means they are orthogonal. So we know that the customers in subset Q are orthogonal customers. And $|Q| = k$.

   [$\Leftarrow$] Now suppose that we have a subset of orthogonal customers $Q = \{x_{t_1}, \cdots, x_{t_k}\}$. We claim that $S = \{v_{t_1}, \cdots, v_{t_k}\}$ is an independent set of G. We also prove this by contradiction. Assume that $\exists 1 \leq i < j \leq n$ s.t. there is an edge $e_q = (v_{t_i}, v_{t_j}) \in E$. Then from transformation we have $A_{t_i,q} = A_{t_j,q} = 1$. That means customer x_{t_i} and customer x_{t_j} both purchased the product q, which results in the contradiction to x_{t_i} and x_{t_j} are orthogonal customers. Therefore we know that for each two nodes in S, there is no edge between them. So S is an independent set of G, with size k.

   Since IS(D) is NP-complete, we know the decision version of this problem is NP-complete. Because of the rough equivalence of decision and original problems, we conclude that this problem is NP-complete. #

4. I'll reduce this problem to a max-flow problem. We want to construct a graph G. First add n nodes $\{x_1, \cdots, x_n\}$, where x_i is corresponding to the i_{th} person. Then add another k nodes $\{y_1, \cdots, y_k\}$, where y_j is corresponding to the j_{th} hospital. For $\forall 1 \le i \le n$, $1 \le j \le k$, if the j_{th} hospital is within a half-hour's driving time of the i_{th} person, then add an edge (x_i, y_j), the capacity $c((x_i, y_j)) = 1$. Then add a source node s. For every x_i, add an edge (s, x_i), the capacity $c((s, x_i)) = 1$. Then add a sink node t. For every y_j, add an edge (y_j, t), the capacity $c((y_j, t)) = \lceil \frac{n}{k} \rceil$.

   Then I'll prove the equivalence. We claim that there is a feasible assignment of injured people that meets all demands if and only if there is a max flow of value n.

   [⇒] Given a feasible assignment, we'll define a flow in G. Initialize the flow $f(e) = 0$ for every edge e. For each injured person, if the i_{th} person is assigned to the j_{th} hospital, then let $f((x_i, y_j)) = 1$. For $1 \le i \le n$, set $f((s, x_i)) = 1$. For $1 \le j \le k$, set $f((y_j, t))$ is the number of people the j_{th} hospital receives. Then we consider the y_j, $\forall 1 \le j \le k$. We know

$$\begin{aligned} f^{in}(y_j) &= \sum_{i:i_{th} \text{ person goes to } j_{th} \text{ hospital}} f(x_i, y_j)) \\ &= \sum_{i:i_{th} \text{ person goes to } j_{th} \text{ hospital}} 1 \\ &= \# \text{ people that go to the } i_{th} \text{ hospital} \\ &= f((y_j, t)) \\ &= f^{out}(y_j) \end{aligned}$$

   So we know that f satisfies the flow conservation constraints.

   Since the assignment meets all the demand, we know that $\forall 1 \le i \le n$, $f((s, x_i)) = 1 = c((s, x_i))$. $\forall 1 \le j \le k$, $f((y_j, t))$ =the number of people the j_{th} hospital receives $\le \lceil \frac{n}{k} \rceil = c((y_j, t))$. $\forall 1 \le i \le n$, $\forall 1 \le j \le k$, $f((x_i, y_j)) \le 1 = c((x_i, y_j))$. So f also satisfies the capacity constraints.

   $|f| = \sum_{1 \le i \le n} f((s, x_i)) = n$. This is the max flow because the capacity of the cut $(\{s\}, V - \{s\})$ is n. So we prove that there is a max flow of value n.

   [⇐] Given a max flow f of value n inn G. Since the capacity are integer, the max flow we get is also integer. Then we construct a feasible assignment that satisfies all the requirements: for each edge (x_i, y_j), if $f((x_i, y_j)) = 1$, then we assign the i_{th} people to the j_{th} hospital. (Since $c((x_i, y_j)) = 1$, we know that $f((x_i, y_j))$ can only be 0 or 1.) The value of the flow equals flow out of s and the capacity of the cut $(\{s\}, V - \{s\})$ is n. So every edge out of s is saturated. From the flow conservation constraint we know that $\forall 1 \le i \le n$, $f^{out}(x_i) = f^{out}(x_i) = 1$. So we ensure every injured person is assigned to exactly one hospital. Then for each hospital, by flow conservation constraint and capacity constraints we know that $\forall 1 \le j \le k$, $f^{in}(x_i) = f^{out}(x_i) \le \lceil \frac{n}{k} \rceil$. Here $f^{in}(x_i)$ = the number of people that go to the i_{th} hospital. So we know that this assignment meets the requirement that every hospital receives at most $\lceil \frac{n}{k} \rceil$ people. So we get a feasible assignment.

So we have successfully reduce the original problem to a max flow problem. We can solve the max flow problem using Ford-Fulkerson Algorithm with input (G, s, t). If the max flow value is equal to n, then we know that there is a feasible assignment: for each edge (x_i, y_j), if $f((x_i, y_j)) = 1$, then we assign the i_{th} people to the j_{th} hospital. If the max flow value is less than n, then we have no feasible assignment.

This reduction needs polynomial time $O(nk)$. And we know that max-flow problem is efficient, so the original problem can be solved in polynomial time $O(n^2(n+k))$. So this algorithm is efficient. #

5. (i) Denote a_{ij} as the indicator whether the employee i is assigned the position j. That is, if the manager assign the position j to the employee i, then $a_{ij} = 1$, otherwise $a_{ij} = 0$. Then we formulate an integer program as follow.

$$\begin{aligned} \max \quad & \sum_{1\le i\le m, 1\le j\le n} s_{ij} a_{ij} & \\ \text{s.t.} \quad & \sum_{1\le i\le m} a_{ij} \le 1 & \forall 1 \le j \le n \\ & \sum_{1\le j\le n} a_{ij} \le 1 & \forall 1 \le i \le m \\ & a_{ij} \in \{0,1\} & \forall 1 \le i \le m, 1 \le j \le n \end{aligned}$$

(ii) We can reduce this problem to a min-cost flow problem. We need to construct a graph G. Add nodes x_i, $1 \le i \le m$. Add nodes y_j, $1 \le j \le n$. Add a source node s, and a sink node t. For $1 \le i \le m$, add an edge (s, x_i), with capacity $c((s, x_i)) = 1$, cost $w((s, x_i)) = 0$. For $1 \le j \le n$, add an edge (y_j, t), with capacity $c((y_j, t)) = 1$, cost $w((y_j, t)) = 0$. For $1 \le i \le m$, $1 \le j \le n$, add an edge (x_i, y_j), with capacity $c((x_i, y_j)) = 1$, weight $w((x_i, y_j)) = -s_{ij}$. So we get a network $G = (V, E, c, w)$, source s and sink t. The formulation takes polynomial time. We want to minimize the total cost of the flow. (min-cost flow)

So the above IP will find the the min-cost flow f. More detailedly, if $a_{ij} = 1$, then the $f((s, x_i)) = f((x_i, y_j)) = f((y_j, t)) = 1$, and there is no flow on other edges. So in IP we want to maximize the sum of $s_{ij} a_{ij}$, that is to minimize the cost of the flow. (Since $w_{ij} = -s_{ij}$). So the solution of the IP is the min-cost flow, where $a_{ij} = f(x_i, y_j)$. And the corresponding objective function will be negative the min-cost.

(iii) If $s_{ij} = 1$, then the resulting IP finds the max bipartite matching. We construct a bipartite graph $G = (X \bigcup Y, E)$. $X = \{x_1, \cdots, x_m\}$, $Y = \{y_1, \cdots, y_n\}$. For $\forall q \le i \le m$, $1 \le j \le n$, add an edge (x_i, y_j). The result of the IP will give us the max bipartite matching of G. If $s_{ij} = 1$, then add the pair (x_i, y_j) to the pair. Since we have the constraints, so we will get a valid matching. Also because we want to maximize the objective function, we will get the maximum bipartite matching.

(iv) (i) The integer program for this instance is:

$$\begin{aligned}
\max \quad & a_{11} + a_{12} + a_{13} + a_{21} + a_{22} + a_{23} \\
\text{s.t.} \quad & a_{11} + a_{12} + a_{13} \le 1 \\
& a_{21} + a_{22} + a_{23} \le 1 \\
& a_{11} + a_{21} \le 1 \\
& a_{12} + a_{22} \le 1 \\
& a_{13} + a_{23} \le 1 \\
& a_{ij} \in \{0, 1\}, \quad \forall 1 \le i \le 2, 1 \le j \le 3
\end{aligned}$$

Its relaxed LP is:

$$\begin{aligned}
\max_{a_{ij} \ge 0} \quad & a_{11} + a_{12} + a_{13} + a_{21} + a_{22} + a_{23} \\
\text{s.t.} \quad & a_{11} + a_{12} + a_{13} \le 1 \\
& a_{21} + a_{22} + a_{23} \le 1 \\
& a_{11} + a_{21} \le 1 \\
& a_{12} + a_{22} \le 1 \\
& a_{13} + a_{23} \le 1
\end{aligned}$$

It is suffice to constrain all variables to be non-negative, and is free to omit the upper bound 1 of a_{ij} because the constraints that sum of variables are no greater 1, and that all variables are non-negative ensure that all the variables are definitely no greater than 1. So we only constrain $a_{ij} \ge 0$. Then I'll take the dual of the relaxed LP. I'll rewrite the relaxed LP as follows. Let $x = (a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23})^T$, $b = c = (1, 1, 1, 1, 1, 1)^T$. Define

$$A = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

So the relaxed LP can be written as

$$\begin{aligned}
\max_{x \ge 0} \quad & c^T x \\
\text{s.t.} \quad & Ax \le b
\end{aligned}$$

Then we can give the dual:

$$\begin{aligned}
\min_{y \ge 0} \quad & b^T y \\
\text{s.t.} \quad & A^T y \ge c
\end{aligned}$$

That is:

$$\begin{aligned}
\min_{y_i \geq 0, 1 \leq i \leq 5} \quad & y_1 + y_2 + y_3 + y_4 + y_5 \\
\text{s.t.} \quad & y_1 + y_3 \geq 1 \\
& y_1 + y_4 \geq 1 \\
& y_1 + y_5 \geq 1 \\
& y_2 + y_3 \geq 1 \\
& y_2 + y_4 \geq 1 \\
& y_2 + y_5 \geq 1
\end{aligned}$$

Assume the dual LP has integer solution. Then it could solve the minimum vertex cover.
First I'll prove that the integer solution $\{y_i\}$ can only be 0 or 1. From the constraints we know that $y_i \geq 0$. Assume that for the optimal solution $\{y_i\}_{i=1}^5$, $\exists t$ s.t. $y_t \geq 2$. Then we consider the solution $\{y_i'\}_{i=1}^5$ so that $y_i' = y_i$ if $i \neq t$, and $y_t' = 1$. Clearly $\{y_i'\}_{i=1}^5$ also meets all the constraints, and can achieve a smaller objective function than $\{y_i\}_{i=1}^5$. This causes a contradiction to $\{y_i\}_{i=1}^5$ is the optimal solution. Therefore we know that $\{y_i\} \leq 1$. So the integer solution $\{y_i\}$ can only be 0 or 1.
Let $G = (V, E)$. $V = \{v_1, \cdots, v_5\}$, $E = \{(v_1, v_3), (v_1, v_4), (v_1, v_5), (v_2, v_3),$ $(v_2, v_4), (v_2, v_5)\}$. Consider a subset $S \subset V$. Let $y_i = \begin{cases} 1 & \text{if } v_i \in S \\ 0 & \text{o.w.} \end{cases}$
For a vertex cover S we need that for each edge (v_i, v_j), we have $y_i + y_j \geq 1$ so that it has an end point in S. So we know that this dual LP is solving the minimum vertex cover problem. #