Kliment Mamykin, UNI 2770

Algorithms for Data Science, Homework 4

Problem 1

In a directed weighted and capacitated graph $G = (V, E, C, W)$, define a "extra" edge set D as edges that connect nodes already connected by some other edge. Those are the edges (in D) that need to be addressed in the following transformation.

Main idea: In solving a min cost flow problem, any edge in the directed graph can be replaced with two sequential edges by introducing a supplementary node and setting the capacities and costs of the edges as described next. If the equivalence is proven, then each edge that is in the "extra" edge set D can be split into two, resulting in a graph with at most one edge between each two nodes. Such graph can be used in the known algos (such as LP) in finding min cost flow that will be equivalent to the original graph.

Consider any edge $e : (u, v) \in E$ with capacity $C_{u,v}$ and cost/weight $W_{u,v}$. Introduce a supplementary node k with 0 demand/supply value and replace edge (u, v) with two sequential edges (u, k) and (k, v). Set capacity $C_{u,k} = C_{k,v} = C_{u,v}$. Set weights such that $W_{u,k} + W_{k,v} = W_{u,v}$, for example $W_{u,k} = W_{k,v} = W_{u,v}/2$. Such transformation:

a) removes direct connection from u to v by construction

b) preserves the avaliable capacity of the flow between u and v by construction

c) preserves the cost of flow from u to v. In the original graph the cost of flow is $W_{u,v}f_{u,v}$, in the transformed graph both introduced edges contribute to the cost. Both edges have the same flow value (due to the flow preservation constraint at node k). Both edges together cost $W_{u,k}f_{u,k} + W_{k,v}f_{k,v} = W_{u,k}f_{u,v} + W_{k,v}f_{u,v} = (W_{u,k} + W_{k,v})f_{u,v} = W_{u,v}f_{u,v}$

and therefore is equivalent to the original graph when solving for the min cost flow.

Final reduction: for every edge in D replace with two edges as described above.

Note: Theoretically, every edge in the original can be split into two, but it will introduce a lot of unnecessary edges (and variables in the corresponding LP formulation), so practically only extra edges for any two nodes need splitting.

Note: There are a couple of ways to split the weights between two new edges, e.g. $W_{u,k} = 1, W_{k,v} = W_{u,v} - 1$ might work, but that assumes the original weight to be at least 1 (so no negative weights introduced). However a split with one edge weight = 0 will work $W_{u,k} = 0, W_{k,v} = W_{u,v}$ for all weights.

Problem 2

Define $MinMonotone$ problem: given a monotone formula ϕ^m in CNF, what is the minimal number of variables that need to be set to True to satisfy the formula.

Decision version of the problem $Monotone(D) : (\phi^m, k) \implies yes/no$, given a ϕ^m and a target k, is there a truth assignment with at most k true variables that satisfies ϕ^m.

$Monotone(D)$ is a NP problem: We can define a certifier $B(x, t)$ where x is an instance of $Monotone(D)$ problem (ϕ^m, k) and t is a truth assignment for variables. Validation can be performed in poly time first by comparing the number of truth vars in t to k, and then evaluating the formula ϕ^m with t (can be done in poly time). The size of $|t| \leq p(|x|)$ - the size of the truth assignment is at most n elements (given ϕ^m having n variables, m clauses).

$Monotone(D)$ is a NP-hard: We can reduce a arbitrary instance of $VC(D)$ (NP-complete) problem to a $Monotone(D)$ problem and show $VC(D) \leq p\, Monotone(D)$

Proof: Given an instance $a = (G, k)$ of $VC(D)$, transform it to an instance $b = (\phi^m, k)$ of $Monotone(D)$ like follows. Define a variable x_i for each node in graph G, and group them in clauses of two literals, one clause for each edge in G. For example a graph with edges $\{(1, 2), (2, 3), (3, 4), (4, 1)\}$ will be transformed to a monotonic formula $\phi^m = (x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (x_3 \vee x_4) \wedge (x_4 \vee x_1)$. This reduction can be done in poly time $O(|E|)$.

Now, given a solution to a, which is a set of vertices S of size $|S| = k$, form a solution to b by setting x_i to true iif the node $i \in S$. Since the vertices in S form a vertice cover, for every edge in G there is at least one node that is in S, and by construction of the formula every clause will have at least one literal evaluating to true, and the whole formula will evaluate to true.

Given a solution to problem b, which is a truth assignment, means that at least one of the literals in each clause evaluates to true for the formula to evaluate to true. And by construction at least one of nodes in each graph edge is selected in the set S, covering each edge.

This reduction proves that $VC(D) \leq p\, Monotone(D)$, and $Monotone(D) \in NP$, therefore $Monotone(D)$ is NP-complete.

Problem 3

The problem of finding the maximum subset of orthogonal customers in NP-complete. It can be proven by reducing a decision version of Independent Set problem $IS(D)$ which is known to be NP-complete to a decision version of the Orthogonal Customers Subset $OCS(D)$ problem.

Define decision version of Orthogonal Customers Subset $OCS(D)$ problem as: given a matrix of purchases $A_{i,j}$ of size $m \times n$ where $A_{i,j} = 1$ if customer i bought product j, and a number k, is there an orthodonal set of customers (where no two customers bought any products in common) of size at least k?

First, lets define a customer **overlap matrix** L, size $m \times m$, where $L_{i,j} = 1$ iif customer i bought at least one common product with customer j, and 0 if customer i is orthogonal to customer j. L can be computed in polynomial time $O(m^2 n)$ as:

$$L = 1(AA^T > 0)$$
$$L_{ij} = \begin{cases} 1 & , (AA^T)_{ij} > 0, \text{overlapping customers i and j} \\ 0 & , (AA^T)_{ij} = 0, \text{orthogonal customers i and j} \end{cases}$$

However we can also show that given an overlap matrix L we can compute some purchases matrix A', such that $1(A'A'^T > 0) = L$ in polinomial time. Construct matrix A' of size $m \times m^2$ such that each column represents a cell in L (we have m^2 cells in L and columns in A'), and $A'_{ik} = A'_{jk} = 1$ if for a kth cell located at row i column j in L has value $L_{ij} = 1$. Matrix product $A'A'^T$ will produce and $m \times m$ matrix where (i, j) element is an inner product of row A'_i and A'_j, and will be 0 if there cells $(i, j), (j, i)$ in L contained 0, and >0 otherwise. There is some redundancy in A' because matrix L is symmetric and some columns can be dropped, but the size and construction time will still be polynomial $O(m^3)$. For example:

$$L = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$
$$A' = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$
$$A'A'^T = \begin{bmatrix} 7 & 2 & 2 & 2 \\ 2 & 3 & 0 & 0 \\ 2 & 0 & 3 & 0 \\ 2 & 0 & 0 & 3 \end{bmatrix}$$
$$1(A'A'^T > 0) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} = L$$

$OCS(D) \in NP$ since given a certificate S of set of orthogonal customers, we can verify the problem is satisfied by first comparing $|S| = k$ and then verifying all customers in S are pairwise orthogonal. The comparison will take poly time $O(|S|^2 n)$.

Reduction: Let $x = (G = (V, E), k)$ be an instance of $IS(D)$. Let each vertex represent a customer, and each edge represent that there is an overlap between the customers representing the vertices. Construct an adjacency matrix L of size $|V| \times |V|$. Treat the adjacency matrix L as customer overlap martix, and construct a product purchase matrix A' as described above. Now we have an instance of $OCS(D)$ problem (with the same k). This concludes the reduction.

A satisfying solution to x is an independent set of vertices S. Since there are no edges between any two vertices in the set, the adjacency matrix will have 0 in every cell (i, j) where i and $j \in S$. Since the adjacency matrix is treated as an overlap matrix, 0s in cells mean there is no overlap between customers i and j, therefor no overlap between any two customers representing vertices in S. Therefore those customers are an orthogonal set.

A satisfying set of orthogonal customers can be similarly converted to an independent set of vertices satisfying x.

$$OCS(D) \in NP, IS(D) \leq pOCS(D) \implies OCS(D) \in NP - complete$$

Problem 4

This problem can be solved efficiently with Max Flow aglorithm (e.g. Ford–Fulkerson algorithm).

Form a graph G with n vertices representing injured people, k vertices representing hospitals, and directed edges (u, v) if an injured person u is within 30 min from hospital v, set the edges capacity to 1. Add source node s with edges to each of the n people nodes, set capacity to 1. Add sink node t, and edges from all hospital nodes to t with capacity $\lceil n/k \rceil$. The problem has a solution iif the max flow value of the network $|f| = n$, meaning each person got assigned to a hospital. The edges between people and hospitals with the flow value = 1 are the people to hospitals assignments. The hospitals to t edges constraints ensure no hospital is overloaded.

Problem 5

i

IP problem formulation:

Let E be a set of pairs (i, j) representing an employee i being interested in a job j.

Vaiables $x_{i,j} \in \{0, 1\}, \forall (i, j) \in E$, representing job assignment of employee i to job j, if $x_{i,j} = 1$ then an employee i is given the job j, and $x_{i,j} = 0$ if no job given. The objective of IP is:

$$\max \sum_{(i,j) \in E} s_{i,j} x_{i,j}$$

Constraints:

$$\text{At most one employee per job constraint: } \sum_{i:(i,j) \in E} x_{i,j} \leq 1 \qquad 0 \leq j \leq n$$

$$\text{At most one job per employee constraint: } \sum_{j:(i,j) \in E} x_{i,j} \leq 1 \qquad 0 \leq i \leq m$$

$$\text{Integer constraint: } x_{i,j} \in \{0, 1\} \qquad \forall (i, j) \in E$$

ii

Problem can be represented as a weighted bipartite graph $G = (L \cup R, E, w)$, where L is a set of employees, and R is a set of jobs, with E as a set of edes representing employee i interested in job j. The weights w represent a set of scores $s_{i,j}$ assigned by supervisor. The problem is to find a bipartite graph matching such that the sum of weights of matching edges is maximized. The solution found by the above IP will be a set of $x_{i,j}$ that indicate edges in E, in other words the job assignment that maximizes the score.

$$x_{i,j} = \begin{cases} 1 & \text{if employee i assigned to job j} \\ 0 & \text{otherwise} \end{cases}$$

iii

If $s_{i,j} = 1 \ \forall (i,j) \in E$, the max weight bipartite matching reduces to a finding a maximum bipartite matching, which can be solved by augmenting the graph with nodes $\{s, t\}$ and connecting s to the set of employees L, connecting the set of jobs R to t, with all edges having capacity 1. The solution found by the above IP will be the max flow for such graph, and edges in E having flow value = 1 will indicate a job assignment.

iv.i

IP formulation:

$$\begin{aligned} &\text{Variables: } x_{i,j}, \forall 1 \le i \le 2, 1 \le j \le 3 \\ &\text{Objective: } \max \sum_{i=1}^{2} \sum_{j=1}^{3} x_{i,j} \\ &\text{s.t. } x_{11} + x_{12} + x_{13} \le 1 \\ &\quad x_{21} + x_{22} + x_{23} \le 1 \\ &\quad x_{11} + x_{21} \le 1 \\ &\quad x_{12} + x_{22} \le 1 \\ &\quad x_{13} + x_{23} \le 1 \\ &\quad x_{i,j} \in \{0, 1\}, \forall 1 \le i \le 2, 1 \le j \le 3 \end{aligned}$$

LP with relaxation of integrality constraints:

$$\begin{aligned} &\text{Variables: } x_{i,j}, \forall 1 \le i \le 2, 1 \le j \le 3 \\ &\text{Objective: } \max_{x_{i,j} \ge 0} \sum_{i=1}^{2} \sum_{j=1}^{3} x_{i,j} \\ &\text{s.t. } x_{11} + x_{12} + x_{13} \le 1 \\ &\quad x_{21} + x_{22} + x_{23} \le 1 \\ &\quad x_{11} + x_{21} \le 1 \\ &\quad x_{12} + x_{22} \le 1 \\ &\quad x_{13} + x_{23} \le 1 \\ &\quad x_{i,j} \le 1 \end{aligned}$$

The last line of constraints $x_{i,j} \le 1$ can be dropped as it's incorporated into the first 5 constraints. If any of the $x_{i,j}$ was greater then 1, the othere constraints could not be satisfied.

Primary LP in matrix/vector form - re-index (vectorize) variables to be one vector $x_k, 1 \le k \le 6$:

$$\text{Variables: } x_k, 1 \leq k \leq 6, \triangleq \vec{x}$$

$$\text{Objective: } \max_{\vec{x} \geq 0} \vec{\mathbb{I}}^T \vec{x} = \min \vec{c}^T \vec{x}$$

$$\text{s.t. } \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \vec{x} = A\vec{x} \leq \vec{\mathbb{I}} = \vec{b}$$

Dual LP:

$$\text{Variables: } y_l, 1 \leq l \leq 5, \triangleq \vec{y}$$

$$\text{Objective: } \min_{\vec{y} \geq 0} \vec{b}^T \vec{y} = \min_{\vec{y} \geq 0} \vec{\mathbb{I}}^T \vec{y} = \min_{y_l \geq 0} \sum_{l=1}^{5} y_l$$

$$\text{s.t. } A^T \vec{y} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} \vec{y} \geq \vec{c} = \vec{\mathbb{I}}$$

$$\Downarrow$$

$$\begin{array}{ccccccc} +y_1 & & +y_3 & & & \geq & 1 \\ +y_1 & & & +y_4 & & \geq & 1 \\ +y_1 & & & & +y_5 & \geq & 1 \\ & +y_2 & +y_3 & & & \geq & 1 \\ & +y_2 & & +y_4 & & \geq & 1 \\ & +y_2 & & & +y_5 & \geq & 1 \end{array}$$

iv.ii

If the above LP has an integral solution, its variables will be binary variables:
$y_k \in \{0, 1\}, 1 \leq l \leq 5$. Since there are 5 binary variables, and there are 5 nodes in the graph that we to create this IP/LP, the variables encode some selection criteria of nodes into a set. There are 6 constraints (one per each edge) that seem to encode that at least one node in each edge is included in the set. The objective is a minimization of the number of nodes included in the set. From this we can conclude that the graph problem the dual LP is solving is a minimum vertex cover problem.