ShanghaiTech University

CS240 Algorithm Design and Analysis

Spring 2023

Problem Set 4

Due: 23:59, April 30, 2023

1. Submit your solutions to the course Blackboard.
2. If you want to submit a handwritten version, scan it clearly.
3. Late homeworks submitted within 24 hours of the due date will be marked down 25%. Homeworks submitted more than 24 hours after the due date will not be accepted unless there is a valid reason, such as a medical or family emergency.
4. You are required to follow ShanghaiTech's academic honesty policies. You are allowed to discuss problems with other students, but you must write up your solutions by yourselves. You are not allowed to copy materials from other students or from online or published resources. Violating academic honesty can result in serious penalties.

Note: When proving that a problem A is NP-complete, clearly divide your answer into three steps:

1. Prove that A is in NP.
2. Choose an NP-complete problem B. For any instance of B, construct an instance of problem A, so that the yes/no answers to the two instances are the same.
3. Show your construction runs in polynomial time.

Problem 1:

Suppose you are the manager of a clothing store which sells j different jackets and h different shirts. c customers come to the store, and each of them has a set of jackets and shirts which they like. Each customer will either buy one jacket and one shirt from the set he likes, or nothing at all. The customer is satisfied if he buys one jacket and one shirt. In addition, each jacket or shirt can be sold to at most one customer. Design an efficient algorithm which maximizes the number of satisfied customers.

Solution:

Construct the network with 6 layers of nodes:

1. Source node s.
2. j jacket nodes, each connected to the source node with capacity 1.
3. c customer nodes, each split into two nodes connected by an edge with capacity 1. The two nodes are defined as $c_{i-\text{left}}$ and $c_{i-\text{right}}$. $c_{i-\text{left}}$ connects to the jacket nodes which the ith customer likes, and $c_{i-\text{right}}$ connects to the trouser nodes which the ith customer likes. The capacity of each edge is also 1.
4. t trouser nodes, each connected to the sink node with capacity 1.
5. Sink node t.

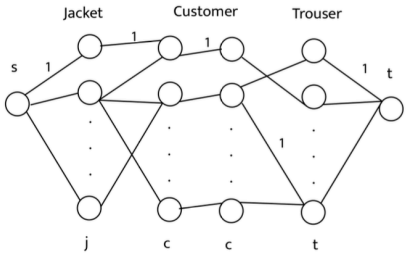

Figure 1: Network structure

Run maxflow to determine the maximum number of successful deals.

Problem 2:

Consider a directed graph $G = (V, E)$. Let $X \subseteq V$ be a set of *start nodes*, and $Y \subseteq V$ be a set of *destination nodes*, and assume that X and Y are disjoint. Your goal is to find a set of paths, where each path starts at a node in X and ends at a node in Y, and none of the paths share an edge. Design an efficient algorithm to determine if such a set of paths exist, and find such a set if possible.

Solution:

We turn G into a flow network as follows. Let k = |X |. We give each edge a capacity of 1, we add a super-source s with capacity-1 edges to each node in X , and we add a super-sink t with a capacity-k edge from each node in S. We then check whether the maximums s-t flow has a value of k.

If so, then it has an integer-valued one, and by deleting s and t, we get the desired set of disjoint paths from X to S. Conversely, if there is a solution to the escape problem, then by sending one unit of flow to each node in X, one unit of flow along each of the paths in the escape solution and c units of flow to t from each node in S that is at the head of c paths, we get a flow of value k.

Problem 3:

SIST wants to hire graduate students to TA courses. Each course needs exactly two TAs, and there is a list of pairs of students who can TA a course together (i.e. if two students are not a pair from the list, they cannot TA a course together). For a given input value k, we want to determine whether there exists a *TA cycle* with k distinct students $s_1, ..., s_k$, such that s_1 and s_2 TA a course together, s_2 and s_3 TA a course together, ..., and s_k and s_1 TA a course together (note that each student in the cycle will TA two courses). Show that this problem is NP-complete.

Solution:

NP problem: We can just check whether there is a simple TA cycle containing k TAs in polynomial time, so it's NP.

Reduction: Directed Hamiltonian Cycle p this problem.

Construction: We construct graph G = (V, E) and TA cycle C, where each vertex V_i represents each TA A_i, and a directed path from V_i to V_j represents A_i works as a TA for A_j in some course. We let K = |V|. Then we prove that G is a Directed Hamiltonian Cycle iff C contains a simple TA cycle.

⇒: If G is a Directed Hamiltonian Cycle with vertex |V|, then there exists a simple TA cycle containing K = |V | TAs.

⇐: If C is a TA cycle containing a simple TA cycle with K TAs, then there exists a Directed Hamiltonian Cycle.

So the problem is NP-complete.

Problem 4:

In the Knapsack problem, we have n items, and the j'th item has weight w_j and value v_j ($j = 1, \ldots, n$). All w_j, v_j values are positive integers. The question is whether there exists a subset of the n items with total weight at most W and total value at least V, for input values W and V. Show that Knapsack is NP-complete.

Solution:

1. Knapsack is in NP. It takes linear time to add the weights and value...

2. Given an instance of Subset Sum: positive integer numbers $s_1, s_2, ..., s_n$ and sum t. We can construct an instance of Knapsack where $a_i = c_i = s_i$ and $b = k = t$. The following deduction implies the new problem is equivalent to the original problem.

$$\begin{cases} \sum_{i \in S} a_i \leq b \Leftrightarrow \sum_{i \in S} s_i \leq t \\ \sum_{i \in S} c \geq k \Leftrightarrow \sum_{i \in S} s_i \geq t \end{cases} \Leftrightarrow \sum_{i \in S} s_i = t$$

If we can solve the new instance of Knapsack, we can find a subset $S \subseteq \{1, 2, 3, ..., n\}$, which will also satisfy the instance of Subset Sum. Meanwhile, if we can find such S of Subset Sum, it is also the solution of the Knapsack instance.

Problem 5:

Let C be a finite set and let S be a collection of subsets of C. The Set Packing problem asks whether, for an input value k, there exist k sets from S which are pairwise disjoint (i.e. no two sets share an element). Show that Set Packing is NP-complete, using a reduction from the Independent Set problem.

Solution:

1. Given a set of K subsets in the collection, there exists a poly-time certifier that checks whether: all $\frac{1}{2}K(K-1)$ pairs in subsets are pairwise disjoint. Therefore SET-PACKING problem is in NP.

2. Known that Independent Set is NP-complete, we will show that SET-PACKING problem is NP-complete by a polynomial-time reduction from Independent Set. Given an instance of Independent Set, we will construct an instance of SET-PACKING problem. W.L.O.G assuming that the graph G =(C, E). For every node $c_i \in C$, we define a subset $S(c_i)$ to be the set of all edges incident with c.
Here the subsets S(c) corresponds to the collection of subsets of C. We notice that two subsets $S(c_1)$ and $S(c_2)$ are disjoint if and only if c_1 and c_2 are not adjacent. Therefore some K subsets in the collection are pairwise disjoint if and only if we can find K pairwise disjoint subsets $S(c_i)$ from $S(c)$.

3. Independent Set is NP-complete and Independent Set can reduce to SET-PACKING problem, hence SET-PACKING problem is NP-complete.

Problem 6:

Suppose you are organizing a summer sports camp which offers n different sports, and you need hire camp counselors who are skilled at these sports. You receive applications from m people, where each person is skilled at some subset of the sports. For an input value $k \leq m$, you want to determine whether it is possible to hire k people, so that for each sport you hire at least one person skilled in that sport. Show that this problem is NP-complete, using a reduction from the Vertex Cover problem.

Solution:

The problem is in NP since, given a set of k counselors, we can check that they cover all the sports.

Suppose we had such an algorithm A; here is how we would solve an instance of Vertex Cover. Given a graph $G = (V, E)$ and an integer k, we would define a sport S_e for each edge e and a counselor C_v, for each vertex v. C_v is qualified in sport S_e if and only if e has an endpoint equal to v.

Now, if there are k counselors that,together, are qualified in all sports, the corresponding vertices in G have the property that each edge has an end in at least one of them;so they define a vertex cover of size k. Conversely if there is a vertex cover of size k, then this setof counselors has the property that each sport is contained in the list of qualifications of at least one of them.

Thus,G has a vertex cover of size at most k if and only if the instance of Efficient Recruiting that we create can be solved with at most k counselors. Moreover,the instance of Efficient Recruiting has size polynomial in the size of G. Thus, if we could determine the answer to the Efficient Recruiting instance in polynomial time, we could also solve the instance of Vertex Cover in polynomial time.