

Lab04-Matroid

CS214-Algorithm and Complexity, Xiaofeng Gao, Spring 2020.

* If there is any problem, please contact TA Yiming Liu.

* Name: Haotian Xue Student ID: 518021910506 Email: xavihart@sjtu.edu.cn

1. Give a directed graph $G = (V, E)$ whose edges have integer weights. Let $w(e)$ be the weight of edge $e \in E$. We are also given a constraint $f(u) \geq 0$ on the out-degree of each node $u \in V$. Our goal is to find a subset of edges with maximal weight, whose out-degree at any node is no greater than the constraint.
 - (a) Please define independent sets and prove that they form a matroid.
 - (b) Write an optimal greedy algorithm based on Greedy-MAX in the form of *pseudo code*.
 - (c) Analyze the time complexity of your algorithm.

Solution of Problem 1

(a) **Definition of independent sets:** Define the edge set $A \subset E$ and the vertex set of A is V_A , also $V_A \subset V$, if for any $u \in V_A$ we have the out degree of u in edge set A is no greater than $f(u)$, we can say A is an independent set. We define the set of independent sets C . We want to prove the $\mathcal{M} = (E, C)$ is a matroid. We define the vertex set of an edge set X is V_X , and the out degree of vertex u in V_X is $out(u, X)$.

Proof. It is easy to prove the hereditary of $\mathcal{M} = (E, C)$, since for any $A \subset B, B \in C$, for any vertex $u \in V_B, u \in V_A$, we can get: $f(u) \geq out(u, B) \geq out(u, A)$. So A is also an independent set, $A \in C$.

To prove the exchange property, we will use an obvious lemma: for edge set X , we have:

$$\sum_{u \in V_X} out(u, X) = |X|$$

Then, for $A, B \in C, |A| < |B|$, we have: $\sum_{u \in V_A} out(u, A) < \sum_{u \in V_B} out(u, B)$, there must exist one vertex u_0 , such that $out(u_0, A) < out(u_0, B) \leq f(u_0)$, which means that the number of edges start from u_0 in A is less than that in B , so we can find an edge $x \in B - A$, define $A' = A \cup \{x\}$, $out(u_0, A') = out(u_0, A) + 1 \leq f(u_0)$. So $A' \in C$.

Considering the two properties above, we have $\mathcal{M} = (E, C)$ is a matroid. □

(b) The pseudo code of MAX-Greedy Algorithm.

Input: Directed graph $G = (V, E)$, constraint list
 $F = \{f(u_1), f(u_2), \dots, f(u_{|V|})\}$, weight list
 $W = \{w(e_1), w(e_2), \dots, w(e_{|E|})\}$

Output: Edge set E_{max} which has the maximum weight.

```

1  $E_{max} = \{\}$ ,  $OutDegreeMap = \{v_1 : 0, v_2 : 0, \dots, v_{|V|} : 0\}$ 
2 Sort the edges by weight in an decreasing order.
3 for  $i = 1$  to  $|E|$  do
4    $u = \text{start vertex of } e_i$ 
5   if  $OutDegree(u) + 1 \leq f(u)$  then
6      $E_{max} = E_{max} \cup \{e_i\}$ 
7      $OutDegree(u) = OutDegree(u) + 1$ 
8 return  $E_{max}$ 

```

(c) The sort part in the algorithm has a time complexity of $O(|E| \log |E|)$. The loop part is $O(|E|)$ since in each loop, we use a map to query and change value of out degrees. So the final time complexity is $O(|E| \log |E|)$.

2. Let X, Y, Z be three sets. We say two triples (x_1, y_1, z_1) and (x_2, y_2, z_2) in $X \times Y \times Z$ are *disjoint* if $x_1 \neq x_2$, $y_1 \neq y_2$, and $z_1 \neq z_2$. Consider the following problem:

Definition 1 (MAX-3DM). *Given three disjoint sets X, Y, Z and a nonnegative weight function $c(\cdot)$ on all triples in $X \times Y \times Z$, **Maximum 3-Dimensional Matching** (MAX-3DM) is to find a collection \mathcal{F} of disjoint triples with maximum total weight.*

- (a) Let $D = X \times Y \times Z$. Define independent sets for MAX-3DM.
- (b) Write a greedy algorithm based on Greedy-MAX in the form of *pseudo code*.
- (c) Give a counterexample to show that your Greedy-MAX algorithm in Q. 2b is not optimal.
- (d) Show that: $\max_{F \subseteq D} \frac{v(F)}{u(F)} \leq 3$. (Hint: you may need Theorem 1 for this subquestion.)

Theorem 1. *Suppose an independent system (E, \mathcal{I}) is the intersection of k matroids (E, \mathcal{I}_i) , $1 \leq i \leq k$; that is, $\mathcal{I} = \bigcap_{i=1}^k \mathcal{I}_i$. Then $\max_{F \subseteq E} \frac{v(F)}{u(F)} \leq k$, where $v(F)$ is the maximum size of independent subset in F and $u(F)$ is the minimum size of maximal independent subset in F .*

Solution of Problem 2

(a)

We consider the set as a 3-D matches between X, Y, Z . The set A is an independent set if and only if every two matches in A are disjoint with each other.

(b)

(x_1, y_1, z_1)	100
(x_1, y_1, z_2)	10
(x_1, y_2, z_1)	10
(x_1, y_2, z_2)	60
(x_2, y_1, z_1)	55
(x_2, y_1, z_2)	40
(x_2, y_2, z_1)	30
(x_2, y_2, z_2)	10

Table 1: Counterexample

Input: Three sets: $X = \{x_1, \dots, x_{n_X}\}, Y = \{y_1, y_2, \dots, y_{n_Y}\}, Z = \{z_1, z_2, \dots, z_{n_Z}\}$ and the weight matrix $W[n_X][n_Y][n_Z]$
Output: The indepent set with the max weight: C
1 $C = \{\}$
2 Construct the list of triple $Triples = \{(x_i, y_j, z_k)\}$ for $i = 1, 2, \dots, n_X,$ $j = 1, 2, \dots, n_Y, k = 1, 2, \dots, n_Z.$
3 Sort $Triples$ by weights of triples in an decreasing order.
4 for $i = 1$ to $n_X n_Y n_Z$ do
5 if $A[i]$ is disjoint with any triples in C then
6 $C = C \cup \{Triple[i]\}$
7 return C

(c)

Counterexample : $X = \{x_1, x_2\}, Y = \{y_1, y_2\}, Z = \{z_1, z_2\}$, the weights of the triples is shown in Table 1.

Using the Greedy-MAX algorithm, we first choose (x_1, y_1, z_1) which has the maximal weight, then we can only choose (x_2, y_2, z_2) . So the independent set accquired is $\{(x_1, y_1, z_1), (x_2, y_2, z_1)\}$ which has a weight of 110, however the independent set $\{(x_1, y_2, z_2), (x_2, y_1, z_1)\}$ has a weight of 115. So the Greedy-MAX cannot get an optimal answer in this example.

(d)

We can use **Theorem 1** to prove the inequality.

Firstly, we define the current independent system (E, \mathcal{I}) which is defined in (a). Each $A \subset \mathcal{I}$ is a disjoint matching on X, Y, Z . Then we define another three independent systems: (E, \mathcal{I}_X) , (E, \mathcal{I}_Y) and (E, \mathcal{I}_Z) , and their definition are as follows:

- (E, \mathcal{I}_X) : $A \subset \mathcal{I}_X$ if and only if : for any $(x_i, y_i, z_i), (x_j, y_j, z_j) \in \mathcal{I}_X, x_i \neq x_j (i \neq j)$.
- (E, \mathcal{I}_Y) : $A \subset \mathcal{I}_Y$ if and only if : for any $(x_i, y_i, z_i), (x_j, y_j, z_j) \in \mathcal{I}_X, y_i \neq y_j (i \neq j)$.
- (E, \mathcal{I}_Z) : $A \subset \mathcal{I}_Z$ if and only if : for any $(x_i, y_i, z_i), (x_j, y_j, z_j) \in \mathcal{I}_X, z_i \neq z_j (i \neq j)$.

3. **Crowdsourcing** is the process of obtaining needed services, ideas, or content by soliciting contributions from a large group of people, especially an online community. Suppose you want to form a team to complete a crowdsourcing task, and there are n individuals to choose from. Each person p_i can contribute v_i ($v_i > 0$) to the team, but he/she can only work with up to c_i other people. Now it is up to you to choose a certain group of people and maximize their total contributions ($\sum_i v_i$).

- (a) Given $\text{OPT}(i, b, c) =$ maximum contributions when choosing from $\{p_1, p_2, \dots, p_i\}$ with b persons from $\{p_{i+1}, p_{i+2}, \dots, p_n\}$ already on board and at most c seats left before any of the existing team members gets uncomfortable. Describe the optimal substructure as we did in class and write a recurrence for $\text{OPT}(i, b, c)$.
- (b) Design an algorithm to form your team using dynamic programming, in the form of *pseudo code*.
- (c) Analyze the time and space complexities of your design.

Remark: You need to include your .pdf and .tex files in your uploaded .rar or .zip file.