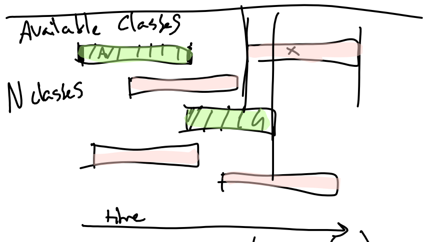
Greedy Algorithms and Exchange Proofs

Tuesday, April 14, 2020 3:22 PM

- Scheduling
- Huffman Codes
- Gale Shapley

Prove solution is optimal through exchange arguments.



Solution Space: $2^N = \mathcal{P}(S)$

Feasible solution: no two classes overlap.

Goal: feasible w/ most # of classes.

Dyn Prog: Best Sched($\overset{i}{\cancel{A}}$, $\underline{\text{Conf}}$) : ← set of conflicts

↖ set of classes to choose from

// # classes in Best Schedule from A, not conflicting w/ Conf.

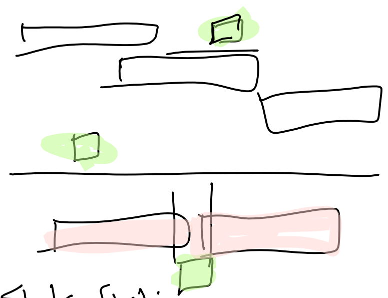
if $|A| = 0$, then 0

otherwise, $A = (\underline{X}, A')$ — one class: X; rest of classes: A'

return max {
// include X
if X doesn't conflict with class in Conf!
1 + Best(A', Conf ∪ {X}) i+1
// don't include X
Best(A', Conf) i+1
}

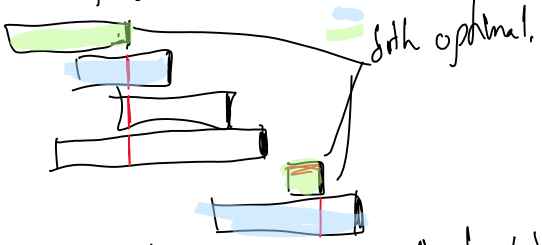
subproblems?
Conf ∈ $\mathcal{P}(S)$

Try 1: Sort the classes by length. Pick shortest first



Try 2: Starts first:

Try 3: Stopping time first



- How do we prove this gives <u>an</u> optimal solution?
- Correctness: $a_0, a_1, \ldots a_n$ (in order by stopping time)

Sched(i, t) :

// # of classes in best schedule from $a_i \ldots a_n$, Starting after t

```
if $i \geq n$ then 0,
otherwise: if $\boxed{start(a_i) \geq t}$:
        1 + Sched(i+1, stop(a_i))
    else:
        Sched(i+1, t)
```

By Induction

Thm: Sched(0, 0) produces an optimal schedule.

Suppose $c_1, \dots c_K$ is an optimal schedule

and $g_1, \dots, g_m$ is the output of our alg. Sched(0,1),

Then $k \geq m$.

Find difference b/w g and c

Case I: $g = c$. g is an optimal

Case II:

$$g_1, \dots g_m, c_{m+1}, \dots c_K$$

$$\begin{array}{ccc} g_1 & \dots & g_m \\ \parallel & & \parallel \\ c_1 & & c_m \end{array}$$

$stop(c_{m+1}) \geq stop(g_m)$

$start(c_{m+1}) \geq stop(g_m)$

.... $sched(i+1, stop(g_m))$ ...

∴ considered c_{m+1} before terminating.

Case III:

$$g_1 \dots g_j \begin{cases} g_{j+1} \quad g_{j+2} \dots g_m & \text{output for alg.} \quad k \geq m. \\ c_{j+1}, c_{j+2}, \dots c_K & \text{optimal} \end{cases}$$

j is last index where g and c agree.

$c' = g_1 \dots g_j, g_{j+1}, c_{j+2}$ is also optimal & feasible.

optimal: total # classes unchanged

$stop(g_{j+1}) \leq Stopping(c_{j+1}) \leq start(c_{j+2})$

So c' not in conflicts.

By induction on critical index j,

we construct optimal schedules that agree with g on more and more classes, until Case I or II. ✓

Huffman Codes:

Σ

Prefix free code:

A	00
B	001
C	1

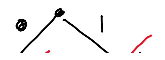
$$\overset{A}{\overbrace{00}}\underset{B}{\underbrace{0\overset{C}{1}}}$$

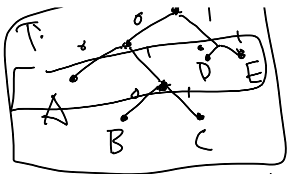
- - - - - 100.1 - -

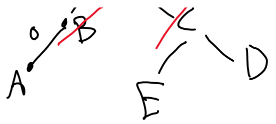
	$p(X)$
A:	0.2
B:	0.4
C:	0.1
D:	0.1
E:	0.2

Source	code
A :	00
B :	010
C :	011
D :	10
E :	11

Code tree:







Given prior knowledge of frequency of source symbols,

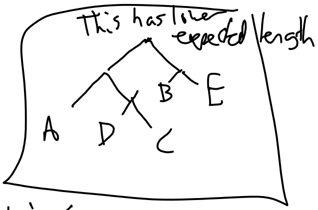
define expected length for code T:

A: 0.2 E: 0.2
B: 0.4
C: 0.1
D: 0.1

$$\sum_{s \in \Sigma} p(s) \cdot depth(s,T)$$

eg: $2 \cdot (.2 + .1 + .2) + 3 \cdot (.4 + .6) =$

This has lower expected length.



Optimal code tree?

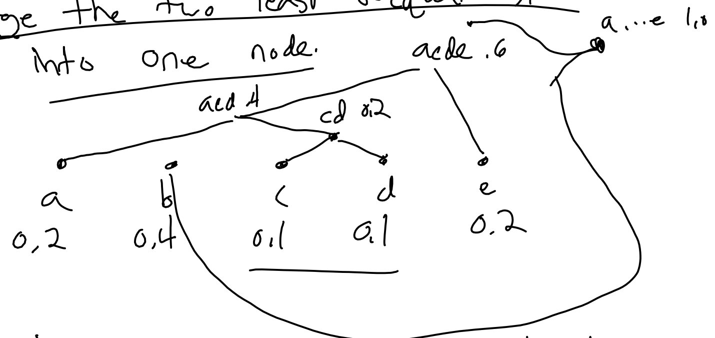
n possible choices in each subproblem.

BST: $\left[\underbrace{a_0 \ldots a_j}_{\text{left subtree}} \middle| \underbrace{a_{j+1} \quad a_n}_{\text{right subtree}}\right]$

Here, each subproblem has... 2^n

Greedy Alg:

⓪ — Merge the two least frequent symbols into one node.



Correctness:
By Induction.

(I) tree w/ 2 leaves

(II) tree w/ 2 subtrees, each has symbols at leaves

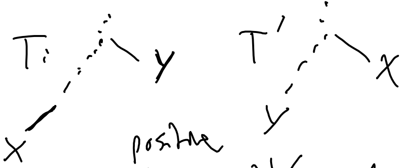
So, gives a code tree

Optimality:

$depth(Y) < depth(X)$

Lemmas:

1: Swapping X for Y in T



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

Prf: $Cost(\underline{T'}) = \ldots\; p(Y) \cdot depth(X,T)$
$+\; p(X) \cdot depth(Y,T)$

$p(X) \cdot depth(Y,T)$ neg.

$$Cost(T) = \dots\ p(X) \cdot depth(X,T) + p(Y) \cdot depth(Y,T)$$

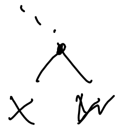
Rephrase: Swapping X for Y, if $depth(X,T) > depth(Y,T)$ and $p(X) > p(Y)$, then this reduces cost.

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

$a_1, \dots X, W$ (X, w are least frequent)

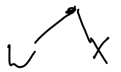
∃ optimal Z, containing

Z:



Proof: Start w/ some optimal tree T. $depth(X,T) < depth(Z)$ in T.

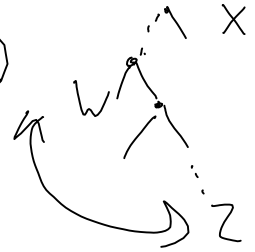
Cases: (I)



done (II)

we know $p(X) \le p(Z)$,

(III)

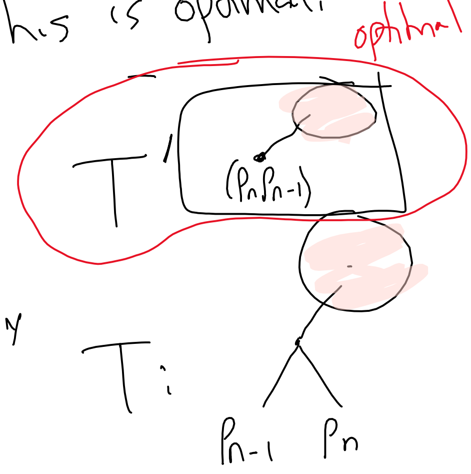


By lemma 1, this reduces cost so T' is also optimal.

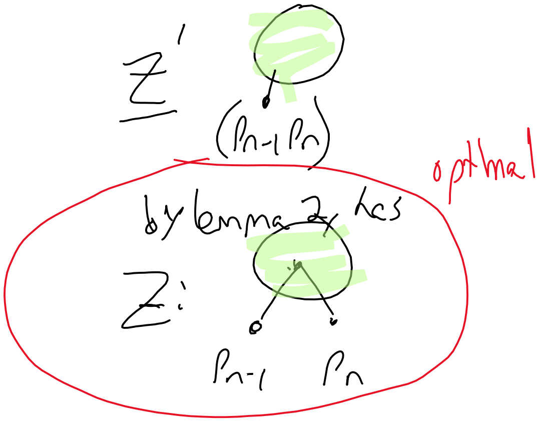
$p(w) \le p(z)$. Apply Lemma 1 to reduce cost.

Thm: This is optimal.

$Cost(T) - Cost(T') = Cost(Z) - Cost(Z')$



(output by our alg)



- Let T be the output of greedy alg.
- Let Z be an optimal tree with P_{n-1}, P_n as leaves (by Lemma 2).
- We know how T was generated, but not that it's optimal.

We know how

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

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

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