Greedy Algorithms and Exchange Proofs

- Scheduling
- Huffman Trees
- Gale-Shapley

Prove solution is optimal through exchange arguments.

Available classes

N classes


Solution space: \( 2^N \times \mathcal{P}(S) \)

Feasible solution: \( (A, C) \) classes overlap

Goal: Select most feasible classes that are disjoint

Dynamic Programming: Best Schedule

Best Schedule from \( A \) not conflicting with \( C \).

If \( |A| = 0 \), then \( \emptyset \).

Otherwise, \( A = (X, A') \).

return \( \max \{ \begin{cases} \text{if } X \text{ does not conflict with class } C, & \text{if Best}(A|C) \text{ conflicts with } C \text{ add } \text{if} \}\text{not}\text{conflict}\text{in}\text{class}\text{for}\text{if} \text{Best}(A',C) \text{ conflicts with } C \text{ add } \text{not}\text{conflict}\text{in}\text{class}\text{for} \text{if} \text{not}\text{conflict}\text{in}\text{class}\text{for} \end{cases} \} \)

Try 1: Sort the classes by length, pick shortest first

Try 2: Sorts fixed:

Try 3: Stopping time first

How & why prove this gives an optimal solution?

Correctness:

Sched \((1, n)\) : 

\(n\) first classes in best schedule 

from \(a_1 \ldots a_n\) 

Subtract after +
By induction else: \( \text{Schedule} \left( t+1, t, n \right) \)

This: \( \text{Schedule} \left( 0, 0 \right) \) produces an optimal schedule.

\[
\text{Suppose } C_1, \ldots, C_k \text{ is an optimal schedule}
\]

and \( S_1, \ldots, S_m \) is the output of our algo. \( \text{Schedule} \left( 0, 1 \right) \).

Then \( k \geq m \).

Find difference \( S_j \) and \( C_i \)

Case I: \( S_j = C_i \). \( S_j \) is an optional

Case II: \( S_j = C_{i+1}, \ldots, C_k \)

\[
\text{Stop} \left( C_{i+1}, \ldots, C_k \right) \geq \text{Stop} \left( S_1, \ldots, S_m \right)
\]

\[
\text{Start} \left( C_{i+1}, \ldots, C_k \right) \geq \text{Start} \left( S_1, \ldots, S_m \right)
\]

\[
\cdots \geq \text{Start} \left( C_{i+1} \right) \geq \text{Start} \left( S_1 \right)
\]

Case III:

\( S_j = C_{i+1} \), \( C_{i+2}, \ldots, C_k \) are both optimal.

\( j \) is last index

When \( S_j \) and \( C_i \) agree.

\( C' = S_j, S_{j+1}, \ldots, S_m \) is also optimal & feasible.

\[
\text{Optimal: } \text{Start} \left( C' \right) \leq \text{Stop} \left( C' \right) \leq \text{Start} \left( C_{i+2} \right)
\]

\( S_j \) not in \( C' \) in any order.

By the above critical index \( j \),

we avoid optimal schedules that agree with \( S_j \) on more and more classes, unlike Case I or II.

Huffman Codes:

\[
\begin{array}{c|c|c|c}
\hline
\text{Prefix-free code:} & \text{Savethese codes:} & \text{Savethese codes:} \\
\hline
A : 00 & A : 01 & A : 0 \\
B : 01 & B : 10 & B : 1 \\
C : 101 & C : 10 & C : 1 \\
D : 1001 & D : 110 & D : 11 \\
E : 1011 & E : 110 & E : 11 \\
\hline
\end{array}
\]
Given prior knowledge of frequency of source symbols, derive expected length for code $T$:

$$
\mathbb{E} = \sum_{s \in \mathcal{S}} p(s) \cdot \text{depth}(s, T)
$$

For example:

$$
2 \cdot (0.2 \cdot 1 + 0.2 \cdot 3) + 3 \cdot (0.4 \cdot 1 + 0.1 \cdot 0.2)
$$

**Optimal Code Tree?**

A possible choice in each subproblem.

**BST:**

$$
\left[ a_0, \ldots, a_j, a_{j+1}, \ldots, a_n \right]
$$

- left subproblem
- right subproblem

Here, each subproblem has... $2^n$.

**Greedy Alg:**

- merge the two least frequent symbols into one node.

![Diagram of a tree with nodes labeled with frequencies]

**Correctness:**

By induction.

1. tree of 2 leaves
2. tree of 2 subproblems

So, gains a subtree.

**Optimality:**

- depth($Y$) < depth($X$)

**Lemmas:**

1. Swapping $X$ for $Y$ in $T$

$$
\text{Cost}(T') - \text{Cost}(T) = (p(X) - p(Y))(\text{depth}(X, T) - \text{depth}(X, T))
$$

**Proof:**

$$
\text{Cost}(T') = \ldots + p(Y) \cdot \text{depth}(Y, T)
$$
\[ \text{Cost}(T) = \ldots p(x) \cdot \text{depth}(X, Y) \]

Redraw: Swapping X for Y, if \( \text{depth}(x) > \text{depth}(y, t) \) and \( p(x) > p(y) \), then this reduces cost.

Lemma 2: There is an optimal tree that has the two least frequent symbols as siblings.

a. \( X, W \) (X, W are least frequent)

\[ \exists \text{optimal } Z, \text{ containing } X, W \]

Proof: Start with optimal tree \( T, \text{depth}(X, Y) < \text{depth}(Z, W) \).

Cases:

1. \( X \) done
2. \( Z \)

By Lemma 1, \( Z \text{ is also optimal.} \)

\[ \text{Cost}(T) - \text{Cost}(T') = \text{Cost}(Z) - \text{Cost}(Z') \]

(ordered by ar abs)

- Let \( T \) be the output of greedy algo.
- Let \( Z \) be an optimal tree with \( p_{n-1}, p_n \) as leaves.

This reduces cost, \( p(C) \leq p(Z) \). Apply Lemma 2 to reduce cost.

Thin: This is optimal.
We know $Z$ is optimal but not its substructure.

Let $T'$ be the tree after merging $(p_{n-1}, p_n)$ to one node.

Same for $Z$.

We know $T'$ is optimal for the smaller problem (by IH), but we do not know it for $Z$.

However, $\text{cost}(Z - Z') = p_{n-1} + p_n = \text{cost}(T - T')$.

$\text{cost}(T') \leq \text{cost}(Z')$.

$\text{cost}(T) = \text{cost}(T') + (p_{n-1} + p_n) \leq \text{cost}(Z') + (p_{n-1} + p_n)$.