Projeto e Análise de Algoritmos (PAA)

Código de Huffman

Roteiro para projetar um algoritmo guloso:

- Compreender o problema
- Propor estratégias/critérios gulosos
- Resolver exemplos usando essas estratégias
- Selecionar a estratégia mais promissora
- Escrever o pseudocódigo do algoritmo
- Analisar a eficiência do algoritmo
- Provar a corretude do critério guloso

Códigos Binários de Comprimento Fixo

- mapeiam cada símbolo de um alfabeto Σ p/ uma seq. de K bits

Exemplo: a = 00, b = 01, c = 10, d = 11 e K = 2

Decodificar: 01,10,00,01,11,10 ⇒ bcabdc

⊛ Quiz: qual o valor mínimo de K em função de Σ? $|\Sigma| \leq 2^K$, por exemplo

↳ Se K já é mínimo, é possível economizar mais? K = 5 p/ o alfabeto latino

Comprimento Variável e Ambiguidade

- a ideia é usar menos bits p/ caracteres muito frequentes
  - mas alguns cuidados são necessários

Exemplo: considere a seguinte codificação a = 0, b = 01, c = 10, d = 1

Texto decodificar: 001 → ab ou → aad ?

⊛ Quiz: como evitar ambiguidade?

Códigos Livres de Prefixo

- Com comprimento variável não dá p/ contar K bits p/ saber o fim de um caractér
- a solução p/ ambiguidade é que o código de um caractér não pode ser prefixo do código de outro

Não exemplo: a = 0 e b = 01 é proibido

Exemplo a = 0, b = 10, c = 110, d = 111

Caractér	Freq
a	60%
b	25%
c	10%
d	5%

- Quiz: qual o comprimento por caractér no código fixo? R.: 2 bits
- Quiz: e o comp. médio do código variável?

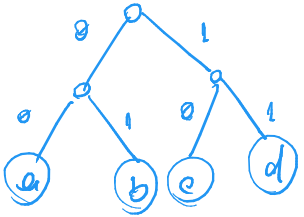
R.: $0,6 \cdot 1 + 0,25 \cdot 2 + 0,1 \cdot 3 + 0,5 \cdot 3 = 1,55$ bits

$0,6 + 0,5 + 0,3 + 0,15$

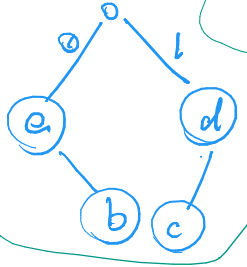
Relação entre Códigos e Árvores

- Para cada código binário temos uma árvore binária correspondente, e vice-versa

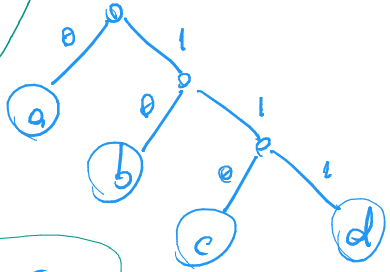
Exemplo 1: a = 00, b = 01, c = 10, d = 11



Exemplo 2: a = 0, b = 01, c = 10, d = 1



Exemplo 3: a = 0, b = 10, c = 110, d = 111



- O caminho da raiz até o caractér corresponde ao código (esq. = 0 e dir. = 1)

⊗ Quiz: qual característica deve ter uma árvore que corresponde a um código livre de prefixo? R: caracteres são folhas

Problema de Compressão

Entrada: Alfabeto Σ e frequências/probabilidade $p_i \geq 0$ p/ cada $i \in \Sigma$

Seja T uma árvore binária que codifica Σ

O custo médio de T é $L(T) = \sum_{i \in \Sigma} p_i \cdot h_i^T$

h_i^T → prof. de i em T = # de bits do código de i

Solução: T c/ menor custo médio

Estratégias Gulosas

- ideia 1: pegar menor freq. e colocar no menor código disponível ✗

Exemplo 1: $p_a = 0,6$; $p_b = 0,25$; $p_c = 0,10$; $p_d = 0,5$ → $L(1) = 1,55$

$a = 0$ $b = 10$ $c = 110$ $d = 111$

Exemplo 2: $p_a = 0,3$; $p_b = 0,25$; $p_c = 0,24$; $p_d = 0,21$

$L(2) = 0,3 \cdot 1 + 0,25 \cdot 2 + 0,24 \cdot 3 + 0,21 \cdot 3 = 2,15 > 2$ (comp. fixo)

$0,3$ $0,5$ $0,45 \cdot 3 = 1,35$

Estratégias Gulosas (Cont.)

importante

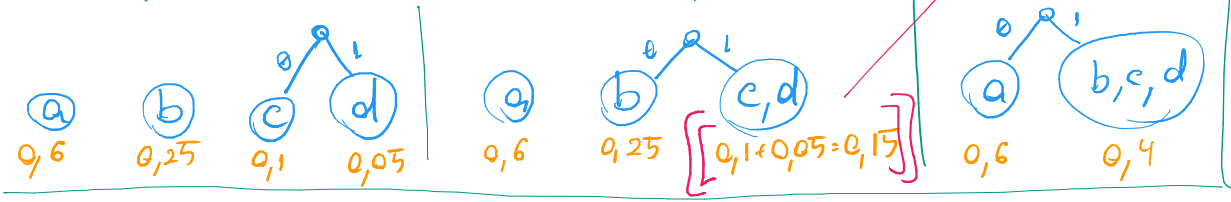
Voltando a pensar na árvore, unir (mesclar) dois nós e adicionar 1 bit ao código de cada um

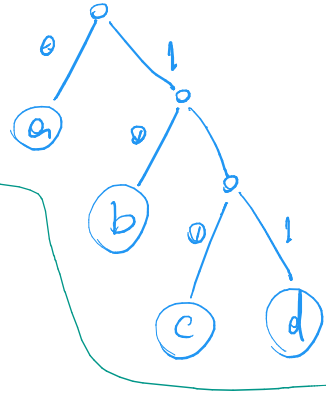
Importante p/ recursão !!!

- Assim, a questão é: quem é legal penalizar?

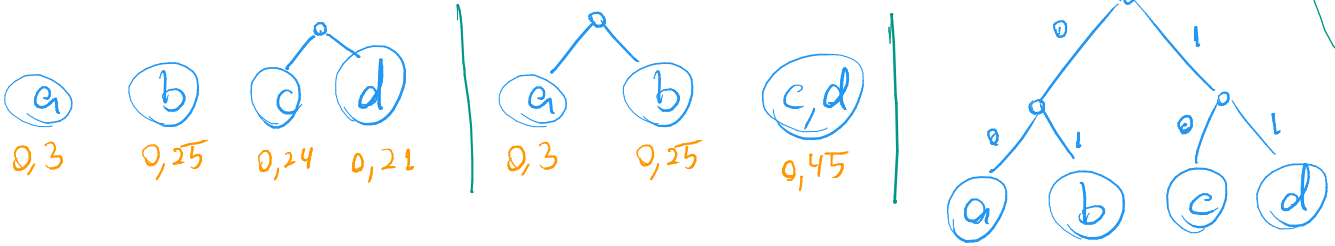
ideia 2: ir de baixo p/ cima "mesclando" nós com menor freq.

Exemplo 1: $p_a = 0,6$; $p_b = 0,25$; $p_c = 0,10$; $p_d = 0,5$





Exemplo 2: $p_a = 0,3$; $p_b = 0,25$; $p_c = 0,24$; $p_d = 0,21$



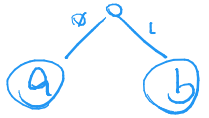
Essa estratégia parece promissora :) !!!

Subproblema Recursivo

- Estrutura recursiva do alg.:
  - "mescle" os dois símbolos u e v c/ menor frequência
  - novo símbolo u|v tem freq. $p_u + p_v$
  - resolva recursivamente o subprob. c/ 1 símbolo a menos

Exemplo ⊗ Quiz: qual o caso base da recursão?

HuffmanRec(Σ, p):

se $|\Sigma| == 2$: devolva 

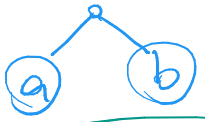
, c/ $a, b \in \Sigma$

Seja a e b o par de símbolos c/ menor freqs p_a, p_b

Funde a e b em a|b c/ freq. $p_a + p_b$

Seja $\Sigma' = \Sigma - \{a,b\} + \{a|b\}$

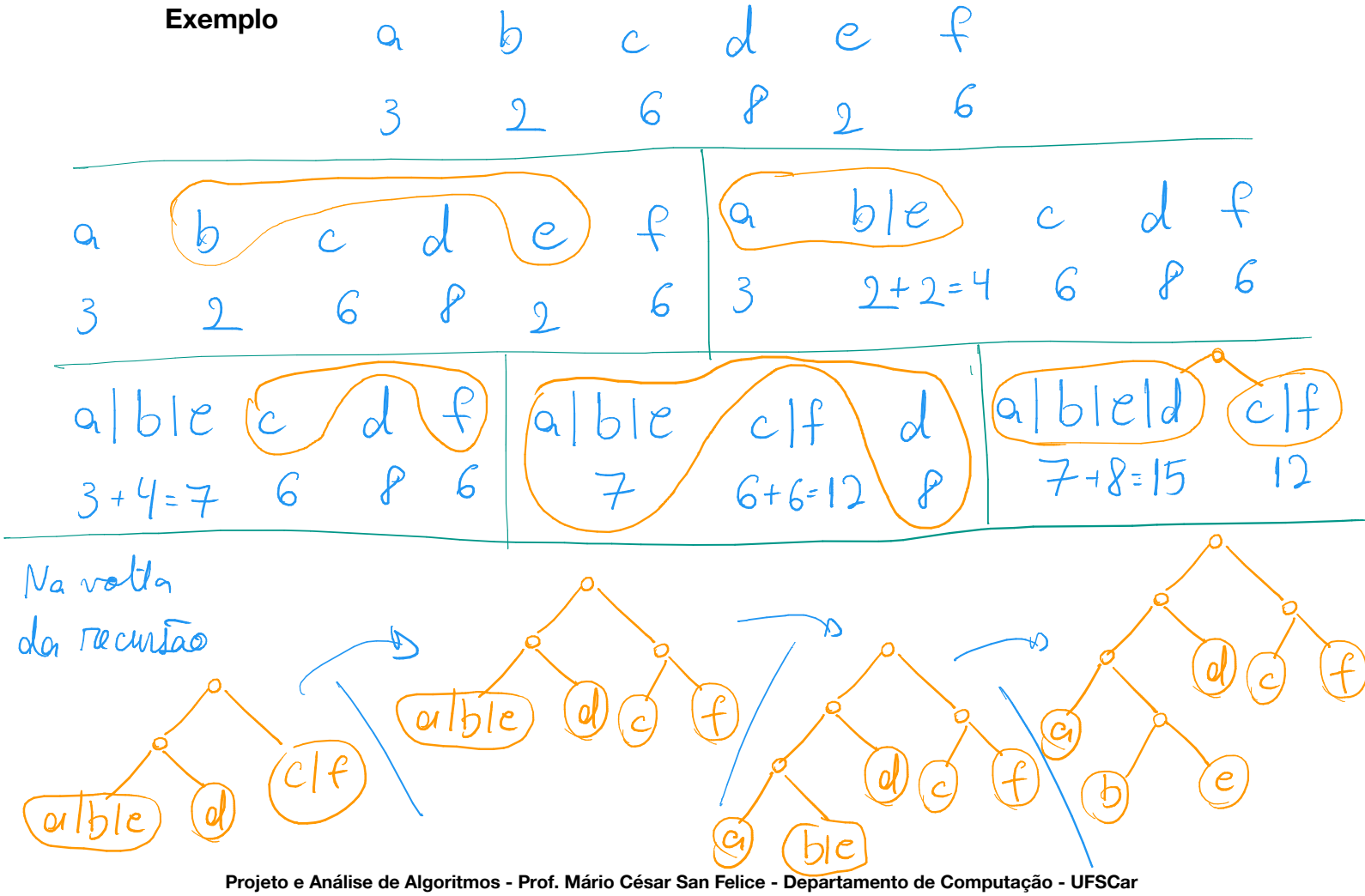
$T' =$ HuffmanRec(Σ', p')

Transforme T' em T expandindo a|b em 

Devolva T

* Note que a|b é folha em T', mas não necessariamente do último nível

Exemplo


a b c d e f
3 2 6 8 2 6
a b c d e f
3 2 6 8 2 6
a b|e c d f
3 2+2=4 6 8 6
a|b|e c d f
3+4=7 6 8 6
a|b|e c|f d
7 6+6=12 8
a|b|e|d c|f
7+8=15 12
Na volta da recursão
a|b|e d c f
a|b|e d c|f
a ble d c f
a b e d c f

Prova de Corretude do Algoritmo

Teorema: O alg. de Huffman produz uma árvore T de custo $L(T)$ mínimo c/ Σ nas folhas

Prova por indução em $n = |\Sigma|$

$$L(T) = \sum_{j \in \Sigma} p_j \cdot h_j^T$$

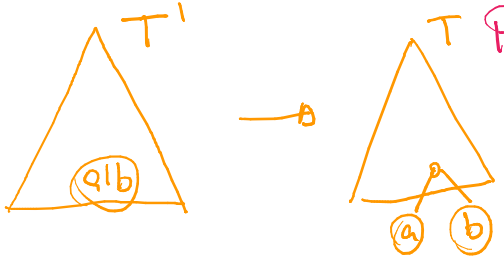
Base: $n = 2$ a única árvore possível é 

H.I.: T' produzida p/ Σ' que tem $n-1$ símbolos é ótima

Passo Indutivo

Seja $\Sigma' = \Sigma - \{a,b\} + \{a|b\}$ c/ a,b tendo freq. mínima em Σ

Observe que $\boxed{L(T) - L(T') = p_a \cdot h_a^T + p_b \cdot h_b^T - (p_a + p_b) \cdot h_{a|b}^{T'}}$



Prof. de a em T = Prof. de b em T = 1 + Prof. de $a|b$ em T'

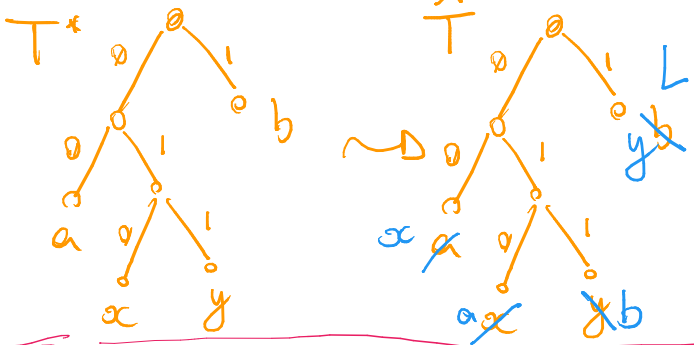
$= \boxed{p_a + p_b}$ e esse resultado vale p/ qualquer T'

Prova do Lema Central

Note que, pela H.I. a chamada recursiva devolve uma T' ótima p/ Σ'. Por isso, T obtida de T' é de custo mínimo dentre as árvores em que a e b são irmãos. Resta analisar o caso em que a e b não são irmãos numa árvore ótima T^*

decorre do resultado $L(T) - L(T') = p_a + p_b$ p/ qualquer T'

Vamos usar um argumento de troca, i.e., sendo x e y irmãos de nível mais baixo de T^*, troque x c/ a e y c/ b.



$$L(\hat{T}) - L(T^*) = p_a(h_x - h_a) + p_x(h_a - h_x) + p_b(h_y - h_b) + p_y(h_b - h_y)$$

$$= (p_a - p_x)(h_x - h_a) + (p_b - p_y)(h_y - h_b)$$

$p_a - p_x \geq 0$, $h_x - h_a \leq 0$, $p_b - p_y \geq 0$, $h_y - h_b \leq 0$, ≤ 0

Logo, sempre tem árvore ótima c/ a e b irmãos

Prova do Teorema

Como sempre existe uma árvore ótima em que os dois símbolos menos frequentes (a e b) são irmãos, e T gerada a partir de T' é uma árvore de custo mínimo p/ Σ dentre as que tem a e b sendo irmãos, temos que T é uma árvore ótima

Eficiência de Diferentes Implementações

- Implementação básica tem n chamadas recursivas (ou iterações) e trab. local $\Theta(n)$ p/ encontrar os dois símbolos menos frequentes em cada passo. Logo é $\Theta(n^2)$
- Usando heap de mínimo p/ manter os símbolos dá p/ fazer em tempo $\Theta(n \log n)$

(*) Quiz: também dá p/ fazer ordenando no início e usando 2 filas ou 1 lista. Como???