

CSE 421 Intro to Algorithms Winter 2001

Huffman Codes: An Optimal Data Compression Method

1

Data Compression

- Binary character code ("code")
 - each k-bit source string maps to unique code word (e.g. k=8)
 - "compression" alg: concatenate code words for successive k-bit "characters" of source
- Fixed/variable length codes
 - all code words equal length?
- Prefix codes
 - no code word is prefix of another (simplifies decoding)

CSE 421, W'01, Ruzzo

2

Compression Example

- 100k file, 6 letter alphabet:
 - a 45%
 - b 13%
 - c 12%
 - d 16%
 - e 9%
 - f 5%
- File Size:
 - ASCII, 8 bits/char: 800kbits
 - $2^3 > 6$; 3 bits/char: 300kbits
 - 00,01,10 for a,b,d; 11xx for c,e,f: 2.52 bits/char $74\% \cdot 2 + 26\% \cdot 4$: 252kbits
 - Optimal?
- Why?
 - Storage, transmission vs 1Ghz cpu

CSE 421, W'01, Ruzzo

3

Q 45%
b 13%
c 12%
d 16%
e 9%
f 5%

```

HUFFMAN(C)
1 n ← |C|
2 Q ← C
3 for i ← 1 to n-1
4   do z ← ALLOCATE-NODE()
5     x ← left[z] ← EXTRACT-MIN(Q)
6     y ← right[z] ← EXTRACT-MIN(Q)
7     f[x] ← f[x] + f[y]
8     INSERT(Q, z)
9 return EXTRACT-MIN(Q)
    
```

Figure 17.5 The steps of Huffman's algorithm for the frequencies given in Figure 17.3. Each pair shows the contents of the queue sorted into increasing order by frequency. At each step, the two trees with lowest frequencies are merged. Leaves are shown as rectangles containing a character and its frequency. Internal nodes are shown as circles containing the sum of the frequencies of its children. An edge connecting an internal node with its children is labeled 0 if it is an edge to a left child and 1 if it is an edge to a right child. The number for a letter is the number of labels on the edges connecting the root to the leaf for that letter. (a) The initial heap of $n = 6$ nodes, one for each letter. (b)-(j) Intermediate steps. (j) The final tree.

Huffman's Algorithm

Algorithm:

```

HUFFMAN(C)
n ← |C|
Q ← C
for i ← 1 to n-1
  do z ← ALLOCATE-NODE()
    x ← left[z] ← EXTRACT-MIN(Q)
    y ← right[z] ← EXTRACT-MIN(Q)
    f[z] ← f[x] + f[y]
    INSERT(Q, z)
return EXTRACT-MIN(Q)
    
```

Analysis:

$O(n)$ heap ops: $O(n \log n)$

Goal:

$$\text{Minimize } B(T) = \sum_{c \in C} \text{freq}(c) * \text{depth}(c)$$

Correctness:

???

CSE 421, W'01, Ruzzo

5

Correctness Strategy

- Optimal solution may not be **unique**, so cannot prove that greedy gives the **only** possible answer.
- Instead, show that greedy's solution is **as good as any**.

CSE 421, W'01, Ruzzo

6

T :

Defn: A pair of leaves is an inversion if $\text{depth}(x) \geq \text{depth}(y)$ and $\text{freq}(x) \geq \text{freq}(y)$

Claim: If we flip an inversion, cost never increases.

Why? All other things being equal, better to give more frequent letter the shorter code.

before — after

$$d(x)f(x) + d(y)f(y) - (d(x)f(y) + d(y)f(x))$$

$$= (d(x) - d(y)) (f(x) - f(y)) \geq 0$$

i.e. cost savings is non-negative

Lemma 1: "Greedy Choice Property"

The 2 least frequent letters might as well be siblings (at deepest level)

- Let a be least freq, b 2nd
- Let u be least freq at max depth, v its sibling
- Then (a,u) and (b,v) are inversions. Swap them.

CSE 421, W'01, Ruzzo 8

Lemma 2: "Optimal Substructure"

Let (C, f) be a problem instance: C an n-letter alphabet with letter frequencies $f(c)$ for c in C.

For any x, y in C, let C' be the $(n-1)$ letter alphabet $C - \{x, y\} \cup \{z\}$ and for all c in C' define

$$f'(c) = \begin{cases} f(c), & \text{if } c \neq x, y, z \\ f(x) + f(y), & \text{if } c = z \end{cases}$$

Let T be an optimal tree for (C, f) .

Then

is optimal for (C, f) among all trees having x, y as siblings

CSE 421, W'01, Ruzzo 9

Proof:

$$B(T) = \sum_{c \in C} d_T(c) \cdot f(c)$$

$$B(T) - B(T') = d_T(x)(f(x) + f(y)) - d_T(z) f(z)$$

$$= (d_T(x) + 1) f(z) - d_T(z) f(z)$$

$$= f(z)$$

Suppose \hat{T} (having x, y as siblings) better than T : $B(\hat{T}) < B(T)$
collapse x, y to z , forming \hat{T}' .
 $B(\hat{T}) - B(\hat{T}') = f(z)$
Then
 $B(\hat{T}') = B(\hat{T}) - f(z) < B(T) - f(z) = B(T')$
contradiction

Theorem: H. gives optimal codes

Proof: induction on $|C|$

- Basis: $n=1, 2$ – immediate
- Induction: $n > 2$
 - Let x, y be least frequent
 - Form C', f', z , as above
 - By induction, T' is opt for (C', f')
 - By lemma 2, $T' \rightarrow T$ is opt for (C, f) among trees with x, y as siblings
 - By lemma 1, some opt tree does have x, y as siblings
 - Therefore, T is optimal.

CSE 421, W'01, Ruzzo 11

Data Compression

- Huffman is optimal.
- BUT still might do better!
 - Huffman encodes fixed length blocks. What if we vary them?
 - Huffman uses one encoding throughout a file. What if characteristics change?
 - What if data has structure? E.g. raster images, video, ...
 - Huffman is lossless. Necessary?
- LZW, MPEG, ...

CSE 421, W'01, Ruzzo 12