For many optimization problems...

Outline

2. Greedy choice property: optimal solution can be obtained by making the 'greedy' choice at each step.
3. Greedy algorithm: optimal solution can be obtained by making the 'greedy' choice at each step.
Algorithm

Activity Selection Problem

Greedy Algorithm for Activity Selection Problem

1. \( \text{return } \{ f \} \)
2. \( \text{do } \{ f \} \cap \{ t \} \neq \emptyset \) \( \text{while } \{ f \} \neq \emptyset \)
3. \( \text{for } i \in \{ f \} \)
4. \( \text{if } i \cap \{ t \} \neq \emptyset \)
5. \( \{ f \} \leftarrow \{ f \} \setminus \{ i \} \)
6. \( \text{end for} \)
7. \( \{ t \} \leftarrow \{ t \} \setminus \{ i \} \)
8. \( \text{return } \{ f \} \)

CREDIBLE-ACTIVITY-SELECTION(\( n \), \( f \))

Assume activities ordered by increasing finishing time

Greedy algorithm for Activity Selection Problem

1. \( \text{initialize } \{ f \} \) as \( \emptyset \)
2. \( \text{for each activity } a \in \{ t \} \)
3. \( \text{if } a \) is compatible with \( \{ f \} \)
4. \( \{ f \} \leftarrow \{ f \} \cup \{ a \} \)
5. \( \text{else} \)
6. \( \text{if } \{ f \} \neq \emptyset \) \( \text{and } \{ f \} \) is compatible with \( a \)
7. \( \{ f \} \leftarrow \{ f \} \cup \{ a \} \)
8. \( \text{end if} \)
9. \( \text{end for} \)
10. \( \text{return } \{ f \} \)

Time: \( O(n \log n) \)

Space: \( O(n) \)
Let $\mathcal{A}$ be an optimal solution for the activity-selection problem.

**Lemma:** If $A$ is an activity, we have either $A \in \mathcal{A}$ or $A \notin \mathcal{A}$.

Proof: By contradiction. Assume there exists an activity $A$ such that $A \in \mathcal{A}$ and $A \notin \mathcal{A}$. Since $A$ is an activity, there must be a time slot $t$ during which $A$ is available.

Let $B$ be an activity that overlaps with $A$ at time $t$. Then, $B \in \mathcal{A}$, contradicting the assumption that $A \notin \mathcal{A}$. Therefore, $A \in \mathcal{A}$ if and only if $A \notin \mathcal{A}$.

**Theorem:** The activity-selection problem has an optimal solution that consists of a single activity.

Proof: Let $\mathcal{A}$ be an optimal solution. Then, $|\mathcal{A}| = 1$.

**Proof:** If $|\mathcal{A}| > 1$, then there exists an activity $A$ such that $A \notin \mathcal{A}$. Since $A$ is an activity, there must be a time slot $t$ during which $A$ is available.

Let $B$ be an activity that overlaps with $A$ at time $t$. Then, $B \in \mathcal{A}$, contradicting the assumption that $A \notin \mathcal{A}$. Therefore, $|\mathcal{A}| = 1$.

**Corollary:** The activity-selection problem has an optimal solution that consists of a single activity.

Proof: Let $\mathcal{A}$ be an optimal solution. Then, $|\mathcal{A}| = 1$.

**Proof:** If $|\mathcal{A}| > 1$, then there exists an activity $A$ such that $A \notin \mathcal{A}$. Since $A$ is an activity, there must be a time slot $t$ during which $A$ is available.

Let $B$ be an activity that overlaps with $A$ at time $t$. Then, $B \in \mathcal{A}$, contradicting the assumption that $A \notin \mathcal{A}$. Therefore, $|\mathcal{A}| = 1$.

**Proposition:** Suppose $G$ is an optimal solution to the activity-selection problem.

Proof: Let $\mathcal{A}$ be an optimal solution. Then, $|\mathcal{A}| = 1$.

**Proof:** If $|\mathcal{A}| > 1$, then there exists an activity $A$ such that $A \notin \mathcal{A}$. Since $A$ is an activity, there must be a time slot $t$ during which $A$ is available.

Let $B$ be an activity that overlaps with $A$ at time $t$. Then, $B \in \mathcal{A}$, contradicting the assumption that $A \notin \mathcal{A}$. Therefore, $|\mathcal{A}| = 1$.

**Theorem:** The activity-selection problem has an optimal solution that consists of a single activity.

Proof: Let $\mathcal{A}$ be an optimal solution. Then, $|\mathcal{A}| = 1$.

**Proof:** If $|\mathcal{A}| > 1$, then there exists an activity $A$ such that $A \notin \mathcal{A}$. Since $A$ is an activity, there must be a time slot $t$ during which $A$ is available.

Let $B$ be an activity that overlaps with $A$ at time $t$. Then, $B \in \mathcal{A}$, contradicting the assumption that $A \notin \mathcal{A}$. Therefore, $|\mathcal{A}| = 1$.
And greedy algorithms...
Huffman codes

Why Greedy algorithm does not work for 0.1 frequency

Example:
- Number of necessary bits: [7] 7
- There are 7 characters to encode (7-bit code word)

Given the number of characters to encode:
- A, B, C, D, E, F, G

Given the length: Fixed length encoding: alphabet

Data encoding

Code and binary code for each character:

Example: Encoding is represented by a binary string

Characters represented by a binary character code (code):

(a: 1010, b: 1011, c: 1000, d: 0110, e: 0101, f: 1110, g: 0100)

Encryption of appearance of characters:

{ a: 5, b: 3, c: 1, d: 2, e: 1, f: 0, g: 1 }

A than the 10000 characters
Prefix tree is an optimal character code for this tree.

Prefix tree is an optimal character code for this tree.

Binary tree

Leaf to leaf representation of a prefix code

Prefix code (same prefix-free code)

Example of leaf to leaf representation of a prefix code

Example of leaf to leaf representation of a prefix code

Example of leaf to leaf representation of a prefix code
The code

Importantly, the sum of all members (cost of tool) is the cost of

The human code chooses the member that incurs the least cost

Moreover

CPT (i.e. a single member is sum of frequencies of all objects belonging

where $\alpha$ is the greedy choice.

Choice of merging together those two clusters of lowest frequency

Lemma 1.2: Consider the cost to build up an optimal tree by

CPT: choice property: Least

Since $D \in \text{optimal tree}$ and $T_j \in \text{optimal tree}$

Therefore $D \in \text{optimal tree}$ and $T_j \in \text{optimal tree}$

Similarly

$0 \in (\text{optimal tree})$ and $0 \notin (\text{optimal tree})$

Exchange $c$ in $T_i$'s, yield

Exchange $b$ in $T_i$'s, yield

$\exists$ $f \in [0, \infty]$, such that $f \notin T_i$'s

Let $c \in T_i$ be such that $c \notin T_i$'s

We show that the cost can be expressed in terms of $B$.

\[ \{x \} \cap \{y \} = \emptyset \subset C \]