Lab03-Greedy Strategy

CS214-Algorithm and Complexity, Xiaofeng Gao, Spring 2019.

$*$ If there is any problem, please contact TA Mingran Peng.

$*$ Name:Xiangyu Liu Student ID:517021910869 Email: liuxiangyu999@sjtu.edu.cn

1. Suppose there is a street with length n, described by an array $A[1...n]$ where $A[i] = 1$ means that there is a house at position i and $A[i] = 0$ means position i is vacant.

   According to some law, every house must be protected by fire hydrant. If a fire hydrant is placed at position i, then all houses at position $i-1, i, i+1$ will be considered protected. Note that hydrants can be placed at the same place with a house.

   Using what you learnt in class, please design an algorithm that computes the minimum number of hydrants needed to protect all houses. You need to write pseudo code, analyze the time complexity, and prove its correctness.

Proof.
- **Greedy Strategy:** We search from the first position. Here are two cases.
  - **Case 1.** If there is no house at position i, we continue on position $i+1$, without placing a hydrant at position i.
  - **Case 2. If there is a house at position i still without protection of any hydrants, we place a hydrant at position $i+1$.** This strategy ensures that positions i, $i+1$, and $i+2$ are all under protection. As a result, we just need to continue our algorithm at position $i+3$.
  - **Complements For Case 2.** In Case 2, if we find such a house at position i and i is already the last position, we just place a hydrant at position i not $i+1$.
- **Pseudo Code:**

```
Algorithm 1: HYDRANT PLACEMENT
input  : An array $A[1...n]$ of $n$ elements.
output: The minimum number of hydrants needed to protect all houses.
1 $count \leftarrow 1$;
2 $i \leftarrow 1$;
3 while $i \le n$ do
4     if $A[i] == 0$ then
5         $i \leftarrow i + 1$;
6     else
7         $count \leftarrow count + 1$;
8         $i \leftarrow i + 3$;
9 return $count$;
```

- **Time Complexity:**

  In our analysis of time complexity, we determine the time complexity by counting the number of iterations of the **while-loop**.

  In each iteration, the indicator i will be **either added by** 1 **or added by** 3. As a result, the maximum number of iterations will be n and the minimum number of iterations will be $\lceil \frac{n}{3} \rceil$.

  As a result, **the time complexity is** $\Theta(n)$.

- **Correctness:** We prove by contradiction.
  Assume greedy is not optimal. Let denote

  $$k = \max\{m | \text{For any optimal solution } A, \text{ its first } m \\ \text{positions of hydrants are the same with the greedy solution}\}$$

  At the same time, we denote the $A^\star$ such optimal solution satisfying $m = k$.
  We assume k^{th} hydrant is located at position i for greedy solution and $A^\star$ and $(k+1)^{th}$ hydrant for greedy solution is located at position j. Now we can get the following lemma.

  Lemma 1. *There is only one hydrant at position$[i+2, \cdots j]$ for the optimal solution $A^\star$ and* ***the position of this $(k+1)^{th}$ hydrant must be located at position $j-2$ or $j-1$ or j.***

  Proof. Since the next hydrant of the k^{th} hydrant for greedy solution is located at position j, it follows that there is only one house at *position*$[i+2, \cdots j-1]$. **If more than one house, the position of $(k+1)^{th}$ hydrant for greedy solution will be less than or equal to $j-1$.**
  Furthermore, according to **Case 2**, this house is located at position $j-1$, which is because greedy solution has placed a hydrant at position j.
  As a result, for the optimal solution $A^\star$, it must place the $(k+1)^{th}$ hydrant at position $j-2$ or $j-1$ or j to protect the house at position $j-1$.
  Since there is only one hydrant at *position*$[i+2, \cdots j]$ for the greedy solution, more than one hydrant at *position*$[i+2, \cdots j]$ in $A^\star$ will contradicts the optimal attribute of $A^\star$. □

  According to the **Lemma 1.** above, we just need to discuss the hydrant position for $A^\star$ in the following two case.

  - **Case 1.** The $(k+1)^{th}$ hydrant is located at $j-2$ or $j-1$.
    Then we can construct another optimal solution **by shifting the $(k+1)^{th}$ hydrant to position j.**
    This mean we find another optimal solution, **having more than k the same positions of hydrants with the greedy solution**, which contradicts the definition of $A^\star$.
    Complements: In order to prove we can carry out such an operation indeed, we complement a proof that there is no hydrant at position j. If there is really a hydrant at position j, it means that the $(k+1)^{th}$ hydrant at position $j-2$ or $j-1$ can be removed. Hence it contradicts that $A^\star$ is optimal.
  - **Case 2.** The $(k+1)^{th}$ hydrant is located at j.
    It means that the greedy solution and $A^\star$ has the same position of $(k+1)^{th}$ hydrants, which contradicts the maximality of k.

  Finally, we have to make complements for the special case that the $(k+1)^{th}$ hydrant for the greedy solution is located at the final position n and the final house is also located at position n not $n-1$. In such case, the $(k+1)^{th}$ hydrant for $A^\star$ can be only located at position $n-1$ or n. If located at position $n-1$, we can imitate proof of **Case 1**, constructing another optimal solution, which leads to the contradiction. If located at position n, the greedy solution will be $A^\star$.

  □

2. (a) 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: This is a very easy problem. 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.

Proof.

- **Algorithm:** Our algorithm is to **find the largest k elements**.
  Firstly, we find the largest number in $A[1 \cdots n]$, say it a_0. Then we repeat the same process for $A[1 \cdots n] \backslash \{a_0\}$ to find the largest number. We repeat the process until we find k elements.
  Here is the pseudo code:

```
Algorithm 2: LARGEST $k$ ELEMENTS
input  : $k$, An array $A[1 \dots n]$ of $n$ elements.
output: An array $B[1 \cdots k]$ containing the $k$ elements we want.
for $i = 1$ to $k$ do
    for $j = 1$ to $n - i$ do
        if $A[j] > A[j+1]$ then
            swap $A[j]$ and $A[j+1]$;
    $B[i] = A[n - i + 1]$;
return $B[1 \cdots k]$;
```

- **Correctness:**
  First Step. We denote $\mathbf{C}$ be the collection of all subsets of A that contains no more than k elements.
  Here we prove $(A, \mathbf{C})$ is a matroid.
  - Hereditary:
    Given any $F \in \mathbf{C}$ and $P \subset F$.
    Since $F \in \mathbf{C}$, we get $|F| \leq k$. According to $P \subset F$, we get $|P| \leq |F| \leq k$. This means $P \in \mathbf{C}$.
  - Exchange Property:
    Consider $B, D \in \mathbf{C}$ with $|B| > |D|$.
    For $b_0 \in B$
    D:
    $$|D \cup \{b_0\}| = |D| + 1 \leq |B| \leq k$$
    This means $D \cup \{b_0\} \in \mathbf{C}$.

  Second Step. We define the associated strictly positive function $c : S \to \mathbb{R}^+$ to each element $x \in S$ as follows:
  $$c(x) = x, \quad \forall x \in S$$
  This definition means:
  $$\text{maximize} \sum_{i=1}^{k} x_i \Leftrightarrow \text{maximize} \sum_{i=1}^{k} c(x_i)$$
  $$\text{subject to} \quad x_i \in S \quad \forall i = 1, 2, \cdots k$$
  Since our algorithm select the k largest element in S and have defined the weight $c(x) = x, \quad \forall x \in S$, our algorithms just **performs the same process as the Greedy-MAX algorithm** and get the same result consequently.
  According to **Corollary of Greedy Theorem for Independent System**[*], our algorithm performs the best result.

[*]Xiaofeng Gao. (2019). Slide07-Matroid [Powerpoint slides].

$\square$

(b) Consider that $B_1, B_2 ... B_n$ are n disjoint sets, and let d_i be integers with $0 \leq d_i \leq |B_i|$. Define $\mathbf{C}$ is a collection of set $X \subseteq \cup_{i=1}^n B_i$, where X has such property:

$$\forall i \in \{1, 2, 3...n\}, |X \cap B_i| \leq d_i$$

Prove that $(\cup_{i=1}^n B_i, \mathbf{C})$ is a matroid.

Remark: You may easily find that the matroid in (a) is a special case of matroid in (b).

Proof.
- Hereditary: Given $X \in \mathbf{C}$ and $Y \subset X$.
  We get:
  $$\forall i \in \{1, 2, 3...n\}, \quad (Y \cap B_i) \subset (X \cap B_i)$$
  This means:
  $$\forall i \in \{1, 2, 3...n\}, \quad |Y \cap B_i| \leq |X \cap B_i| \leq d_i$$
  Therefore, $\mathbf{C}$ is hereditary.
- Exchange Property:
  Consider $X, Y \in \mathbf{C}$ with $|X| > |Y|$.
  Since $B_1, B_2 ... B_n$ are n disjoint sets, we get:
  $$|X| = |X \cap (\cup_{i=1}^n B_i)| = |\cup_{i=1}^n (X \cap B_i)| = \sum_{i=1}^n |X \cap B_i|$$
  Similarly, we get:
  $$|Y| = |Y \cap (\cup_{i=1}^n B_i)| = |\cup_{i=1}^n (Y \cap B_i)| = \sum_{i=1}^n |Y \cap B_i|$$
  Since $|X| > |Y|$, we claim that **there must be some i, such that** $|Y \cap B_i| < |X \cap B_i|$. We will prove it by contradiction.
  Proof. Assume that $|Y \cap B_i| \geq |X \cap B_i|, \forall i \in \{1, 2 \cdots n\}$, we will conclude that:
  $$|Y| = \sum_{i=1}^n |Y \cap B_i| \geq \sum_{i=1}^n |X \cap B_i| = |X|$$
  This contradicts $|X| > |Y|$. $\square$
  Now we can find an element $z \in (X \cap B_i) \backslash (Y \cap B_i)$, namely an element $z \in X \backslash Y$ since $B_1, B_2 ... B_n$ are disjoint sets.
  Here we will prove $Y \cup \{z\} \in \mathbf{C}$.
  Case 1.
  $$\forall j \neq i, |(Y \cup \{z\}) \cap B_j| = |(Y \cap B_j) \cup (\{z\} \cap B_j)| = |(Y \cap B_j) \cup \emptyset| = |(Y \cap B_j)| \leq d_j$$
  Case 2.
  $$|(Y \cup \{z\}) \cap B_i| = |(Y \cap B_i) \cup (\{z\} \cap B_i)| = |(Y \cap B_i) \cup \{z\}| = |(Y \cap B_i)| + 1 \leq d_i$$
  In conclusion, we find an element $z \in X \backslash Y$, such that $Y \cup \{z\} \in \mathbf{C}$.

$\square$

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