# Lab03-Greedy Strategy

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

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

∗ Name: KylinChen  Student ID: 517030910155  Email: k1017856853@icloud.com

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 postion $i-1, i, i+1$ 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.

**Solution.**

**Algorithm Description:**

1) First, we put a **flag** at street head, Then go to the next 2) step.
2) If the **flag** points a house, go to 3) step. If the **flag** doesn't point a house, go to 4) step.
3) Put a hydrant in the next adjacent position, which can cover this house. Then move the **flag** to next three position, which can't be covered by previous hydrants. Finally, we can go to 5) step.
4) Now that we didn't find the house, we move the **flag** to next one position. Finally, we can go to 5) step.
5) If the position is out of range, end it to get all the hydrants. Otherwise, we can go to 2) step.

This algorithm can described by pseudo code as follows. Algorithm. 1 is to find the greedy set of all hydrants position, while Algorithm. 2 is to find the greedy optimal number of hydrants.

**Pseudo Code:**

```
Algorithm 1: Greedy-Selection
    Input: An array $A[1 \cdots n]$
    Output: Set $B$ contains all the optimal locations of hydrants
$B \leftarrow \emptyset$;
$i \leftarrow 1$;
while $i \leq n$ do
    if $A[i] == 1$ then
        $B \leftarrow B \cup \{i+1\}$;
        $i \leftarrow i+3$;
    else
        $i \leftarrow i+1$;
```

**Algorithm 2:** Greedy-Count

```
Input: An array $A[1 \cdots n]$
Output: Number $C$, which describe the number of hydrants

$C \leftarrow 0$;
$i \leftarrow 1$;
while $i \leq n$ do
    if $A[i] == 1$ then
        $C \leftarrow C + 1$;
        $i \leftarrow i + 3$;
    else
        $i \leftarrow i + 1$;
```

**Complexity Analysis:**

Let input array $= A$ and $n = A's$ length.

- If the input array $A$ is 0-array, the **while-loop** will take maximum $O(n)$ time.
- If the input array $A$ satisfies restrictions:$A[1 + 3 \times k] = 1$ ($k \in \mathbb{N}$), The **while-loop** will take minimun $O(\frac{n}{3}) = O(n)$ time.

Thus, Greedy-Selection algorithm takes $O(n)$ time.

**Correctness Proof:**

**Claim.** The Greedy-Selection algorithm is optimal.

(If there exsits more than one optimal solution, greedy solution has same number of hydrants with other optimal solution)

**Proof.** Define $G$ is Greed-Selection, $O$ is Optimized-Selection. Assume before k-th selection, $G$ and $O$ have same positions chosen. In the k-th selection, $G$ select position $e$ while $O$ select position $f$.

$Invariant1$ : There exists an optimal selection $O$ that makes $G$ and $O$ have same position choice through the k-th selection.

$Invariant2$ : There exists an optimal selection $O$ that makes $G$ and $O$ have different position choice at the k-th selection, but for k+1 selction, it make no sense for k-th selection's homogeny.

- **case 1:** If $e = f$, it means $O$ and $G$ select same position, which satisfies $Invariant1$.

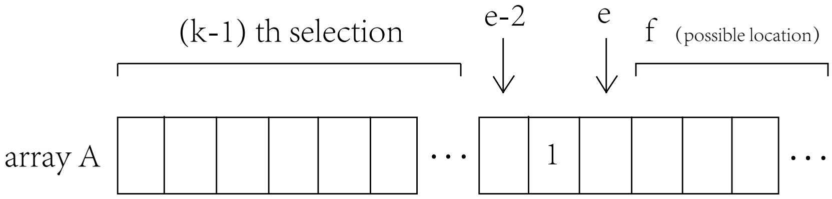

Figure 1: An Illustration of Case 2

- **case 2:** If $e < f$, it means $O$ choose a position $f$ after $G$ chosen $e$. With the greedy algorithm description, we can claim that position $e-1$ must be 1(have a house), and $e-2$ can't exist a hydrant, otherwise position $e$ can't be chosen by $G$. It means that, position $e-1$ have not be covered by any hydrant in (k-1)-th selection. For optimal selection choosing position $f$ after $e$, $e-1$ can't be covered, which have a contradiction with optimal definition, therefore this situation can't exist.
- **case 3:** If $e > f$, greedy algorithm choose a later position than optimal algorithm, we can devide this case into two small one.

  **key observation:** According to the contradiction in **case 2**, we can claim in arbitrary intreval [f,m], $G$ choosing positions less than $O$ choosing positions, otherwise, $O$ will have not covered 1(house).

  · **key observation proof:** Assume in the interval [f,m], $G$ choose d positions, and $O$ have chosen no more than d-1 positions. The presupposition $e > f$ means $f \leq e-1$, for greedy algorithm, a chosen position $i$ must have a house at position $i-1$. Therefore, the interval [f,m] have more than d houses, and for these d houses just before chosen position, which mutually at a distance of 3, according to **Pigeonhole Principle**, the $O$ algorithm can't cover all house, it's a contradiction.

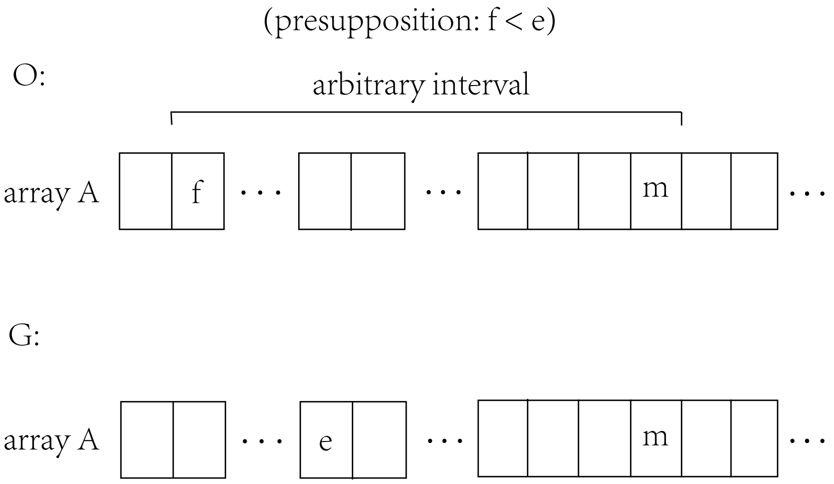

Figure 2: An Illustration of Case 3 key observation

  * **case 3a:** If there exsit r-th selection ($r > k$) before array A ends (street ending), which satisfies that in r-th selection, $G$ and $O$ choose the same postion, which satisfies $Invariant1$. And according to key observation, $G$ is no worse than $O$ between [k,r].
  * **case 3b:** If there not exist any same selection position after postion $f$, we can set A[n] as the end element both algorithm have chosen. In that case, according to key observation, $G$ is no worse than $O$ after k-th selection until ending.

As is mentioned above, Greedy-Selection algorithm is no worse than optimal algorithm, and their selection equals when there is noly one optimal solution.

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 Description:** We first sort these $k$ numbers with nonincreasing ordering. Then we get the maximum number one by one until we get total $k$ numbers.

**Pseudo Code:**

```
Algorithm 3: Greedy-Selection
  Input: An array $A[1 \cdots n]$
  Output: Set $B$ contains k numbers which have optimal sum

Sort n numbers in nonincreasing ordering $A[1] \geq A[2] \geq \cdots \geq A[n]$;
$B \leftarrow \emptyset$;
for $i \leftarrow 1$ to $k$ do
    $B \leftarrow B \cup A[i]$;
return $B$;
```

**Correctness Proof:** Denote $\mathbf{C}$ be the collection of all subsets of $A$ that contains no more than $k$ elements. We first prove system $(A, \mathbf{C})$ satisfies **Hereditary** and **Exchange Property**.

- **Hereditary:** If $D \subset B$ and $B \in \mathbf{C}$, meaning $D$ is a subset of $A$ which contains no more than $k$ elements. Therefore, $D$ must be an element in subset collection $\mathbf{C}$.
- **Exchange Property:** Once we choice two element $D, F \in \mathbf{C}$ and $|D| < |F|$, it means $|D| < k$, $\exists x \in F \backslash D$, $D \cup \{x\}$ can be a subset of $A$ which is no more than k. Therefore, $D \cup \{x\} \in \mathbf{C}$.

Now that the problem can be converted into matroid, as its subproblems follow this conversion, this problem can use greedy algorithm (for example, Alg. 3) to get one of optimal solution, so our algorithm is correct.

(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 $X$, 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.**

System $(\cup_{i=1}^{n} B_i, \mathbf{C})$ is a matroid **iff** System $(\cup_{i=1}^{n} B_i, \mathbf{C})$ satisfies **Hereditary** and **Exchange Property**.

- **Hereditary:** If $P \in \mathbf{C}$, which means $\forall i \in \{1, 2, 3 ... n\}, |P \cap B_i| \leq d_i$. Then for any subsets P' of P, it's obvious that $|P' \cap B_i| \leq |P \cap B_i| \leq d_i$, P' must be an element of collection $\mathbf{C}$. It says that $P' \subset P, P \in \mathbf{C} \Rightarrow P' \in \mathbf{C}$
- **Exchange Property:** For any elements $P, Q \in \mathbf{C}$ and $|P| < |Q|$. With regard to set $P$, there must exists a group of inequation:

$$|P \cap B_1| \leq d_1$$

$$|P \cap B_2| \leq d_2$$

$$\dots$$

$$|P \cap B_k| < d_k$$

$$\dots$$

$$|P \cap B_n| \leq d_n$$

$\exists k \in \{1, 2, 3...n\}$, which makes $|P \cap B_k| < d_k$ (equal mark can't be established in this inequation). Otherwise, any extension will violate constraint equation

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

which will lead Q not exists.

In the other hand, since $|P| < |Q|$ and $Q \in \mathbf{C}$, $Q$ must have an extension element $\epsilon$ in $B_k$ (k-th inequation can't get equal mark established in P's constraint inequations, otherwise which can lead Q not exists as before). Then we can choose this extension $\epsilon$ as $x \in Q \backslash P$, $P \cup \{x\}$ will satisfies the constraint inequations, which means $P \cup \{x\} \in C$.

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

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