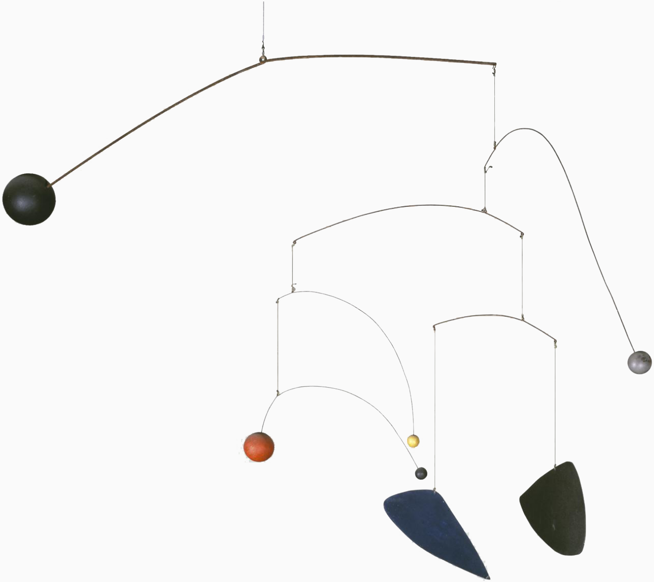
Balanced Mobiles

Yassine Hamoudi, Sophie Laplante, Roberto Mantaci

May 17, 2017

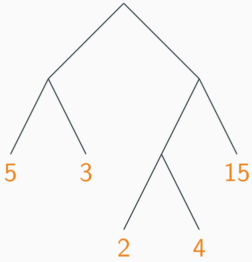
ENS Lyon – IRIF, Paris Diderot



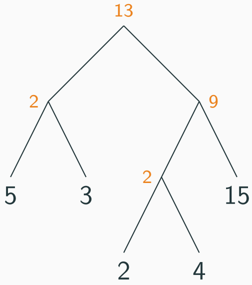
Mobile. Calder, 1932.

Introduction

A mobile M is a full binary tree in which each leaf has a (positive) weight.

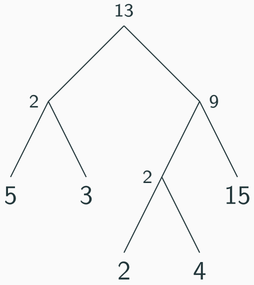


The local imbalance δ of a node is the difference (in absolute value) between the weights W_L and W_R of the left and right subtrees of the node.



Introduction

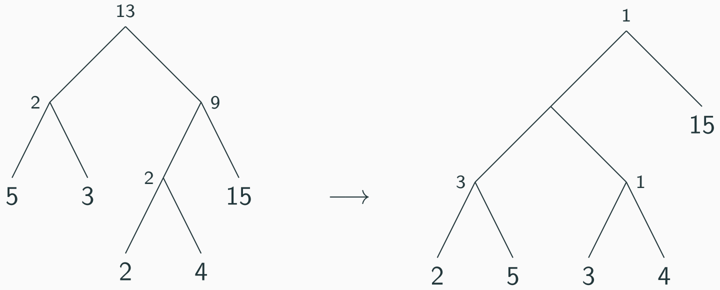
The imbalance Δ_M of M is the sum of all the local imbalances.



$$\Delta_M = 13 + 2 + 9 + 2 = 26$$

Introduction

The Balanced Mobiles problem consists in constructing, for a given set of weights $\{w_1, \ldots, w_n\}$, a mobile of imbalance as small as possible.



$\Delta_M = 26$ $\qquad \Delta_M = 5$

Contents

The Smallest algorithm

The Smallest algorithm ($\sim$ Huffman coding)

Input: weights $w_1 \leq \cdots \leq w_n$

Output: imbalance $\textsc{Smallest}(w_1, \ldots, w_n)$

if $n = 2$ **then**

  Return $|w_2 - w_1|$

else

  Return $|w_2 - w_1| + \textsc{Smallest}(\textsc{Sort}(w_1 + w_2, w_3, \ldots, w_n))$

The Smallest algorithm ($\sim$ Huffman coding)

Input: weights $w_1 \leq \cdots \leq w_n$
Output: imbalance $\textsc{Smallest}(w_1, \ldots, w_n)$
if $n = 2$ **then**
  Return $|w_2 - w_1|$
else
  Return $|w_2 - w_1| + \textsc{Smallest}(\textsc{Sort}(w_1 + w_2, w_3, \ldots, w_n))$

1 2 2 3 4 6

The Smallest algorithm ($\sim$ Huffman coding)

Input: weights $w_1 \leq \cdots \leq w_n$

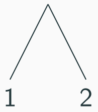
Output: imbalance $\textsc{Smallest}(w_1, \ldots, w_n)$

if $n = 2$ **then**

  Return $|w_2 - w_1|$

else

  Return $|w_2 - w_1| + \textsc{Smallest}(\textsc{Sort}(w_1 + w_2, w_3, \ldots, w_n))$



3   2   3   4   6

The Smallest algorithm ($\sim$ Huffman coding)

Input: weights $w_1 \leq \cdots \leq w_n$

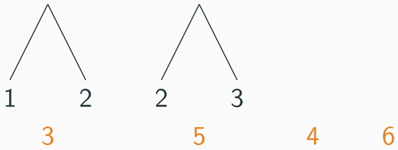
Output: imbalance $\textsc{Smallest}(w_1, \ldots, w_n)$

if $n = 2$ **then**

  Return $|w_2 - w_1|$

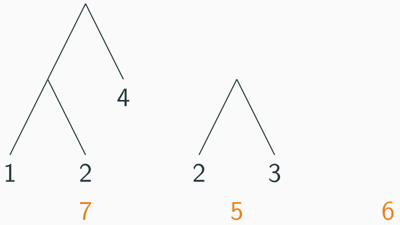
else

  Return $|w_2 - w_1| + \textsc{Smallest}(\textsc{Sort}(w_1 + w_2, w_3, \ldots, w_n))$



The Smallest algorithm ($\sim$ Huffman coding)

Input: weights $w_1 \leq \cdots \leq w_n$
Output: imbalance $\textsc{Smallest}(w_1, \ldots, w_n)$
if $n = 2$ **then**
  Return $|w_2 - w_1|$
else
  Return $|w_2 - w_1| + \textsc{Smallest}(\textsc{Sort}(w_1 + w_2, w_3, \ldots, w_n))$



The Smallest algorithm ($\sim$ Huffman coding)

Input: weights $w_1 \leq \cdots \leq w_n$

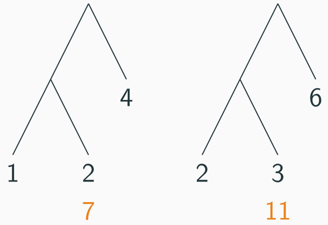
Output: imbalance $\textsc{Smallest}(w_1, \ldots, w_n)$

if $n = 2$ **then**

  Return $|w_2 - w_1|$

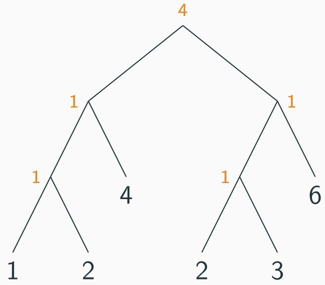
else

  Return $|w_2 - w_1| + \textsc{Smallest}(\textsc{Sort}(w_1 + w_2, w_3, \ldots, w_n))$



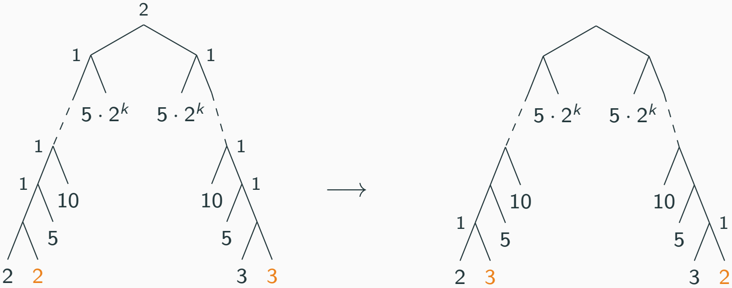
The Smallest algorithm ($\sim$ Huffman coding)

Input: weights $w_1 \leq \cdots \leq w_n$
Output: imbalance $\textsc{Smallest}(w_1, \ldots, w_n)$
if $n = 2$ **then**
  Return $|w_2 - w_1|$
else
  Return $|w_2 - w_1| + \textsc{Smallest}(\textsc{Sort}(w_1 + w_2, w_3, \ldots, w_n))$



$\Delta = 8$

Non-optimality of Smallest



$\Delta = 4 + 2k$ $\qquad$ $\Delta = 2$

Optimal cases for Smallest

Theorem

The SMALLEST *algorithm is optimal in the following cases:*

- *when the smallest possible imbalance is 0 or 1*
- *when all the weights are equal*
- *when all the weights are powers of two*

It runs in $\mathcal{O}(n \log n)$ *time.*

The R-Smallest algorithm

Given two mobiles of weights A and B, if $w_1 \leq A \leq B$ then this rotation does not increase the imbalance:

The R-Smallest algorithm

Given two mobiles of weights A and B, if $w_1 \leq A \leq B$ then this rotation does not increase the imbalance:

Lemma

For any weights $w_1 \leq \cdots \leq w_n$, there exists an optimal mobile in which the sibling of the leaf of weight w_1 is also a leaf.

The R-Smallest algorithm

Given $w_1 \leq \cdots \leq w_n$ and a threshold δ, try for all i to find a mobile of imbalance $\leq \delta - |w_1 - w_i|$ on $\{w_1 + w_i, w_2, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n\}$.

The R-Smallest algorithm

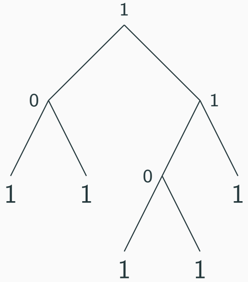
Given $w_1 \leq \cdots \leq w_n$ and a threshold δ, try for all i to find a mobile of imbalance $\leq \delta - |w_1 - w_i|$ on $\{w_1 + w_i, w_2, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n\}$.

Theorem

For any weights $w_1, \ldots, w_n$, the R-SMALLEST *algorithm finds the optimal imbalance Δ in time $\mathcal{O}\big(\log(n) n^{\min(\Delta, n)+1}\big)$.*

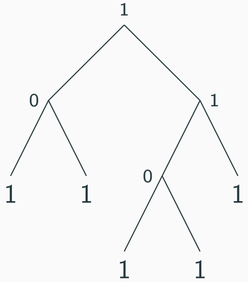
All-Equal Weights

Δ_n : optimal imbalance for the weights $w_1 = \cdots = w_n = 1$



$\Delta_5 = 2$

Δ_n : optimal imbalance for the weights $w_1 = \cdots = w_n = 1$



$\Delta_5 = 2$

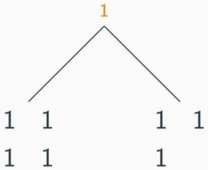
→ the SMALLEST algorithm is optimal in this case

→ a PARTITION algorithm (inspired from 2-PARTITION) is also optimal

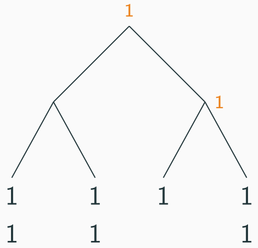
Partition algorithm

$$\begin{matrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & \end{matrix}$$

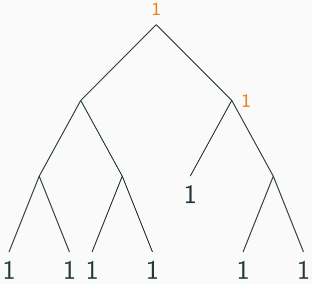
Partition algorithm



Partition algorithm



Partition algorithm



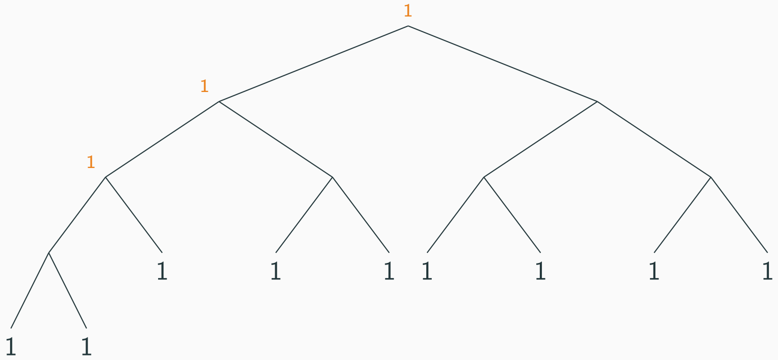
$\Delta = 2$

First recurrence relation for Δ_n

The optimal imbalance Δ_n verifies:

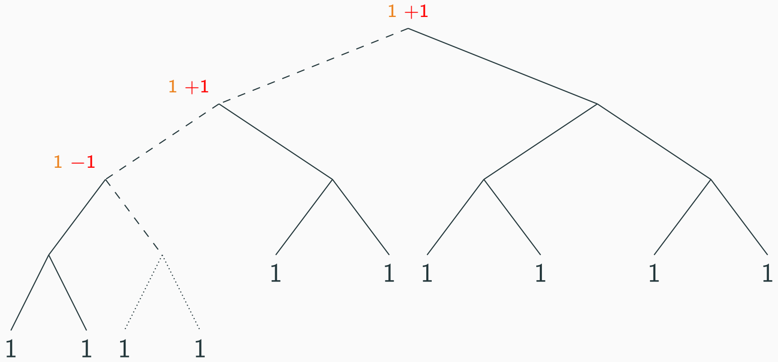
$$\begin{cases} \Delta_1 = 0 \\ \Delta_{2n} = 2\Delta_n \\ \Delta_{2n+1} = 1 + \Delta_n + \Delta_{n+1} \end{cases}$$

Smallest algorithm



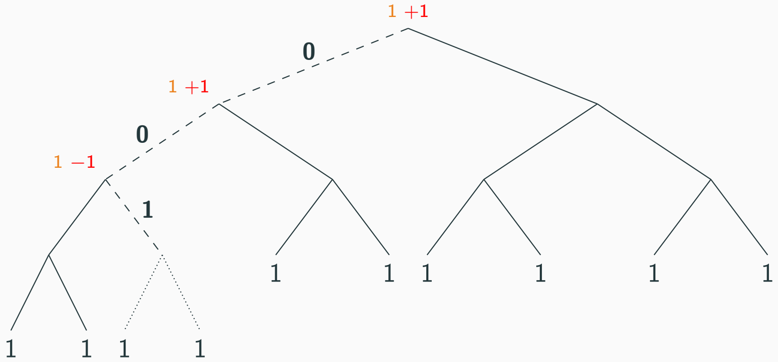
A mobile over 9 weights, built with SMALLEST.

Smallest algorithm



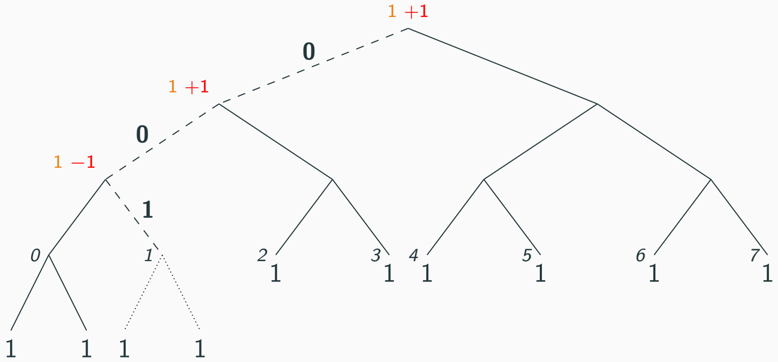
A mobile over 10 weights, built with SMALLEST.

Smallest algorithm



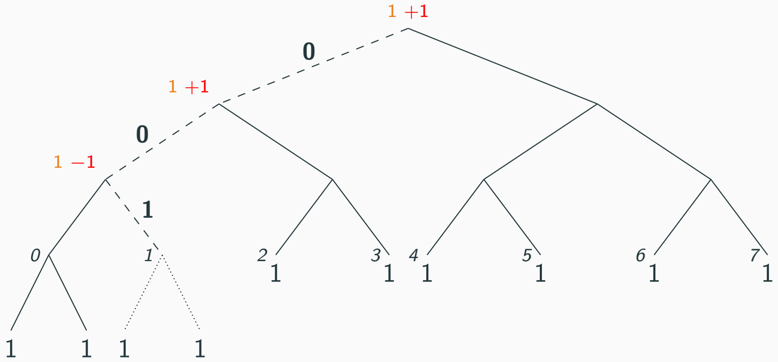
Increase of $|001|_0 - |001|_1$

Smallest algorithm



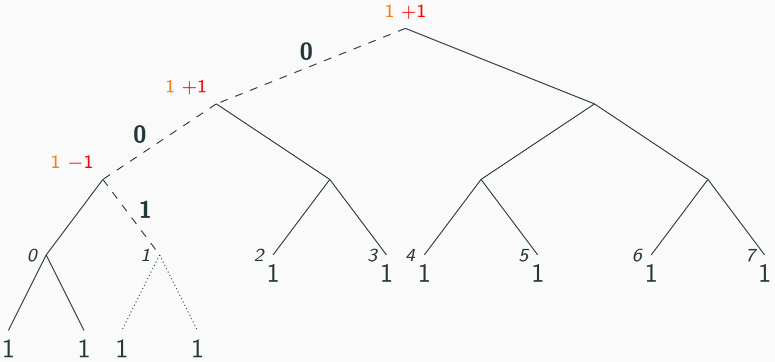
Increase of $|001|_0 - |001|_1 = |1001|_0 - (|1001|_1 - 1)$

Smallest algorithm



Increase of $|001|_0 - |001|_1 = \left|2^3+1\right|_0 - \left(\left|2^3+1\right|_1 - 1\right)$

Smallest algorithm



Increase of $|001|_0 - |001|_1 = \left|2^3+1\right|_0 - \left(\left|2^3+1\right|_1 - 1\right)$

$$S_{10} = S_9 + |9|_0 - |9|_1 + 1$$

Second recurrence relation for Δ_n

The imbalance S_n obtained by SMALLEST verifies:

$$\begin{cases} S_1 = 0 \\ S_{n+1} = S_n + |n|_0 - |n|_1 + 1 \end{cases}$$

Second recurrence relation for Δ_n

The imbalance S_n obtained by SMALLEST verifies:

$$\begin{cases} S_1 = 0 \\ S_{n+1} = S_n + |n|_0 - |n|_1 + 1 \end{cases}$$

Proposition

Using $\Delta_{2n} = 2\Delta_n, \Delta_{2n+1} = 1 + \Delta_n + \Delta_{n+1}$ *we prove* $S_n = \Delta_n$.

The Δ_n function

$$\begin{cases} \Delta_1 = 0 \\ \Delta_{2n} = 2\Delta_n \\ \Delta_{2n+1} = 1 + \Delta_n + \Delta_{n+1} \end{cases} \quad \text{and} \quad \begin{cases} \Delta_1 = 0 \\ \Delta_{n+1} = \Delta_n + |n|_0 - |n|_1 + 1 \end{cases}$$

The Δ_n function

$$\begin{cases} \Delta_1 = 0 \\ \Delta_{2n} = 2\Delta_n \\ \Delta_{2n+1} = 1 + \Delta_n + \Delta_{n+1} \end{cases} \quad \text{and} \quad \begin{cases} \Delta_1 = 0 \\ \Delta_{n+1} = \Delta_n + |n|_0 - |n|_1 + 1 \end{cases}$$

If $b_k b_{k-1} \dots b_0$ is the binary representation of n :

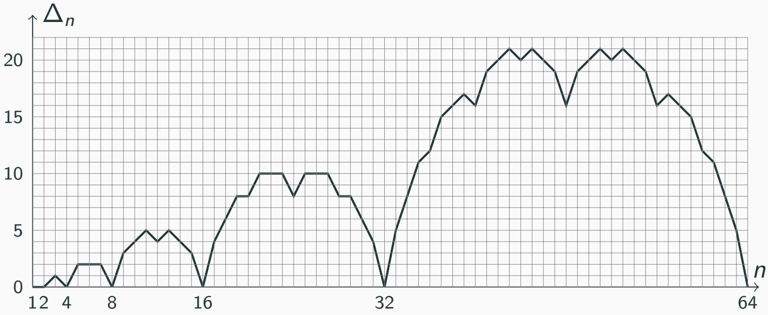
$$\Delta_n = 2 \cdot (n \mod 2^k) + \sum_{i=0}^{k-1} (-1)^{b_i} \cdot (n \mod 2^{i+1})$$

The Δ_n function

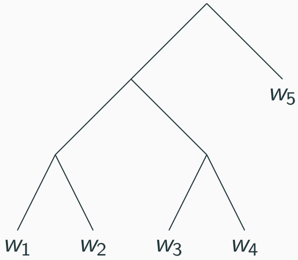
$$\begin{cases} \Delta_1 = 0 \\ \Delta_{2n} = 2\Delta_n \\ \Delta_{2n+1} = 1 + \Delta_n + \Delta_{n+1} \end{cases} \quad \text{and} \quad \begin{cases} \Delta_1 = 0 \\ \Delta_{n+1} = \Delta_n + |n|_0 - |n|_1 + 1 \end{cases}$$

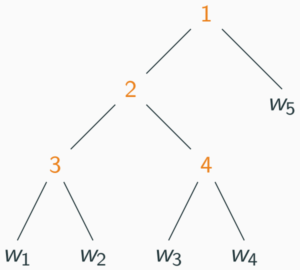
If $b_k b_{k-1} \dots b_0$ is the binary representation of n :

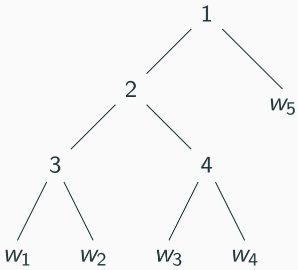
$$\Delta_n = 2 \cdot (n \mod 2^k) + \sum_{i=0}^{k-1} (-1)^{b_i} \cdot (n \mod 2^{i+1})$$



Integer Linear Programming


w_5
w_1 w_2 w_3 w_4

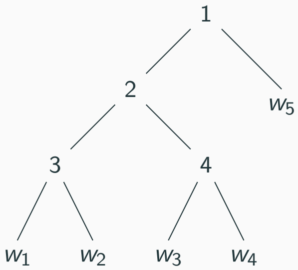

1
2
w_5
3
4
w_1
w_2
w_3
w_4



	1	2	3	4
w_1	$\ell_{1,1} = 1$	$\ell_{1,2} = 1$	$\ell_{1,3} = 1$	$\ell_{1,4} = 0$
	$r_{1,1} = 0$	$r_{1,2} = 0$	$r_{1,3} = 0$	$r_{1,4} = 0$
w_2	$\ell_{2,1} = 1$	$\ell_{2,2} = 1$	$\ell_{2,3} = 0$	$\ell_{2,4} = 0$
	$r_{2,1} = 0$	$r_{2,2} = 0$	$r_{2,3} = 1$	$r_{2,4} = 0$
w_3	$\ell_{3,1} = 1$	$\ell_{3,2} = 0$	$\ell_{3,3} = 0$	$\ell_{3,4} = 1$
	$r_{3,1} = 0$	$r_{3,2} = 1$	$r_{3,3} = 0$	$r_{3,4} = 0$
w_4	$\ell_{4,1} = 1$	$\ell_{4,2} = 0$	$\ell_{4,3} = 0$	$\ell_{4,4} = 0$
	$r_{4,1} = 0$	$r_{4,2} = 1$	$r_{4,3} = 0$	$r_{4,4} = 1$
w_5	$\ell_{5,1} = 0$	$\ell_{5,2} = 0$	$\ell_{5,3} = 0$	$\ell_{5,4} = 0$
	$r_{5,1} = 1$	$r_{5,2} = 0$	$r_{5,3} = 0$	$r_{5,4} = 0$

Define:

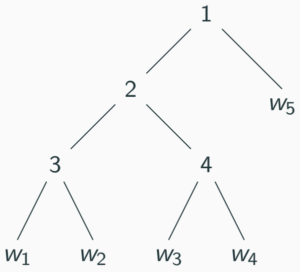
- $\ell_{i,u} = 1$ if w_i is in the left subtree of the node u, 0 otherwise
- $r_{i,u} = 1$ if w_i is in the right subtree of the node u, 0 otherwise



	1	2	3	4
w_1	$\ell_{1,1} = 1$ $r_{1,1} = 0$	$\ell_{1,2} = 1$ $r_{1,2} = 0$	$\ell_{1,3} = 1$ $r_{1,3} = 0$	$\ell_{1,4} = 0$ $r_{1,4} = 0$
w_2	$\ell_{2,1} = 1$ $r_{2,1} = 0$	$\ell_{2,2} = 1$ $r_{2,2} = 0$	$\ell_{2,3} = 0$ $r_{2,3} = 1$	$\ell_{2,4} = 0$ $r_{2,4} = 0$
w_3	$\ell_{3,1} = 1$ $r_{3,1} = 0$	$\ell_{3,2} = 0$ $r_{3,2} = 1$	$\ell_{3,3} = 0$ $r_{3,3} = 0$	$\ell_{3,4} = 1$ $r_{3,4} = 0$
w_4	$\ell_{4,1} = 1$ $r_{4,1} = 0$	$\ell_{4,2} = 0$ $r_{4,2} = 1$	$\ell_{4,3} = 0$ $r_{4,3} = 0$	$\ell_{4,4} = 0$ $r_{4,4} = 1$
w_5	$\ell_{5,1} = 0$ $r_{5,1} = 1$	$\ell_{5,2} = 0$ $r_{5,2} = 0$	$\ell_{5,3} = 0$ $r_{5,3} = 0$	$\ell_{5,4} = 0$ $r_{5,4} = 0$

The imbalance is:

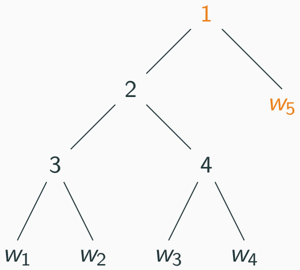
$$\sum_{u=1}^{n-1} \left| \sum_{i=1}^{n} w_i(\ell_{i,u} - r_{i,u}) \right|$$



	1	2	3	4
w_1	$\ell_{1,1} = 1$ $r_{1,1} = 0$	$\ell_{1,2} = 1$ $r_{1,2} = 0$	$\ell_{1,3} = 1$ $r_{1,3} = 0$	$\ell_{1,4} = 0$ $r_{1,4} = 0$
w_2	$\ell_{2,1} = 1$ $r_{2,1} = 0$	$\ell_{2,2} = 1$ $r_{2,2} = 0$	$\ell_{2,3} = 0$ $r_{2,3} = 1$	$\ell_{2,4} = 0$ $r_{2,4} = 0$
w_3	$\ell_{3,1} = 1$ $r_{3,1} = 0$	$\ell_{3,2} = 0$ $r_{3,2} = 1$	$\ell_{3,3} = 0$ $r_{3,3} = 0$	$\ell_{3,4} = 1$ $r_{3,4} = 0$
w_4	$\ell_{4,1} = 1$ $r_{4,1} = 0$	$\ell_{4,2} = 0$ $r_{4,2} = 1$	$\ell_{4,3} = 0$ $r_{4,3} = 0$	$\ell_{4,4} = 0$ $r_{4,4} = 1$
w_5	$\ell_{5,1} = 0$ $r_{5,1} = 1$	$\ell_{5,2} = 0$ $r_{5,2} = 0$	$\ell_{5,3} = 0$ $r_{5,3} = 0$	$\ell_{5,4} = 0$ $r_{5,4} = 0$

The weight w_i cannot be simultaneously in the left and right subtrees of the node u.

$$\forall i, u, \ell_{i,u} + r_{i,u} \leq 1$$



	1	2	3	4
w_1	$\ell_{1,1} = 1$	$\ell_{1,2} = 1$	$\ell_{1,3} = 1$	$\ell_{1,4} = 0$
	$r_{1,1} = 0$	$r_{1,2} = 0$	$r_{1,3} = 0$	$r_{1,4} = 0$
w_2	$\ell_{2,1} = 1$	$\ell_{2,2} = 1$	$\ell_{2,3} = 0$	$\ell_{2,4} = 0$
	$r_{2,1} = 0$	$r_{2,2} = 0$	$r_{2,3} = 1$	$r_{2,4} = 0$
w_3	$\ell_{3,1} = 1$	$\ell_{3,2} = 0$	$\ell_{3,3} = 0$	$\ell_{3,4} = 1$
	$r_{3,1} = 0$	$r_{3,2} = 1$	$r_{3,3} = 0$	$r_{3,4} = 0$
w_4	$\ell_{4,1} = 1$	$\ell_{4,2} = 0$	$\ell_{4,3} = 0$	$\ell_{4,4} = 0$
	$r_{4,1} = 0$	$r_{4,2} = 1$	$r_{4,3} = 0$	$r_{4,4} = 1$
w_5	$\ell_{5,1} = 0$	$\ell_{5,2} = 0$	$\ell_{5,3} = 0$	$\ell_{5,4} = 0$
	$r_{5,1} = 1$	$r_{5,2} = 0$	$r_{5,3} = 0$	$r_{5,4} = 0$

If the weight w_i is the right (resp. left) child of the node u, then none of the other leaves can be in the right (resp. left) subtree of the node u.

$$\forall i \neq j, \forall u, \begin{cases} (1 - \ell_{i,u}) + \sum_{v>u}(\ell_{i,v} + r_{i,v}) \geq \ell_{j,u} \\ (1 - r_{i,u}) + \sum_{v>u}(\ell_{i,v} + r_{i,v}) \geq r_{j,u} \end{cases}$$

$$\texttt{Minimize} \sum_{u=1}^{n-1} \left| \sum_{i=1}^{n} w_i(\ell_{i,u} - r_{i,u}) \right| \text{ subject to:}$$

$\forall i, u, \ell_{i,u} + r_{i,u} \leq 1$

$\forall i, \ell_{i,1} + r_{i,1} = 1$

$\forall u, \sum_i \ell_{i,u} > 0$ and $\sum_i r_{i,u} > 0$

$$\forall i \neq j, \forall u, \begin{cases} (1 - \ell_{i,u}) + \sum_{v>u}(\ell_{i,v} + r_{i,v}) \geq \ell_{j,u} \\ (1 - r_{i,u}) + \sum_{v>u}(\ell_{i,v} + r_{i,v}) \geq r_{j,u} \end{cases}$$

$$\forall i \neq j, \forall u < v, \begin{cases} \ell_{i,u} + (\ell_{i,v} + r_{i,v} + \ell_{j,v} + r_{j,v}) \leq 2 + \ell_{j,u} \\ r_{i,u} + (\ell_{i,v} + r_{i,v} + \ell_{j,v} + r_{j,v}) \leq 2 + r_{j,u} \end{cases}$$

$$\forall i \neq j, \forall u < u', \begin{cases} 2 - \ell_{i,u} - \ell_{j,u} + \sum_{w=u+1}^{u'}(\ell_{i,w} + r_{i,w}) \geq r_{j,u'} + r_{j,u'} \\ 2 - r_{i,u} - r_{j,u} + \sum_{w=u+1}^{u'}(\ell_{i,w} + r_{i,w}) \geq \ell_{j,u'} + \ell_{j,u'} \end{cases}$$

The Evaluation Trees problem

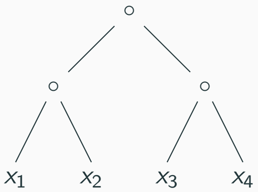
Non-Abelian Evaluation Trees

Problem

Given n elements $x_1, \ldots, x_n$ of a set $\mathcal{X}$ equipped with an associative operator $\circ : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ and a cost function $c : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$, find the optimal evaluation tree to compute $x_1 \circ x_2 \circ \ldots \circ x_n$.

Problem

Given n elements $x_1, \ldots, x_n$ of a set $\mathcal{X}$ equipped with an associative operator $\circ : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ and a cost function $c : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$, find the optimal evaluation tree to compute $x_1 \circ x_2 \circ \ldots \circ x_n$.

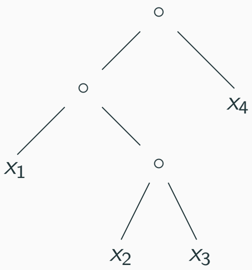


$$((x_1 \circ x_2) \circ (x_3 \circ x_4))$$

Non-Abelian Evaluation Trees

Problem

Given n elements $x_1, \ldots, x_n$ of a set $\mathcal{X}$ equipped with an associative operator $\circ : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ and a cost function $c : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$, find the optimal evaluation tree to compute $x_1 \circ x_2 \circ \ldots \circ x_n$.

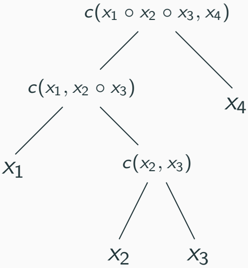


$$((x_1 \circ (x_2 \circ x_3)) \circ x_4)$$

Non-Abelian Evaluation Trees

Problem

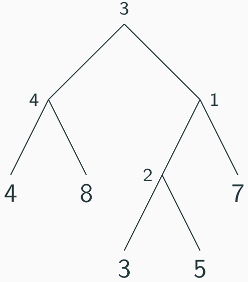
Given n elements $x_1, \ldots, x_n$ of a set $\mathcal{X}$ equipped with an associative operator $\circ : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ and a cost function $c : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$, find the optimal evaluation tree to compute $x_1 \circ x_2 \circ \ldots \circ x_n$.



$$c(x_1 \circ x_2 \circ x_3, x_4) + c(x_1, x_2 \circ x_3) + c(x_2, x_3)$$

Ordered Balanced Mobiles

Given a sequence of weights $(w_1, \ldots, w_n)$, find a mobile of imbalance as small as possible with these weights in the same order from left to right.



An optimal mobile for the sequence $(4, 8, 3, 5, 7)$.

Ordered Balanced Mobiles

Given a sequence of weights $(w_1, \ldots, w_n)$, find a mobile of imbalance as small as possible with these weights in the same order from left to right.

This is a Non-Abelian Evaluation Trees problem with:

- $x_i = w_i$ and $\mathcal{X} = \mathbb{N}$
- $c(x, y) = |x - y|$
- $x \circ y = x + y$

Matrix Chain Multiplication

Given a sequence of matrices $(M_1, \ldots, M_n)$ where $dim(M_i) = (n_{i-1}, n_i)$, find the optimal way to compute the product $M_1 \times \cdots \times M_n$.

Matrix Chain Multiplication

Given a sequence of matrices $(M_1, \ldots, M_n)$ where $dim(M_i) = (n_{i-1}, n_i)$, find the optimal way to compute the product $M_1 \times \cdots \times M_n$.

This is a Non-Abelian Evaluation Trees problem with:

- $x_i = (n_{i-1}, n_i)$ and $\mathcal{X} = \mathbb{N} \times \mathbb{N}$
- $c(x, y) = n \times m \times k$ where $x = (n, m)$ and $y = (m, k)$
- $x \circ y = (n, k)$ where $x = (n, m)$ and $y = (m, k)$

Matrix Chain Multiplication

Given a sequence of matrices $(M_1, \ldots, M_n)$ where $dim(M_i) = (n_{i-1}, n_i)$, find the optimal way to compute the product $M_1 \times \cdots \times M_n$.

This is a Non-Abelian Evaluation Trees problem with:

- $x_i = (n_{i-1}, n_i)$ and $\mathcal{X} = \mathbb{N} \times \mathbb{N}$
- $c(x, y) = n \times m \times k$ where $x = (n, m)$ and $y = (m, k)$
- $x \circ y = (n, k)$ where $x = (n, m)$ and $y = (m, k)$

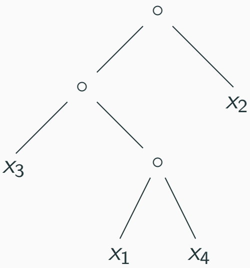
Dynamic programming in $\mathcal{O}\left(n^3\right)$:

$$C[i,j] = \begin{cases} 0 & \text{if } i = j \\ \min\limits_{i \leq k < j} \{C[i,k] + C[k+1,j] + c(x_i \circ \cdots \circ x_k, x_{k+1} \circ \cdots \circ x_j), \} & \text{if } i < j \end{cases}$$

Abelian Evaluation Trees

Problem

Given n elements $x_1, \ldots, x_n$ of a set $\mathcal{X}$ equipped with an associative and commutative operator $\circ : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ and a cost function $c : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$, find the optimal evaluation tree to compute $x_1 \circ x_2 \circ \ldots \circ x_n$.



$$((x_3 \circ (x_1 \circ x_4)) \circ x_2)$$

Balanced Mobiles

Given a set of weights $\{w_1, \ldots, w_n\}$, find a mobile of imbalance as small as possible with these weights.

This is an $\textsc{Abelian Evaluation Trees}$ problem with:

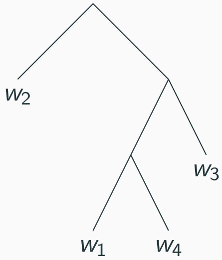
- $x_i = w_i$ and $\mathcal{X} = \mathbb{N}$
- $c(x, y) = |x - y|$
- $x \circ y = x + y$

Huffman Coding

Given an alphabet $\{a_1, \ldots, a_n\}$ and the number of occurencies w_i of each a_i, find a prefix-free binary code $(c_1, \ldots, c_n)$ that minimizes $\sum_i w_i \cdot |c_i|$.

Huffman Coding

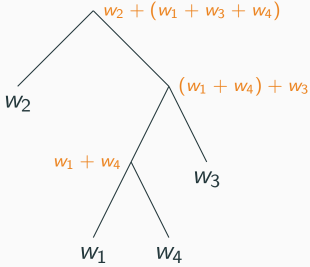
Given an alphabet $\{a_1, \ldots, a_n\}$ and the number of occurencies w_i of each a_i, find a prefix-free binary code $(c_1, \ldots, c_n)$ that minimizes $\sum_i w_i \cdot |c_i|$.



$$3w_1 + w_2 + 2w_3 + 3w_4$$

Huffman Coding

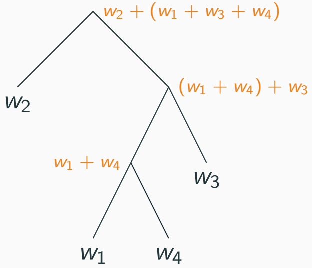
Given an alphabet $\{a_1, \ldots, a_n\}$ and the number of occurencies w_i of each a_i, find a prefix-free binary code $(c_1, \ldots, c_n)$ that minimizes $\sum_i w_i \cdot |c_i|$.



$$3w_1 + w_2 + 2w_3 + 3w_4$$

Huffman Coding

Given an alphabet $\{a_1, \ldots, a_n\}$ and the number of occurencies w_i of each a_i, find a prefix-free binary code $(c_1, \ldots, c_n)$ that minimizes $\sum_i w_i \cdot |c_i|$.



$$3w_1 + w_2 + 2w_3 + 3w_4$$

This is an Abelian Evaluation Trees problem with:

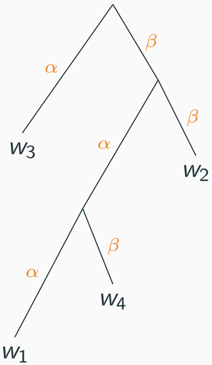
- $x_i = w_i$ and $\mathcal{X} = \mathbb{N}$
- $c(x, y) = x + y$
- $x \circ y = x + y$

Generalized Huffman Coding

The coding alphabet is made of two letters of unequal lengths α and β.

Generalized Huffman Coding

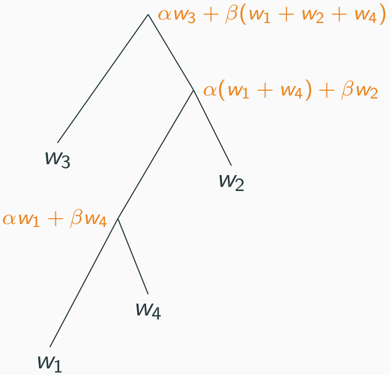
The coding alphabet is made of two letters of unequal lengths α and β.



$$(2\alpha + \beta)w_1 + 2\beta w_2 + \alpha w_3 + (\alpha + 2\beta)w_4$$

Generalized Huffman Coding

The coding alphabet is made of two letters of unequal lengths α and β.



$$(2\alpha + \beta)w_1 + 2\beta w_2 + \alpha w_3 + (\alpha + 2\beta)w_4$$

Generalized Huffman Coding

The coding alphabet is made of two letters of unequal lengths α and β.

This is an Abelian Evaluation Trees problem with:

- $x_i = w_i$ and $\mathcal{X} = \mathbb{N}$
- $c(x, y) = \alpha x + \beta y$
- $x \circ y = x + y$

Generalized Huffman Coding

The coding alphabet is made of two letters of unequal lengths α and β.

- The case $\alpha = \beta$ is Huffman Coding [Huf52]. This is solved in $\mathcal{O}(n \log n)$ by Smallest.
- The case $w_1 = \cdots = w_n$ is solved in poly-time [Var71, GY96, CG01].
- First known algorithm for the general case is an ILP [Kar61].
- Dynamic programming algorithm in $\mathcal{O}\left(n^{\max(\alpha,\beta)}\right)$ [GR98, BGLR02].
- PTAS [GMY12]

No poly-time algorithm for the general case nor it is known to be NP-hard.

Dynamic programming

Using dynamic programming, compute for all $S \subseteq \{1, \ldots, n\}$ the optimal cost $C(S)$ for $\circ_{i \in S} x_i$.

$$C(S) = \min_{S' \subseteq S, S' \neq \varnothing} c\left(\mathop{\circ}_{i \in S'} x_i, \mathop{\circ}_{i \in S \setminus S'} x_i\right) + C(S') + C(S \setminus S')$$

It runs in $2^{\mathcal{O}(n)}$ time and $\mathcal{O}(2^n)$ space.

Integer Linear Programming (Balanced Mobiles)

$$\texttt{Minimize} \sum_{u=1}^{n-1} \left| \sum_{i=1}^{n} (\ell_{i,u} - r_{i,u}) \cdot w_i \right| \text{ subject to:}$$

$\forall i, u, \ell_{i,u} + r_{i,u} \leq 1$

$\forall i, \ell_{i,1} + r_{i,1} = 1$

$\forall u, \sum_i \ell_{i,u} > 0$ and $\sum_i r_{i,u} > 0$

$$\forall i \neq j, \forall u, \begin{cases} (1 - \ell_{i,u}) + \sum_{v>u} (\ell_{i,v} + r_{i,v}) \geq \ell_{j,u} \\ (1 - r_{i,u}) + \sum_{v>u} (\ell_{i,v} + r_{i,v}) \geq r_{j,u} \end{cases}$$

$$\forall i \neq j, \forall u < v, \begin{cases} \ell_{i,u} + (\ell_{i,v} + r_{i,v} + \ell_{j,v} + r_{j,v}) \leq 2 + \ell_{j,u} \\ r_{i,u} + (\ell_{i,v} + r_{i,v} + \ell_{j,v} + r_{j,v}) \leq 2 + r_{j,u} \end{cases}$$

$$\forall i \neq j, \forall u < u', \begin{cases} 2 - \ell_{i,u} - \ell_{j,u} + \sum_{w=u+1}^{u'} (\ell_{i,w} + r_{i,w}) \geq r_{j,u'} + r_{j,u'} \\ 2 - r_{i,u} - r_{j,u} + \sum_{w=u+1}^{u'} (\ell_{i,w} + r_{i,w}) \geq \ell_{j,u'} + \ell_{j,u'} \end{cases}$$

Integer Linear Programming (Huffman Coding)

$$\texttt{Minimize} \sum_{u=1}^{n-1} \sum_{i=1}^{n} (\alpha \cdot \ell_{i,u} + \beta \cdot r_{i,u}) \cdot w_i \text{ subject to:}$$

$$\forall i, u, \ell_{i,u} + r_{i,u} \leq 1$$

$$\forall i, \ell_{i,1} + r_{i,1} = 1$$

$$\forall u, \textstyle\sum_i \ell_{i,u} > 0 \text{ and } \sum_i r_{i,u} > 0$$

$$\forall i \neq j, \forall u, \begin{cases} (1 - \ell_{i,u}) + \sum_{v>u} (\ell_{i,v} + r_{i,v}) \geq \ell_{j,u} \\ (1 - r_{i,u}) + \sum_{v>u} (\ell_{i,v} + r_{i,v}) \geq r_{j,u} \end{cases}$$

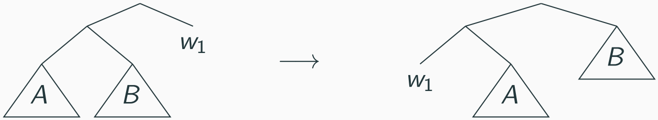
$$\forall i \neq j, \forall u < v, \begin{cases} \ell_{i,u} + (\ell_{i,v} + r_{i,v} + \ell_{j,v} + r_{j,v}) \leq 2 + \ell_{j,u} \\ r_{i,u} + (\ell_{i,v} + r_{i,v} + \ell_{j,v} + r_{j,v}) \leq 2 + r_{j,u} \end{cases}$$

$$\forall i \neq j, \forall u < u', \begin{cases} 2 - \ell_{i,u} - \ell_{j,u} + \sum_{w=u+1}^{u'} (\ell_{i,w} + r_{i,w}) \geq r_{j,u'} + r_{j,u'} \\ 2 - r_{i,u} - r_{j,u} + \sum_{w=u+1}^{u'} (\ell_{i,w} + r_{i,w}) \geq \ell_{j,u'} + \ell_{j,u'} \end{cases}$$

R-Smallest algorithm

The R-SMALLEST algorithm can be used whenever the "rotation property" holds:

$$\forall x \le y \le z, \quad c(x, y) + c(x \circ y, z) \le c(y, z) + c(y \circ z, x)$$



Enumeration of Trees

Enumerate all the leaf-labelled, unordered, full binary trees with n leaves.

Bijection with perfect matchings over $N = 2(n-1)$ vertices [Che90, BDN10]

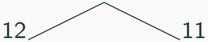
There are $N!! = \mathcal{O}\left(n^{n/2}\right)$ such matchings.

Enumeration of Trees

Enumerate all the leaf-labelled, unordered, full binary trees with n leaves.

Bijection with perfect matchings over $N = 2(n-1)$ vertices [Che90, BDN10]

$\longrightarrow$



There are $N!! = \mathcal{O}\left(n^{n/2}\right)$ such matchings.

Enumeration of Trees

Enumerate all the leaf-labelled, unordered, full binary trees with n leaves.

Bijection with perfect matchings over $N = 2(n-1)$ vertices [Che90, BDN10]

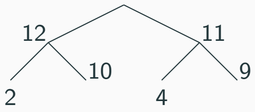
There are $N!! = \mathcal{O}\left(n^{n/2}\right)$ such matchings.

Enumeration of Trees

Enumerate all the leaf-labelled, unordered, full binary trees with n leaves.

Bijection with perfect matchings over $N = 2(n-1)$ vertices [Che90, BDN10]

$\longrightarrow$

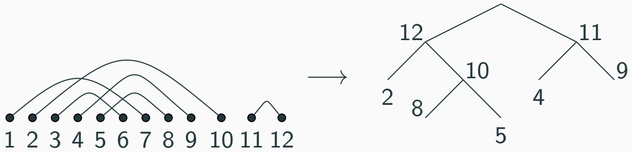


There are $N!! = \mathcal{O}\left(n^{n/2}\right)$ such matchings.

Enumeration of Trees

Enumerate all the leaf-labelled, unordered, full binary trees with n leaves.

Bijection with perfect matchings over $N = 2(n-1)$ vertices [Che90, BDN10]

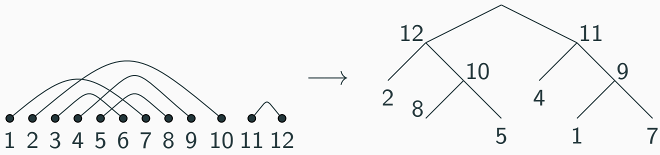


There are $N!! = \mathcal{O}\left(n^{n/2}\right)$ such matchings.

Enumeration of Trees

Enumerate all the leaf-labelled, unordered, full binary trees with n leaves.

Bijection with perfect matchings over $N = 2(n-1)$ vertices [Che90, BDN10]



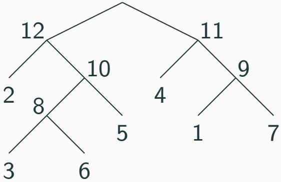
There are $N!! = \mathcal{O}\left(n^{n/2}\right)$ such matchings.

Enumeration of Trees

Enumerate all the leaf-labelled, unordered, full binary trees with n leaves.

Bijection with perfect matchings over $N = 2(n-1)$ vertices [Che90, BDN10]

$\longrightarrow$

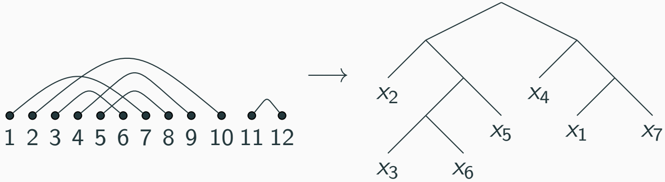


There are $N!! = \mathcal{O}\left(n^{n/2}\right)$ such matchings.

Enumeration of Trees

Enumerate all the leaf-labelled, unordered, full binary trees with n leaves.

Bijection with perfect matchings over $N = 2(n-1)$ vertices [Che90, BDN10]



There are $N!! = \mathcal{O}\left(n^{n/2}\right)$ such matchings.

Algorithms for Abelian Evaluation Trees

	Time	Space
Dynamic Programming	$2^{\mathcal{O}(n)}$	$\mathcal{O}(2^n)$
Integer Programming*	n^2 variables and n^4 constraints	
R-Smallest*	$\mathcal{O}\left(\log(n) n^{\min(C,n)+1}\right)$	$\mathcal{O}(n \log n)$
Enumeration of Trees	$\mathcal{O}\left(n^{n/2}\right)$	$\mathcal{O}(n \log n)$

Conclusion

Open questions

1. Is Balanced Mobiles NP-hard?
2. What are the polynomial-time instances of Abelian Evaluation Trees?
3. For which instances is Smallest optimal?
4. Which instances admit an approximation scheme?

What if the shape of the mobile is fixed and one has just to find the permutation of the weights that minimizes the imbalance?

References i

O. Bernardi, B. Duplantier, and P. Nadeau.
A bijection between well-labelled positive paths and matchings.
In *Séminaire Lotharingien de Combinatoire*, volume 63, page B63e, 2010.

Phillip G. Bradford, Mordecai J. Golin, Lawrence L. Larmore, and Wojciech Rytter.
Optimal prefix-free codes for unequal letter costs: Dynamic programming with the Monge property.
Journal of Algorithms, 42(2):277–303, 2002.

V. Choi and M. J. Golin.
Lopsided trees, I: Analyses.
Algorithmica, 31(3):240–290, 2001.

W. Y. C. Chen.
A general bijective algorithm for trees.
Proc. Nat. Acad. Sci. U.S.A.,, 87(24):9635—9639, 1990.

Mordecai J. Golin, Claire Mathieu, and Neal E. Young.
Huffman coding with letter costs: A linear-time approximation scheme.
SIAM J. Comput., 41(3):684–713, 2012.

M. J. Golin and G. Rote.
A dynamic programming algorithm for constructing optimal prefix-free codes with unequal letter costs.
IEEE Transactions on Information Theory, 44(5):1770–1781, Sep 1998.

Mordecai J. Golin and Neal Young.
Prefix codes: Equiprobable words, unequal letter costs.
SIAM Journal on Computing, 25(6):1281–1292, 1996.

D.A. Huffman.
A method for the construction of minimum-redundancy codes.
Proceedings of the IRE, 40(9):1098–1101, Sept 1952.

Richard M. Karp.
Minimum-redundancy coding for the discrete noiseless channel.
Information Theory, IRE Transactions on, 7(1):27–38, January 1961.

Ben Varn.
Optimal variable length codes (arbitrary symbol cost and equal code word probability).
Information and Control, 19(4):289 – 301, 1971.

Powers-Of-Two Weights

Irregular mobiles

A mobile M is irregular if:

- it is an optimal mobile built on powers-of-two weights
- it cannot be built by $\textsc{Smallest}$
- its imbalance is *less* than the one obtained by $\textsc{Smallest}$ on the same weights.

Irregular mobiles

A mobile M is irregular if:

- it is an optimal mobile built on powers-of-two weights
- it cannot be built by SMALLEST
- its imbalance is *less* than the one obtained by SMALLEST on the same weights.

Proposition

The SMALLEST *algorithm is optimal for powers-of-two weights if and only if there is no irregular mobiles.*

A special irregular mobile

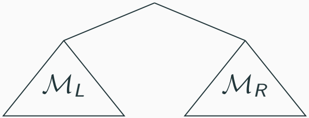
Assume by contradiction that there exist irregular mobiles.

A special irregular mobile

Assume by contradiction that there exist irregular mobiles.

Take such a mobile $\mathcal{M}$ with:

- the smallest maximum weight
- the smallest number of leaves (among the irregular mobiles having the smallest maximum weight)



A special irregular mobile

Note that:

- the maximum weight of $\mathcal{M}$ is at least 2

A special irregular mobile

Note that:

- the maximum weight of $\mathcal{M}$ is at least 2
- $\mathcal{M}$ has at least one leaf of weight 1

A special irregular mobile

Note that:

- the maximum weight of $\mathcal{M}$ is at least 2
- $\mathcal{M}$ has at least one leaf of weight 1
- w.l.o.g. $\mathcal{M}_L$ and $\mathcal{M}_R$ are built by Smallest

A special irregular mobile

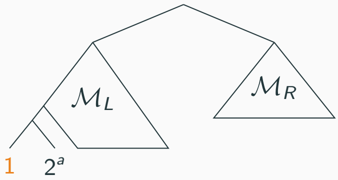
Note that:

- the maximum weight of $\mathcal{M}$ is at least 2
- $\mathcal{M}$ has at least one leaf of weight 1
- w.l.o.g. $\mathcal{M}_L$ and $\mathcal{M}_R$ are built by Smallest
- $\mathcal{M}_L$ and $\mathcal{M}_R$ have at most one leaf of weight 1 each

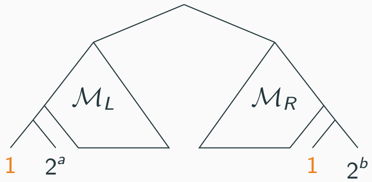
A special irregular mobile

Note that:

- the maximum weight of $\mathcal{M}$ is at least 2
- $\mathcal{M}$ has at least one leaf of weight 1
- w.l.o.g. $\mathcal{M}_L$ and $\mathcal{M}_R$ are built by Smallest
- $\mathcal{M}_L$ and $\mathcal{M}_R$ have at most one leaf of weight 1 each

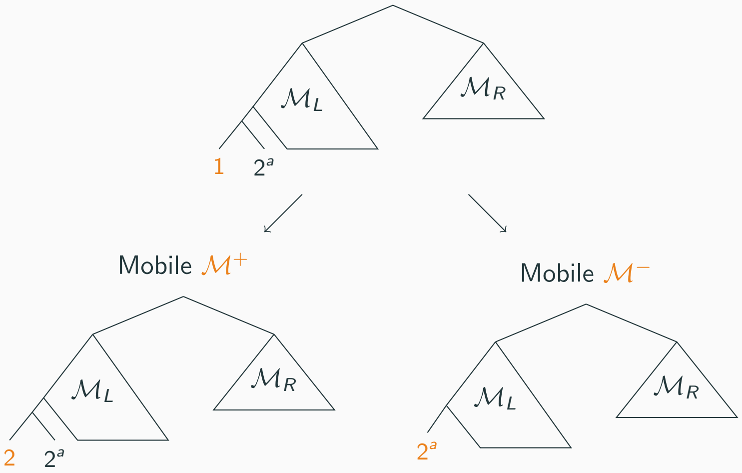


First type



Second type

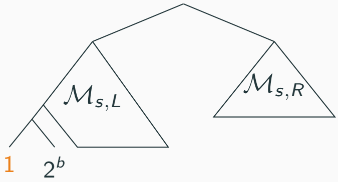
First possible shape



$$\Delta_{\mathcal{M}} = \frac{1}{2}\Delta_{\mathcal{M}^+} + \frac{1}{2}\Delta_{\mathcal{M}^-} + 2^{a-1}$$

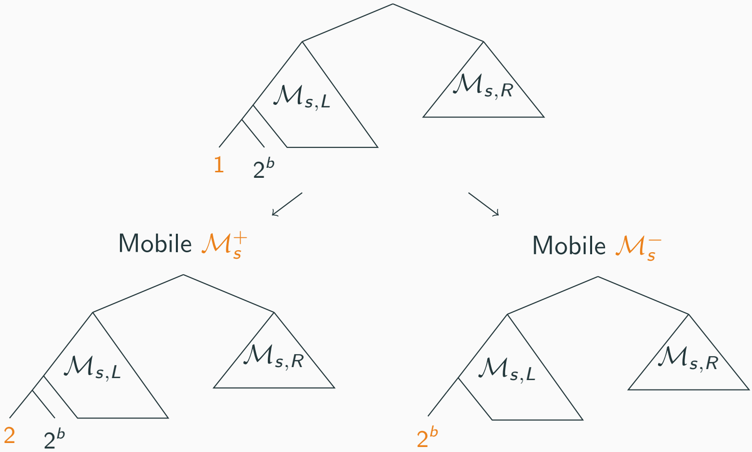
First possible shape

Take the (non-optimal) mobile $\mathcal{M}_s$ built by Smallest:



First possible shape

Take the (non-optimal) mobile $\mathcal{M}_s$ built by SMALLEST:



$$\Delta_{\mathcal{M}_s} = \tfrac{1}{2}\Delta_{\mathcal{M}_s^+} + \tfrac{1}{2}\Delta_{\mathcal{M}_s^-} + 2^{b-1}$$

First possible shape

Note that:

- $\mathcal{M}_s^+$ and $\mathcal{M}_s^-$ are also built by $\textsc{Smallest}$
- $\mathcal{M}^+$ and $\mathcal{M}^-$ cannot be irregular
- $2^b \leq 2^a$

First possible shape

Note that:

- $\mathcal{M}_s^+$ and $\mathcal{M}_s^-$ are also built by $\textsc{Smallest}$
- $\mathcal{M}^+$ and $\mathcal{M}^-$ cannot be irregular
- $2^b \leq 2^a$

Consequently: $\Delta_{\mathcal{M}_s^+} \leq \Delta_{\mathcal{M}^+}$ and $\Delta_{\mathcal{M}_s^-} \leq \Delta_{\mathcal{M}^-}$

First possible shape

Note that:

- $\mathcal{M}_s^+$ and $\mathcal{M}_s^-$ are also built by $\textsc{Smallest}$
- $\mathcal{M}^+$ and $\mathcal{M}^-$ cannot be irregular
- $2^b \leq 2^a$

Consequently: $\Delta_{\mathcal{M}_s^+} \leq \Delta_{\mathcal{M}^+}$ and $\Delta_{\mathcal{M}_s^-} \leq \Delta_{\mathcal{M}^-}$

Thus :

$$\Delta_{\mathcal{M}_s} = \frac{1}{2}\Delta_{\mathcal{M}_s^+} + \frac{1}{2}\Delta_{\mathcal{M}_s^-} + 2^{b-1} \leq \frac{1}{2}\Delta_{\mathcal{M}^+} + \frac{1}{2}\Delta_{\mathcal{M}^-} + 2^{a-1} = \Delta_{\mathcal{M}}$$

It contradicts $\Delta_{\mathcal{M}_s} > \Delta_{\mathcal{M}}$.