CS 410/584, Algorithm Design & Analysis: Lecture 2

Memoizing

Dynamic Programming in a recursive (top-down) framework

- Whenever you compute the value of a subproblem

"memoized f"

`f` becomes `mf` plus `MEMO` table

```
mf(x)
    if MEMO[x] defined then {}
    else MEMO[x] ← f(x)
    return(MEMO[x])
```

Replace all calls to `f` by calls to `mf`

- includes recursive calls to f in the body of f
- but not the call in mf.

Greedy Algorithm

A series of locally optimum choices produces a globally optimum solution

What is needed:

- Greedy choice property — of all the optimal solutions, at least one one contains the initial "greedy" choice.
- Optimal substructure — as with dynamic programming Optimal solutions contain optimal solutions to subproblems.

© 1994, 2003, 2006, 2008, 2009

David Maier

Greedy Example

Activity Selection

Have a set of possible activities to do biggest #

What is the most non-conflicting events I can attend.

Have start and finish times

Order by increasing end time

LIST	A ✓	B ✗	C ✗	D ✓	E ✗	F ✓	G ✗
	2-5	1-6	3-6	6-8	5-9	10-11	10-12

Sched = { A, D, F }

Schedule Algorithm

```
Sched ⇐ ∅
for each ACT in LIST do
   if ACT compatible with Sched
   then Sched ⇐ Sched ∪ {ACT}
```

no time overlap

Complexity: O(sort LIST)

Correct?

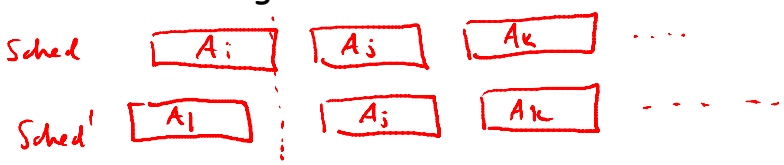
Let LIST = $A_1\ A_2 \cdots A_n$

© 1994, 2003, 2006, 2008, 2009

David Maier

Greedy Choice

- Suppose `Sched` is a maximal legal schedule that doesn't include A_1 (activity with earliest finish)
- Let A_i be the activity in `Sched` with earliest finish
- Then A_1.finish $\geq$ A_i.finish
- So `Sched'` = Sched − $\{A_i\} \cup \{A_1\}$ is also a maximum legal schedule.



Optimal Substructure

- Assume `Sched` is maximal legal schedule for $A_1 \ldots A_n$ that contains A_1
- Let A_i be the first activity in LIST with A_i.start $\geq$ A_1.finish
- Then `Sched` − $\{A_1\}$ is a maximal legal schedule for $A_i \ldots A_n$
  - must be legal
  - if Sched' were a better (larger) schedule for $A_i \ldots A_n$, then Sched' $\cup \{A_1\}$ is a better schedule to $A_1 \ldots A_n$

© 1994, 2003, 2006, 2008, 2009

David Maier

Huffman Codes

← blank

◊	A	E	H	N	R	S	T
000	001	010	011	100	101	110	111

THE◊RAT◊SAT◊AT◊THE◊HEN◊REST◊

28 chars
3 bits
84 bits

Frequencies

◊	A	E	H	N	R	S	T
7	3	4	3	1	2	2	6
00	100	01	1010	101110	101110	101111	11

Contribution

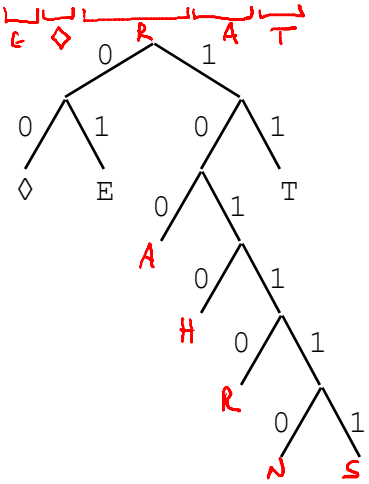
14	9	8	12	6	10	12	12	$\bar{Z} = 83$

Prefix Code

No code word is a prefix of another

111010010010110100 11

T H E ◊ R A T



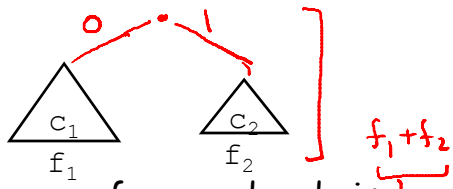
Every interior node has two children

© 1994, 2003, 2006, 2008, 2009

David Maier

Greedy Algorithm for Code Generation

Start with n separate 1-character codes
Merge two codes with lowest frequencies



Frequency of merged code is $f_1 + f_2$

Until there is only one tree,

Huffman Code Example

7	3	4	3	1	2	2	6
◊	A	E	H	N	R	S	T

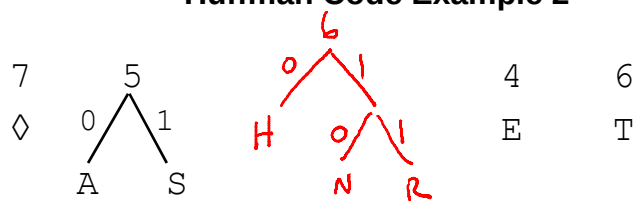
7	3	4	3	3 (0: N, 1: R)		2	6
◊	A	E	H			S	T

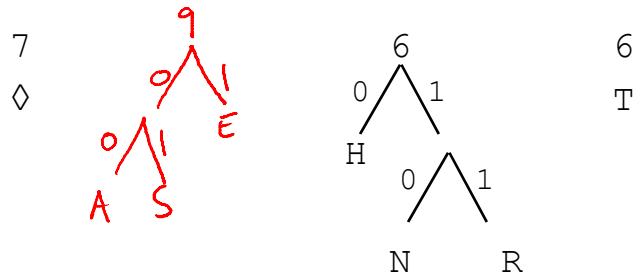
7	5 (0: A, 1: S)	4	3	3 (0: N, 1: R)		6	
◊		E	H			T	

© 1994, 2003, 2006, 2008, 2009

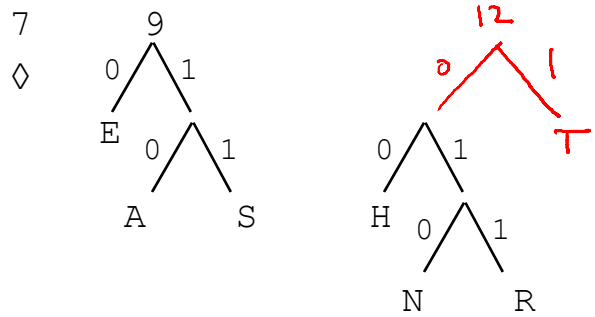
David Maier

Huffman Code Example 2





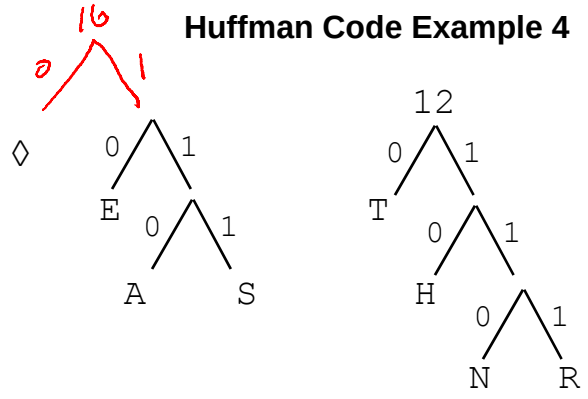
Huffman Code Example 3



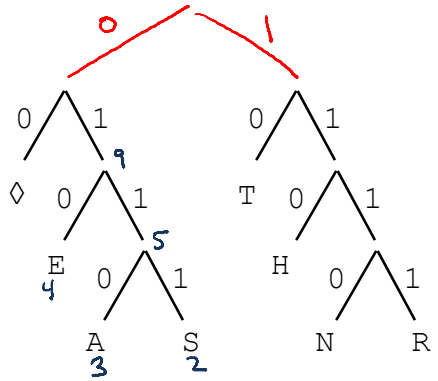
© 1994, 2003, 2006, 2008, 2009

David Maier

Huffman Code Example 4



Huffman Code Example 5



Why does it work?

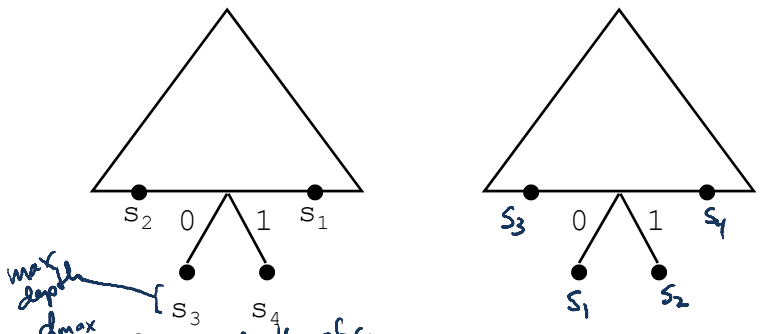
© 1994, 2003, 2006, 2008, 2009

David Maier

Greedy Choice

- Greedy choice

s_1, s_2 lowest frequency chars.



max depth d_{max}

d_1 – depth of s_1
d_2 – depth of s_2
d_{max} - depth s_3, s_4

f_i is frequency of s_i

Greedy Choice 2

Contribution of $s_1 - s_4$ in original tree

$$f_3 \cdot d_{max} + f_4 \cdot d_{max} + f_1 \cdot d_1 + f_2 \cdot d_2$$

$$f_1 \cdot d_{max} + f_2 \cdot d_{max} + f_3 \cdot d_1 + f_4 \cdot d_2$$

s_1–s_3 switch:

old	$(f_3 \cdot d_{max} + f_1 \cdot d_1)$
new	$-(f_1 \cdot d_{max} + f_3 \cdot d_1)$
subtract	$(f_3 - f_1) \cdot d_{max} + (f_1 - f_3) d_1$

$(f_3 - f_1) \cdot d_{max} - (f_3 - f_1) \cdot d_1$

$\underbrace{(f_3 - f_1)}_{\geq 0} \cdot \underbrace{(d_{max} - d_1)}_{>0} \geq 0$

© 1994, 2003, 2006, 2008, 2009

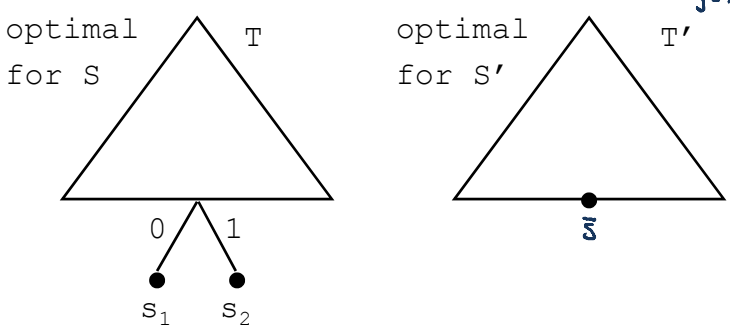
David Maier

Optimal Substructure

$S' = S - \{s_1, s_2\} \cup \{\bar{s}\}$

$\bar{s} = f_1 + f_2$

Let S be the set of symbols



$\text{cost}(T) = \text{cost}(T') + f_1 + f_2$

a lower cost tree for T'' for S' would give a lower-cost tree for T.

© 1994, 2003, 2006, 2008, 2009

David Maier