Lab04-Matroid

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

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

∗ Name:_XudongLu_ Student ID:_519021910372_ Email: _luxudong2001@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. i. **hereditary**

   If $B \in C$ and $A \subseteq B$, then we confidently get that $E\backslash B$ is connected. A has fewer edges than B, same vertex number with B and it is still connected. The complement graph of A has more edges than B.

   We can regard the complement graph of A as adding edges to that of B. So it can be obviously get that the complement graph of A is connected.

   An example here is shown as below. The red edges are the edges of the graphs. The dotted lines can be regarded as the elements c that $c \in B$ but $c \notin A$. We can also see the removing of the lines as adding new edges to the complement graph of A. Thus the new complement grap is still connected. We prove the hereditary property.

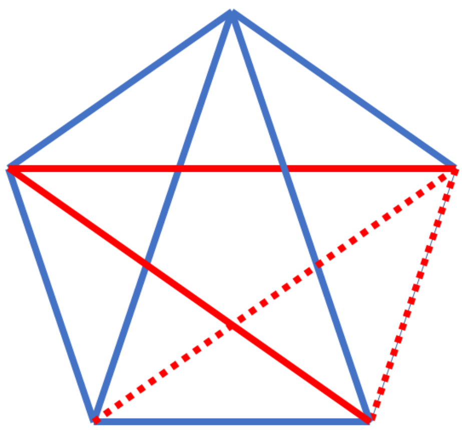

Figure 1: set A

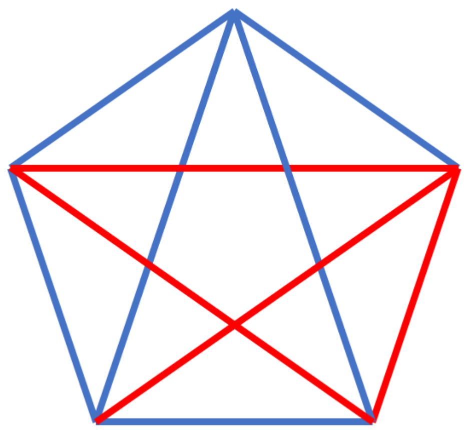

Figure 2: set B

   ii. **exchange property**

   In order to prove the assumption, we assume two sets A and B and $|A| < |B|$. A and B both have the same number of vertex set V, so adding the edge $e \in B\backslash A$ will link two vertexs togther. This means that the complement graph of A will have more edges than B.

   It is obvious that the worst case of a connected graph is that it is a tree. It will have at least $V - 1$ edges. In the complement graphs of A and B, we can just move away the basic $V - 1$ edges which make them connected. The graph B has more edges than A. In this case, we can simply get one, say x, in B but not in A and put it into graph A. Then $x \in B\backslash A, A \cup x \in C$.

   Here is also an example. The red edges are the edges of the graphs. By adding the purple edge into set A, the complement graph of A is still connected.

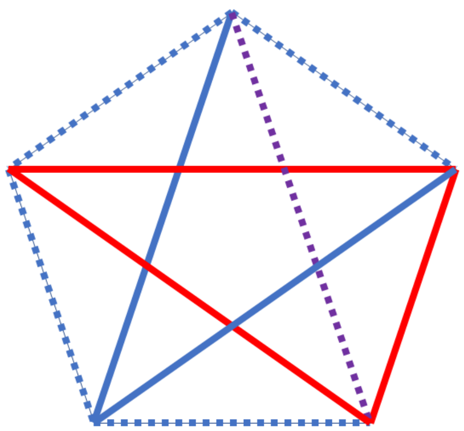

Figure 3: set A

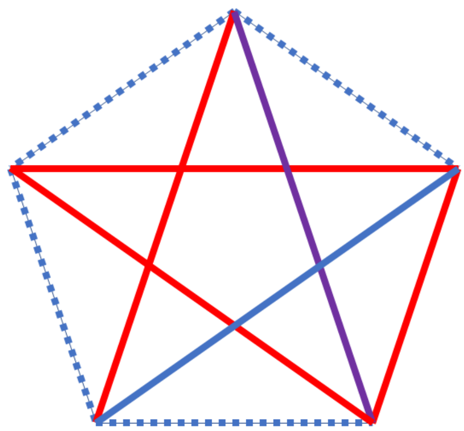

Figure 4: set B

□

(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.

Algorithm 1: Greedy-MAX

Sort the numbers in A into a nonincreasing order

$S \leftarrow \Phi$

for i=1:k **do**

  $S \leftarrow S \cup A[i]$

Solution. We can prove the correctness of this algorithm by proving the assumption that when $\mathbf{C}$ is the collection of all subsets of A that contains no more than k elements. $(A, \mathbf{C})$ is a matroid.

Lemma1 $(A, \mathbf{C})$ is a matroid.

i. **hereditary**
We define $A \subseteq B, B \in C$, where $\mathbf{C}$ is a set having no more than k elements. It is obvious that A is a set containing no more than k elements.

ii. **exchange property**
Also, if $|A| < |B| \leq k$, we can find an element $s \in B, s \notin A$. We add this number into A and $|A| \leq k$, $A \in \mathbf{C}$. Thus $(A, \mathbf{C})$ is a matroid.

After proving this lemma, we can use *Greedy-MAX* solution to solve this problem correctly.

□

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. Tab. 2 is the **table** of sorting the elements by the weight into a decreasing sequence.

Table 2: sorting

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

Tab. 3 is the **table** of choosing elements one by one and see wether it belongs to the **matroid**.

Table 3: selecting

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

Accroding to the table, we find that the optimal penalty is 30, and one possible sequence is $a_5 \to a_4 \to a_3 \to a_6 \to a_7 \to a_1 \to a_2$.

$\square$

(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. We can solve the problem by traversing the set A. We first traverse the set and get the maximum **deadline** of the set. We define is as d_{max}. Then we create an array d of d_{max} elements and initialize it all to 0.

Then we traverse the set again. In this case, for each $a \in A$, we add 1 to $d[a.deadline]$.

Then for $a \in A$, we compute the prefix sum of the sequence d.

In the end, we check the array d. If there exists certain $d[a.deadline] > a.deadline$, we think the set is not independent. Otherwise we think the set is independent.

The algorithm is showed as bellow.

We need to prove that d_{max} is equal to or smaller than $|A|$. Because if $d_{max} \geq |A|$, we can just throw the element with the specific deadline away. Because all other jobs can be done in $[0, d_{max} - 1]$. The certain job can do the job in the interval $[d_{max} - 1, d_{max}]$.

$\square$

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:

Algorithm 2: Independent checking

$d_{max} \leftarrow 0$;
$prev \leftarrow 0$;
for $a \in A$ **do**
  $d_{max} = max(d_{max}, a.deadline)$;
$d \leftarrow int[d_{max}]$;
for $a \in A$ **do**
  $d[a.deadline] + +$;
for $i = 1 : d_{max}$ **do**
  if $d[i] \neq 0$ **then**
    $d[i] = d[i] + prev$;
    $prev = d[i]$;
for $a \in A$ **do**
  if $d[a.deadline] > a.deadline$ **then**
    return $false$
return $true$;

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.

A subset of $I \subseteq D$ is independent if every two triples in I are disjoint.

The **proof** is easy as A is a subset of a set B of which all the triples are distinct, then the elements of A are distinct.

(b) Write a greedy algorithm based on Greedy-MAX in the form of *pseudo code*.

Algorithm 3: Item-Greedy

Sort the weight of D in A into a nonincreasing order $c(d_1) \geq c(d_2) \geq \cdots \geq c(d_n)$
$\mathcal{S} \leftarrow \Phi$
for i=1:n **do**
  if $\mathcal{S} \cup d(i) \in \mathcal{C}$ **then**
    $\mathcal{S} \leftarrow \mathcal{S} \cup d(i)$

(c) Give a counter-example to show that your Greedy-MAX algorithm in Q. 3b is not optimal.

We define $X = \{1, 2\}, Y = \{3, 4\}, Z = \{5, 6\}$.

We define the function $c(\cdot)$ as $c(1, 3, 5) = 9, c(1, 4, 5) = 7, c(2, 3, 5) = 3, c(2, 4, 5) = 6, c(1, 3, 6) = 5, c(1, 4, 6) = 4, c(2, 3, 6) = 3, c(2, 4, 6) = 0$.

By the greedy algorithm, we get the answer that we first choose $c(1, 3, 5) = 9$ then choose $c(2, 4, 6) = 0$, the answer we get is 9.

However, the optimal answer is $c(2, 4, 5) + c(1, 3, 6) = 11$. So the greedy algorithm is not the optimal solution.

(d) Show that: $\max_{F \subseteq D} \frac{v(F)}{u(F)} \leq 3$. (Hint: you may need Theorem 1 for this subquestion.)

Solution. In order to prove this, we need to compose three matroids. We define three subsets of $\mathcal{D}$. In the firt subset, the first element of the triples are all distinct. In the second subset, the second elements of the triples are all distinct. In the third subset, the third element of the triples are all distinct. We call these three subsets $1-\mathcal{D}$, $2-\mathcal{D}$ and $3-\mathcal{D}$.

We prove that the three subsets are all **matroids**.

Lemma2 $1-\mathcal{D}$, $2-\mathcal{D}$ and $3-\mathcal{D}$ are all matroids.

i. **hereditary**
We consider $B \in 1-\mathcal{D}$, the elements' first elements are all different. Then we consider $A \subseteq B$, it is obvious that the first elements of the elements in A are all different. So $1-\mathcal{D}$ has the property of hereditary.

ii. **exchange property**
We still consider $1-\mathcal{D}$, $A \in 1-\mathcal{D}$ and $B \in 1-\mathcal{D}$, $|A| < |B|$. Then B have more first elements than A and all the elements are distinct, where we can find one $c \in B \backslash A$. The first element of c is still different than any other ones of A. Thus by adding this element to A, $A \cup c \in 1-\mathcal{D}$.

So we have now proved that $1-\mathcal{D}$ satisfies the property of a matroid. The proof of $2-\mathcal{D}$ and $3-\mathcal{D}$ is similar.

It is obvious that $\mathcal{D}$ is the intersection of $1-\mathcal{D}$, $2-\mathcal{D}$ and $3-\mathcal{D}$. Thus by **Theorem** 1, we can get that $\max\limits_{F \subseteq D} \frac{v(F)}{u(F)} \leq 3$.

$\square$

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.