Lab04-Matroid

CS214-Algorithm and Complexity, Xiaofeng Gao, Spring 2021.

∗ If there is any problem, please contact TA Haolin Zhou.

∗ Name: Zilong Li   Student ID: 518070910095   Email: logcreative-lzl@sjtu.edu.cn

1. *Property of Matroid.*

   (a) Consider an arbitrary undirected graph $G = (V, E)$. Let us define $M_G = (S, C)$ where $S = E$ and $C = \{I \subseteq E \mid (V, E \backslash I) \text{ is connected}\}$. Prove that M_G is a **matroid**.

   Proof. Hereditary If $A \subset B$, $B \in C$, then because $(V, E\backslash B)$ is connected, cut less edges will also lead to a connected graph, $(V, E\backslash A)$ is connected.

   Exchange Property If $A, B \in C$, $|A| < |B|$, it is to be proved that $\exists e \in B\backslash A, A \cup \{e\} \in C$.

   Proposition 1 (Least Edges)**.** *To connect n vertices, there should be at least $n-1$ edges. And at the least scenario, edges form a tree without a loop.*

   So, $|E\backslash A| > |E\backslash B| \geq |V| - 1$, i.e., $|E\backslash A| \geq |V|$. This means that $E\backslash A$ has at least one loop. Because every edges exceed the original minimum spanning tree could add one more loop in the graph, so $E\backslash A$ has more loops than $E\backslash B$. There must be an edge e on one of the loops that $E\backslash B$ doesn't contain.

   $$e \in E\backslash A \text{ and } e \notin E\backslash B \Rightarrow e \notin A \text{ and } e \in B \Rightarrow e \in B\backslash A$$

   Because e is on the loop of $E\backslash A$, remove the edge won't affect the connectivity on all the vertices. So $A \cup \{e\} \in C$.

   □

   (b) Given a set A containing n real numbers, and you are allowed to choose k numbers from A. The bigger the sum of the chosen numbers is, the better. What is your algorithm to choose? Prove its correctness using **matroid**.

   Remark: Denote $\mathbf{C}$ be the collection of all subsets of A that contains no more than k elements. Try to prove $(A, \mathbf{C})$ is a matroid.

   Solution. Prove to the Remark.

   Hereditary If $B \subset D, D \in \mathbf{C}$, then $|B| \leq |D| \leq k$, $B \in \mathbf{C}$.

   Exchange Property If $B, D \in \mathbf{C}$, $|B| < |D| \leq k$, insert $x \in D\backslash B$ to B denoted as B', $|B'| = |B| + 1 \leq k$, $B' \in \mathbf{C}$.

   So $(A, \mathbf{C})$ is a matroid. Greedy-MAX algorithm is used on the cost function of the value of the element a_i. The corollary about the weighted matroid confirms the correctness of this algorithm.

   Algorithm 1: Greedy-MAX on Number Choosing for Maximum Sum

   Input: set A with n real numbers
   Output: set B with size k chosen in A to be the maximum sum

   ```
   Sort A in decreasing order a_1 ≥ a_2 ≥ ··· ≥ a_n;
   B ← ∅;
   for i ← 1 to n do
       if |B ∪ {a_i}| ≤ k then
           B ← B ∪ {a_i};
   return B;
   ```

   □

2. *Unit-time Task Scheduling Problem.* Consider the instance of the **Unit-time Task Scheduling Problem** given in class.

(a) Each penalty ω_i is replaced by $80 - \omega_i$. The modified instance is given in Tab. 1. Give the final schedule and the optimal penalty of the new instance using Greedy-MAX.

Table 1: Task

a_i	1	2	3	4	5	6	7
d_i	4	2	4	3	1	4	6
ω_i	10	20	30	40	50	60	70

Solution. Sort the weight in an decreasing order first, then apply the algorithm.

Check the number of tasks $N_t(A)$ whose deadline is earlier or equal to t in the task set A, shown in Table 2.

Table 2: Checking Table

A	$N_0(A)$	$N_1(A)$	$N_2(A)$	$N_3(A)$	$N_4(A)$	$N_5(A)$
$\{a_1\}$	0	0				
$\{a_1, a_2\}$	0	0	1			
$\{a_1, a_2, a_3\}$	0	0	1	1		
$\{a_1, a_2, a_3, a_4\}$	0	0	1	2	4	
$\{a_1, a_2, a_3, a_4, a_5\}$	0	1	2	3	5	
$\{a_1, a_2, a_3, a_4, a_6\}$	0	0	1	2	5	
$\{a_1, a_2, a_3, a_4, a_7\}$	0	0	1	2	4	4

Choose the set A satisfying $N_t(A) \le t$ for all $0 \le t \le |A|$. The chosen A is $\{a_1, a_2, a_3, a_4, a_7\}$. Convert A into an canonical form, which is the final schedule:

a_2	a_4	a_1	a_3	a_7	a_5	a_6

And the penalty comes to

$$50 + 60 = 110$$

□

(b) Show how to determine in time $O(|A|)$ whether or not a given set A of tasks is independent. (**Hint**: You can use the lemma of equivalence given in class)

Solution. By the lemma of equivalence, to determine whether or not a given set A of tasks is independent, just to calculate whether:

$$\text{For } t = 0, 1, \cdots, n, N_t(A) \le t$$

Count the type of deadline time to traverse A once, then validate each $N_t(A)$ by the type could cut the time complexity down to $O(|A|)$. The algorithm is shown in Alg. 2.

Algorithm 2: Determine the Independence

Input: task set A

Output: whether A is independent

$D \leftarrow [0, 0, \cdots, 0]$ with size $\max\{d_i\}_{i=1}^{|A|}$;
foreach d_i *in* A **do**
  $D[d_i] \leftarrow D[d_i] + 1$;
$N \leftarrow 0$;
for $j \leftarrow 1$ *to* $|D|$ **do**
  $N \leftarrow N + D[j]$;
  if $N > j$ **then return** *false*;
return *true*;

□

3. *MAX-3DM.* 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 non-negative 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 counter-example to show that your Greedy-MAX algorithm in Q. 3b is not optimal.

(d) Show that: $\max\limits_{F \subseteq D} \frac{v(F)}{u(F)} \leq 3$. (Hint: you may need Theorem 2 for this subquestion.)

Theorem 2. *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 $\max\limits_{F \subseteq E} \frac{v(F)}{u(F)} \leq k$, 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. (a)

Definition 2 (Function Tripple Set)**.** *Let function tripple set on X: $\mathcal{I}_X$ be the family of subsets A of E such that no two triples in any subset share an element in X. The same definition for $\mathcal{I}_Y$ and $\mathcal{I}_Z$.*

Remark of Definition 2: It is called *function tripple set* because the following funciton could be contructed:

$$x \in X' \mapsto (x, y, z) \in A$$

where $A \in \mathcal{I}_X$ and X' is the domain of this function (a bijection), which is a subset of X: $X' \subset X$.

Then $\mathcal{I}_X$, $\mathcal{I}_Y$, and $\mathcal{I}_Z$ are independent sets for MAX-3DM. Prove the hereditary. Suppose $B \subset A, A \in \mathcal{I}_X$, then B won't have two tripples share the same element in X. Otherwise, one of such two tripples won't appear in A. So, $B \in \mathcal{I}_X$. The same for $\mathcal{I}_Y$ and $\mathcal{I}_Z$.

$\mathcal{I}_X$, $\mathcal{I}_Y$, and $\mathcal{I}_Z$ are in fact the matroids for MAX-3DM. The extra proof is the exchange property. Suppose $A, B \in \mathcal{I}_X, |A| < |B|$. According to the remark, the domain of such a function is smaller for A: $X'_A \subset X'_B$. Thus, there must exist $x_0 \in X'_B \backslash X'_A$. And the corresponding tripple (x_0, y_0, z_0) in B will not share the element in X with the tripples in A, because $x_0 \notin X'_A$. Thus, $A \cup \{(x_0, y_0, z_0)\} \in \mathcal{I}_X$. The same for $\mathcal{I}_Y$ and $\mathcal{I}_Z$.

(b) The Greedy-MAX algorithm reuqires us to find the intersection of $\mathcal{I}_X$, $\mathcal{I}_Y$, and $\mathcal{I}_Z$.

Algorithm 3: Greedy-MAX on MAX-3DM

Input: $D = X \times Y \times Z$, non-negative weight funciton $c(\cdot)$ on all tripples in D
Output: collection $\mathcal{F}$ of disjoint tripples in D with maximum total weight

```
Sort all tripples (x_i, y_i, z_i) in D such that their weight is ordered decreasingly;
F ← ∅;
foreach (x_i, y_i, z_i) in D do
    if F ∪ {(x_i, y_i, z_i)} ∈ I_X ∩ I_Y ∩ I_Z then
        F ← F ∪ {(x_i, y_i, z_i)};
return F;
```

(c) The counter example could be contructed as follows for $X = Y = Z = \{0, 1, 2\}$:

$$c(0,0,0) = 10 \qquad c(1,1,1) = 7 \qquad c(2,2,0) = 7$$
$$c(0,0,2) = 7 \qquad c(2,2,2) = 1$$

The unidentified tripple weighs 0.

The Greedy-MAX will get $\mathcal{F} = \{(0,0,0), (1,1,1), (2,2,2)\}$, which has the weight summation of $10 + 7 + 1 = 18$. However, the optimal solution $\mathcal{F}^* = \{(0,0,2), (1,1,1), (2,2,0)\}$, which has the weight summation of $7 + 7 + 7 = 21$. So the greedy solution is not the optimal soltion.

(d) It is proved that $\mathcal{I}_X$**,** $\mathcal{I}_Y$**, and** $\mathcal{I}_Z$ **are matroids for MAX-3DM** in Q. 3a. And the algorithm gets the intersection of the three matroids, which satisfying the condition. So the original problem could be easily followed by Theorem 2.

Simplified Proof of Theorem 2. (is referenced[*]) Consider two maximal independent subsets I (minimum size) and J (maximum size) of F with respect to $(E, \mathcal{I})$. For each $1 < i < k$, let I_i be a maximal independent subset of $I \cup J$ with respect to $(E, \mathcal{I}_i)$ that contains I. For any $e \in J \backslash I$, it occurs in at most $k - 1$ different subsets $I_i \backslash I$ otherwise contradicting the maximality of I.

$$\sum_{i=1}^{k} |I_i| - k|I| = \sum_{i=1}^{k} |I_i \backslash I| \leq (k-1)|J \backslash I| \leq (k-1)|J|$$

Now, for each $1 \leq i \leq k$, let J_i be a maximal independent subset of $I \cup J$ with respect to $(E, \mathcal{I}_i)$ that contains J. Since, for each $1 \leq i \leq k$, $(E, \mathcal{I}_i)$ is a matroid, we must have $|I_i| = |J_i|$. In addition, for every $1 \leq i \leq k$, $|J| \leq |J_i|$. Therefore,

$$k|J| \leq \sum_{i=1}^{k} |J_i| = \sum_{i=1}^{k} |I_i| \leq k|I| + (k-1)|J|$$

which follows $|J| \leq k|I|$.

□

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

[*]Chapter 2.1-2.2 in "Design and Analysis of Approximation Algorithms" by D.-Z. Du, K.-I. Ko, and X. D. Hu, Springer, 2012.