Lab04-Matroid

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

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

∗ Name:Zirui Liu Student ID:519021910343 Email: L.prime@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. To prove that M_G is a matroid, it has to satisfy those three conditions. For the first,$S = E$, and obviously S is not empty. For the second, since $(V, E\backslash I)$ is connected, so (V, E) must be connected too. Since adding more edges to a graph that is already connected would not make it disconnected, for any $B \in C$, and $A \subseteq B$, we have $A \in C$. For the third, for any $A \in C, B \in C, |A| < |B|$, similar to the way we prove the second condition, since C can be defined as " the power set featuring all the edges as elements, for $X \in B - A$, we can always have $A \cup \{X\} \in C$. So in conclusion, M_G is a **matroid**.

   □

   (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 **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. The solution is as following: We sorted the n numbers, then we choose from them from the biggest to the smallest until we have chosen k numbers. These k numbers are those we wish to find. Next I will prove the correctness of the greedy algorithm.

   We suppose that $M = (U, F)$ is a weighted matrioid and that U is sorted into monotonically decreasing order. We also assume that $\omega(x)$ represents the weight of element x. Then we let x to be the biggest element of U such that $\{x\} \in F$. If any such x exists, then there exists an optimal subset A of U containing x, thus the optimizism of the greedy algorithm can be proved. The proof is as following.

   If no such x exists, then the only independent subset is the empty set like $(F = \{\varnothing\})$ then the question is meaningless. We assume that F contains another non-empty optimal subset B, then there exists two possibilities: $x \in B$ or $x \notin B$. In the first case, we can simply make $A = B$ to prove the correctness. In the second case, we assume that $x \notin B$. Then we assume there exists $y \in B$ so that $\omega(y) > \omega(x)$. Since $y \in B$ and $B \in F$ so we can get $\{y\} \in F$. If $\omega(y) > \omega(x)$, then y would be the first element of U, making a contradiction with our original assumption.

   So, $\forall y \in B, \omega(x) \geq \omega(y)$. We can now construct a set $A \in F$ such that $x \in A, |A| = |B|$, and $\omega(A) \geq \omega(B)$. We then use the second characteristic of **Matroid**, we can always find an element of B to add to A without destructing the independence. We can repeat this process until we get $|A| = |B|$. Then by construction we have $A = B - \{y\} \cup \{x\}$ for $y \in B$. Then since $\omega(A) \geq \omega(B)$, we have $\omega(A) = \omega(B) - \omega(y) + \omega(x) \geq \omega(B)$. Since B is optimal, then A containing x is also optimal. So the correctness of our algorithm is proved.

   □

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. **??**. 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. The optimal penalty is 30 for the following schedule: $a_5 \Rightarrow a_6 \Rightarrow a_4 \Rightarrow a_3 \Rightarrow a_7 \Rightarrow a_2 \Rightarrow a_1$ □

   (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. To determine in time $O(|A|)$ whether or not a given set A of tasks is independent, we can use the lemma of equivalence. That is to say, we need to prove that the tasks in A are scheduled in order of monotonicallly increasing deadlines. So what we have to do is to check if all the tasks in A whether their deadline time is in monotonicallly increasing order or not. On the other hand, we can solver the problem by using the lemma: set A is independent if and only if for any $t = 0, 1, 2, ...n$, $N_t(A) \le t$. The pseudo code is as follows:

   Algorithm 1:

   Input: An array $a[1, \cdots, n]$

   Output: a bool value, indicating A is independent or not.

   ```
   for i ← 1 to n do
       a[i] ← a[i] + 1;
   Nt ←0
   for i ← 1 to n do
       N ← Nt + a[i];
       if Nt > i 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. **??** is not optimal.

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

Solution. (1):
We define that: $\forall\,(x_i, y_i, z_i)\,and\,(x_j, y_j, z_j) \in \mathcal{F}, (x_i, y_i, z_i)\,and\,(x_j, y_j, z_j)$ are disjoint. Then we have this $\mathcal{F}$ to be independent.
(2):
We define (x_i, y_i, z_i) as a_i

Algorithm 2:

Input: An array $a[1, \cdots, n]$
Output: An array A, indicating the result we want.

```
sort F by c in non-increasing order;
A ← ∅;
for i ← 1 to n do
    if A ∪ {a_i} ∈ F then
        A ← A ∪ {d_i};
return A
```

(3):
We give that: $a_1 = (x_1, y_1, z_1) = (a, b, c)$, having the weight of 10. $a_2 = (x_2, y_2, z_2) = (a, e, f)$, having the weight of 9. $a_3 = (x_3, y_3, z_3) = (g, b, k)$, having the weight of 8. for any other a_i, their total weight conbined is less than 5. By the greedy algorithm we should choose a_1 since it has the largest weight, but the best solution is choosing a_1, (whether we consider other elements does not matter), but the best choice is choosing a_2 and a_3.
(4):
According to $Theorem 1$, we can transform the question to proving that $(E, \mathcal{I}_i)$ for $i = 1, 2, 3$ are **matroids**. We start by constructing three sets for $\mathcal{I}_i$.
$\mathcal{I}_1$: $\forall\,(x_i, y_i, z_i)\,and\,(x_j, y_j, z_j) \in E,\ x_i \neq x_j$.
$\mathcal{I}_2$: $\forall\,(x_i, y_i, z_i)\,and\,(x_j, y_j, z_j) \in E,\ y_i \neq y_j$.
$\mathcal{I}_3$: $\forall\,(x_i, y_i, z_i)\,and\,(x_j, y_j, z_j) \in E,\ z_i \neq z_j$.
And $\mathbf{I} = \bigcap_{i=1}^{k} \mathcal{I}_i$.
Heredity:
It is easy to understand that the heredity can always be satisfied.
Exchange Property:
For $A, B \subseteq \mathcal{I}_1$, $|A| < |B|$, there exist $a_i = (x_i, y_i, z_i) \in B - A$, considering that $\mathcal{I}_1$: $\forall\,(x_i, y_i, z_i)\,and\,(x_j, y_j, z_j) \in E,\ x_i \neq x_j$, so $\forall\,(x_j, y_j, z_j) \in A,\ x_i \neq x_j$, so $A \cup \{a_i\} \in \mathcal{I}_1$. Thus the exchange property for $(E, \mathcal{I}_1)$ is proved. The same as $(E, \mathcal{I}_2)$ and $(E, \mathcal{I}_3)$ by using y_i and z_i. So in conclusion, $(E, \mathcal{I}_i)$ for $i = 1, 2, 3$ are **matroids**, so $\max\limits_{F \subseteq D} \frac{v(F)}{u(F)} \leq 3$.

□

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

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