Combinatorial entropy
and succinct data structures

Gilles Schaeffer

based in part on joined works with
L. CastelliAleardi, O. Devillers,
E. Fusy and D. Poulalhon

Analysis of Algorithms, 2009
Before we start... Geometric data; meshes

Among data structures for geometric data, I pick meshes...
Before we start… ∃ very large geometric data

St. Matthew (Stanford’s Digital Michelangelo Project, 2000)
186 millions vertices
6 Giga bytes (for storing on disk)
minutes for loading the model from disk

David statue (Stanford’s Digital Michelangelo Project, 2000)
2 billions polygons
32 Giga bytes (without compression)

No existing algorithm nor data structure for dealing with the entire model
Before we start... What we are aiming at

Mesh compression

Geometric data structures

Transmission

disk storage
Before we start... What we are aiming at

Mesh compression

Geometric data structures

Transmission
disk storage

MERGE INTO: Compact representations of geometric data structures
Starter: the encoding of plane trees

ordered tree with $n$ edges

balanced parenthesis word of length $2n$
Starter: the encoding of plane trees

ordered tree with \( n \) edges

balanced parenthesis word of length \( 2n \)

\[
1 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0
\]

\( \Rightarrow \) \( 2n \) bits for encoding an ordered tree with \( n \) edges
Starter: the encoding of plane trees

ordered tree with \( n \) edges

balanced parenthesis word of length \( 2n \)

\[
1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0
\]

\( \Rightarrow 2n \) bits for encoding an ordered tree with \( n \) edges

Compare to the standard explicit representation:

\( 3n \) pointers \( \approx 96 \) bits

\( 3n \log n \) in theory
ordered tree with $n$ edges

balanced parenthesis word of length $2n$

⇒ $2n$ bits for encoding an ordered tree with $n$ edges

enumeration: $\|B_n\| = \frac{1}{n+1} \binom{2n}{n} \approx 2^{2n} n^{-\frac{3}{2}}$
Starter: the encoding of plane trees

ordered tree with \( n \) edges

balanced parenthesis word of length \( 2n \)

\[ \Rightarrow 2n \text{ bits for encoding an ordered tree with } n \text{ edges} \]

enumeration: \[ \| \mathcal{B}_n \| = \frac{1}{n+1} \binom{2n}{n} \approx 2^{2n} n^{-\frac{3}{2}} \]

\[ \log_2 \| \mathcal{B}_n \| = 2n + O(\lg n) \text{ bpv} \]
Starter: the encoding of plane trees

ordered tree with $n$ edges

balanced parenthesis word of length $2n$

$\Rightarrow 2n$ bits for encoding an ordered tree with $n$ edges

enumeration: $\|B_n\| = \frac{1}{n+1} \binom{2n}{n} \approx 2^{2n} n^{-\frac{3}{2}}$

$log_2 \|B_n\| = 2n + O(\lg n)$ bpv

This is an optimal encoding!

it matches asymptotically the information-theory lower bound
Starter: the encoding of plane trees

ordered tree with $n$ edges

balanced parenthesis word of length $2n$

$\Rightarrow 2n$ bits for encoding an ordered tree with $n$ edges

enumeration: $\|B_n\| = \frac{1}{n+1} \binom{2n}{n} \approx 2^{2n} n^{-\frac{3}{2}}$

$\log_2 \|B_n\| = 2n + O(\lg n)$ bpv

This is an optimal encoding!

it matches asymptotically the information-theory lower bound
Starter: the encoding of plane trees

ordered tree with $n$ edges

balanced parenthesis word of length $2n$

$\Rightarrow 2n$ bits for encoding an ordered tree with $n$ edges

enumeration: $\|B_n\| = \frac{1}{n+1} \binom{2n}{n} \approx 2^{2n} n^{-\frac{3}{2}}$

$\log_2 \|B_n\| = 2n + O(\lg n)$ bpv

This is an optimal encoding!

it matches asymptotically the information-theory lower bound
ordered tree with $n$ edges

balanced parenthesis word of length $2n$

Navigation in the tree: handlers
ordered tree with $n$ edges

balanced parenthesis word of length $2n$

Navigation in the tree: handlers
ordered tree with $n$ edges

balanced parenthesis word of length $2n$

Navigation in the tree: handlers
ordered tree with \( n \) edges

balanced parenthesis word of length \( 2n \)

Navigation in the tree: handlers

- move the handler to first son
- move the handler to next brother
- move the handler to father
ordered tree with $n$ edges

balanced parenthesis word of length $2n$

Navigation in the tree: handlers
move the handler to first son
move the handler to next brother
move the handler to father

Constant time with standard (pointer) representation
but the pointer based representation uses $\Theta(n \log n)$ bits
Starter: linear space data structures for plane trees?

ordered tree with $n$ edges

balanced parenthesis word of length $2n$

Navigation in the tree: handlers

- move the handler to first son
- move the handler to next brother
- move the handler to father

Constant time with standard (pointer) representation

but the pointer based representation uses $\Theta(n \log n)$ bits
Starter: linear space data structures for plane trees?

ordered tree with $n$ edges
balanced parenthesis word of length $2n$

Navigation in the tree: handlers

- move the handler to first son
- move the handler to next brother
- move the handler to father

Constant time with standard (pointer) representation
but the pointer based representation uses $\Theta(n \log n)$ bits
Starter: linear space data structures for plane trees?

ordered tree with \( n \) edges

balanced parenthesis word of length \( 2n \)

Navigation in the tree: handlers

move the handler to first son

move the handler to father

move the handler to next brother

handler = index of opening bracket

index \( \rightarrow \) index +1

Constant time with standard (pointer) representation

but the pointer based representation uses \( \Theta(n \log n) \) bits
Starter: linear space data structures for plane trees?

ordered tree with \( n \) edges

balanced parenthesis word of length \( 2n \)

Navigation in the tree: handlers

- move the handler to first son \( \text{index} \rightarrow \text{index} + 1 \)
- move the handler to next brother \( \text{index} \rightarrow \text{matching(index)} + 1 \)
- move the handler to father

Constant time with standard (pointer) representation but the pointer based representation uses \( \Theta(n \log n) \) bits
Starter: linear space data structures for plane trees?

ordered tree with \( n \) edges

balanced parenthesis word of length \( 2n \)

Navigation in the tree: handlers

- move the handler to first son: \( \text{index} \rightarrow \text{index} + 1 \)
- move the handler to next brother: \( \text{index} \rightarrow \text{matching(index)} + 1 \)
- move the handler to father: \( \text{index} \rightarrow \text{outer(index)} \)

Constant time with standard (pointer) representation

but the pointer based representation uses \( \Theta(n \log n) \) bits
ordered tree with $n$ edges

balanced parenthesis word of length $2n$

Navigation in the tree: handlers

- move the handler to first son: $\text{index} \rightarrow \text{index} + 1$
- move the handler to next brother: $\text{index} \rightarrow \text{matching(index)} + 1$
- move the handler to father: $\text{index} \rightarrow \text{outer(index)}$

Constant time with standard (pointer) representation up to linear time!

but the pointer based representation uses $\Theta(n \log n)$ bits
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into $m$ small blocks of size $\varepsilon$

\[
\begin{array}{ccccc}
  b_1 & b_2 & b_3 & b_4 & b_5 \\
  ((( ( ) ) ) ) & ( ( ) ) ( ) ( ) & ( ( ) ) ( ) ( ) & 2n \text{ bits}
\end{array}
\]
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into $m$ small blocks of size $\varepsilon$

\[
\begin{array}{ccccc}
  b_1 & b_2 & b_3 & b_4 & b_5 \\
  ((((()( ()))(((()( ()))))) 2n \text{ bits}
\end{array}
\]

**matching(index):** go slowly inside block
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into $m$ small blocks of size $\varepsilon$

$\begin{array}{ccccc}
  b_1 & b_2 & b_3 & b_4 & b_5 \\
  \boxed{(((()))((())(\quad((())\quad((())\quad())\quad())\quad)})} & 2n \text{ bits}
\end{array}$

**matching(index):** go slowly inside block  
if border reached: interblock
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into $m$ small blocks of size $\varepsilon$

```
 b_1  b_2  b_3  b_4  b_5

(((()  ))  ((  ))  ((  ))
```

$2n$ bits

**matching(index):** go slowly inside block  
if border reached: interblock
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into $m$ small blocks of size $\varepsilon$

$\begin{array}{c}
b_1 \\
| \\
( ) \\
| \\
b_2 \\
| \\
( ) \\
| \\
b_3 \\
| \\
( ) \\
| \\
b_4 \\
| \\
( ) \\
| \\
b_5 \\
\end{array}$

$2n$ bits

**matching(index):** go slowly inside block if border reached: interblock encode interblock explicitly: up to $n$ edges $\Rightarrow$ space $n \log n$
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into $m$ small blocks of size $\varepsilon$

```
b_1   b_2   b_3   b_4   b_5
( ( ( ( ) ) ) ( ) ( ( ( ) ( ) ) ( ) ) )
```

$2n$ bits

**matching(index):** go slowly inside block  
if border reached: interblock
encode interblock explicitly: up to $n$ edges $\Rightarrow$ space $n \log n$

encode $\leq m-1$ **pioneers** (outermost between blocks) $\Rightarrow$ space $m \log n$
Starter: linear space data structures for plane trees
(Jacobson, FoCS89)

Decompose into \( m \) small blocks of size \( \varepsilon \)

\[
\begin{array}{cccc}
b_1 & b_2 & b_3 & b_4 & b_5 \\
(((()|())()|())|())|())|())|())|()|()
\end{array}
\]

2n bits

\((1, 22)(2, 9)(3, 6)(10, 19)(15, 16)(20, 21)\)

**matching(index):** go slowly inside block \( b \) if border reached: interblock encode interblock explicitly: up to \( n \) edges \( \Rightarrow \) space \( n \log n \) encode \( \leq m-1 \) pioneers (outermost between blocks) \( \Rightarrow \) space \( m \log n \)
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into $m$ small blocks of size $\varepsilon$

$$\begin{array}{cccc}
b_1 & b_2 & b_3 & b_4 & b_5 \\
((()()) & (()) & (()) & (()) & ()
\end{array}$$  

$2n$ bits

$$(1, 22)(2, 9)(3, 6)(10, 19)(15, 16)(20, 21)$$

**matching(index):** go slowly inside block  
if border reached: interblock encode interblock explicitly: up to $n$ edges $\Rightarrow$ space $n \log n$
encode $\leq m-1$ pioneers (outermost between blocks) $\Rightarrow$ space $m \log n$
the explicit representation must allow navigation...
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into $m$ small blocks of size $\varepsilon$

```
b_1  b_2  b_3  b_4  b_5
(((()  )())(  (()())(  )))
```

2n bits

```
B  1100000001000010000100
```

$m \log n$ bits

```
T  22  9  19  16  21
```

matching(3): 3, 4, 5, interblock, $r_B(3) = 2$, $T(2) = 9, 9, 8, 7, 6$.

matching(index): go slowly inside block if border reached: interblock
encode interblock explicitely: up to $n$ edges $\Rightarrow$ space $n \log n$
encode $\leq m-1$ pioneers (outermost between blocks) $\Rightarrow$ space $m \log n$
the explicit representation must allow navigation...
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into $m$ small blocks of size $\varepsilon$

<table>
<thead>
<tr>
<th>$b_1$</th>
<th>$b_2$</th>
<th>$b_3$</th>
<th>$b_4$</th>
<th>$b_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(((()))</td>
<td>(())</td>
<td>(())</td>
<td>(())</td>
<td>(())</td>
</tr>
</tbody>
</table>

$B$: 110000000100010000100

$m \log n$ bits

$T$: 22 9 19 16 21

$2n$ bits

matching(3): 3,4,5, interblock, $r_B(3) = 2$, $T(2) = 9$, 9,8,7,6.

**matching(index):** go slowly inside block  if border reached: interblock
encode interblock explicitely: up to $n$ edges $\Rightarrow$ space $n \log n$
encode $\leq m-1$ pioneers (outermost between blocks) $\Rightarrow$ space $m \log n$
the explicit representation must allow navigation...
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into \( m \) small blocks of size \( \varepsilon \)

\[
\begin{array}{cccccc}
\text{\( b_1 \)} & \text{\( b_2 \)} & \text{\( b_3 \)} & \text{\( b_4 \)} & \text{\( b_5 \)} \\
( ( ( ( ) ( )) ) & ( ( ) ( ) ( ) ) & ( ( ) ) ( ) ( ) ( ) & ( ) ( ) & ( ) ( )
\end{array}
\]

2n bits

\[
\begin{array}{cccccccc}
\text{\( B \)} & \text{11000} & \text{00001} & \text{00001} & \text{00001} & \text{100}
\end{array}
\]

low weight bit vectors

select/rank queries

\[
\begin{array}{cccccc}
\text{\( T \)} & \text{22} & \text{9} & \text{19} & \text{16} & \text{21}
\end{array}
\]

\( m \log n \) bits

O(n) extra bits

matching(3): 3, 4, 5, interblock, \( r_B(3) = 2 \), \( T(2) = 9, 9, 8, 7, 6 \).

matching(index): go slowly inside block if border reached: interblock encode interblock explicitly: up to \( n \) edges \( \Rightarrow \) space \( n \log n \) encode \( \leq m-1 \) pioneers (outermost between blocks) \( \Rightarrow \) space \( m \log n \)

Taking \( \varepsilon = \Theta(\log n) \): space \( m \log n = O(n) \), queries in \( O(\log n) \)
Starter: linear space data structures for plane trees
(Jacobson, Focs89)

Decompose into $m$ small blocks of size $\varepsilon$

$T$

$B$

$22 \ 9 \ 19 \ 16 \ 21$

$11\ 0000\ 0001\ 00001\ 00001\ 00$

matching(3): 3, 4, 5, interblock, $r_B(3) = 2, T(2) = 9, 9, 8, 7, 6$.

matching(index): go slowly inside block if border reached: interblock
encode interblock explicitly: up to $n$ edges $\Rightarrow$ space $n \log n$
encode $\leq m-1$ pioneers (outermost between blocks) $\Rightarrow$ space $m \log n$

Taking $\varepsilon = \Theta(\log n)$: space $m \log n = O(n)$, queries in $O(\log n)$

succinct data structures: want space $2n + o(n)$ and queries in $O(1)$
Combinatorial entropy and succinct data structures

\( \mathcal{A}_n \): structures of size \( n \), with \( \log_2 |\mathcal{A}_n| = \alpha n + O(n) \).
but large explicit representation (using \( O(n) \) pointers of size \( \log n \))

**Aim 1 (compression):** find an encoding with \( \alpha \) bits per size unit
with linear time encoding/decoding procedures
Combinatorial entropy and succinct data structures

$A_n$: structures of size $n$, with $\log_2 |A_n| = \alpha n + O(n)$.

but large explicit representation (using $O(n)$ pointers of size $\log n$)

**Aim 1 (compression):** find an encoding with $\alpha$ bits per size unit with linear time encoding/decoding procedures

**Aim 2 (succinct data struct):** idem + efficient query support
answer natural queries in constant time ($\log$-time if not constant)
Combinatorial entropy and succinct data structures

$A_n$: structures of size $n$, with $\log_2 |A_n| = \alpha n + O(n)$.
but large explicit representation (using $O(n)$ pointers of size $\log n$)

**Aim 1 (compression):** find an encoding with $\alpha$ bits per size unit
with linear time encoding/decoding procedures

**Aim 2 (succinct data struc):** idem + efficient query support
answer natural queries in constant time ($\log$ time if not constant)

**Aim 3 (dynamical s.d.s.):** idem + update of the structure
update the structure in $\log$ time (amortized if not worst case)
Combinatorial entropy and succinct data structures

$A_n$: structures of size $n$, with $\log_2 |A_n| = \alpha n + O(n)$.
but large explicit representation (using $O(n)$ pointers of size $\log n$)

**Aim 1 (compression):** find an encoding with $\alpha$ bits per size unit
with linear time encoding/decoding procedures

**Aim 2 (succinct data struc):** idem + efficient query support
answer natural queries in constant time ($\log$-time if not constant)

**Aim 3 (dynamical s.d.s.):** idem + update of the structure
update the structure in $\log$-time (amortized if not worst case)

**Aim 0:** understand and deal with entropy reduction...
Entropy reduction and parametrized classes

ordered trees with $n$ vertices

entropy 2bpv
Entropy reduction and parametrized classes

ordered trees with $n$ vertices

degree 2 and 0 only: complete binary trees

$(2n + 1$ vertices: $n$ nodes, $n + 1$ leaves)
Entropy reduction and parametrized classes

ordered trees with $n$ vertices

degree 2 and 0 only: complete binary trees
$(2n + 1$ vertices: $n$ nodes, $n + 1$ leaves)$

degree 3 and 0 only: complete ternary
$(3n + 1$ vertices: $n$ nodes, $2n + 1$ leaves)$

entropy 2bpv

$\frac{1}{3} \log_2 \frac{27}{2} \approx 1.25 \text{ bpv}$
Entropy reduction and parametrized classes

ordered trees with \( n \) vertices

- degree 2 and 0 only: complete binary trees
  \((2n + 1 \text{ vertices: } n \text{ nodes, } n + 1 \text{ leaves})\)

- degree 3 and 0 only: complete ternary
  \((3n + 1 \text{ vertices: } n \text{ nodes, } 2n + 1 \text{ leaves})\)

more generally, \( n_i \) vertices of degree \( i \)

entropy

- 2bpv

- 1bpv

- \( \frac{1}{3} \log_2 \frac{27}{2} \approx 1.25 \text{ bpv} \)
Entropy reduction and parametrized classes

ordered trees with \( n \) vertices

- degree 2 and 0 only: complete binary trees
  \((2n + 1 \text{ vertices: } n \text{ nodes}, n + 1 \text{ leaves})\)
  \(1\text{bpv}\)

- degree 3 and 0 only: complete ternary
  \((3n + 1 \text{ vertices: } n \text{ nodes}, 2n + 1 \text{ leaves})\)
  \(\frac{1}{3} \log_2 \frac{27}{2} \approx 1.25 \text{ bpv}\)

more generally, \( n_i \) vertices of degree \( i \)

**Old Thm:** \(|\mathcal{T}(n_0, \ldots, n_k)| = \frac{1}{n} \left(\begin{array}{c} n \\ n_0, n_1, \ldots, n_k \end{array}\right)\)
Entropy reduction and parametrized classes

ordered trees with $n$ vertices

degree 2 and 0 only: complete binary trees
$(2n + 1 \text{ vertices: } n \text{ nodes, } n + 1 \text{ leaves})$

degree 3 and 0 only: complete ternary
$(3n + 1 \text{ vertices: } n \text{ nodes, } 2n + 1 \text{ leaves})$

more generally, $n_i$ vertices of degree $i$

**Old Thm:** $|\mathcal{T}(n_0, \ldots, n_k)| = \frac{1}{n} \binom{n}{n_0, n_1, \ldots, n_k}$
if $n = \sum n_i = 1 + \sum in_i$

entropy $2 \text{ bpv}$

$1 \text{ bpv}$

$\frac{1}{3} \log_2 \frac{27}{2} \approx 1.25 \text{ bpv}$

$\log_2 \left( \frac{n}{n_0, n_1, \ldots, n_k} \right) \frac{1}{n}$

$\log_2 \prod_i \alpha_i^{n_i} \alpha_i^{-\alpha_i}$
if $n_i = \alpha_i n$
Entropy reduction and parametrized classes

ordered trees with \( n \) vertices

degree 2 and 0 only: complete binary trees
(\( 2n + 1 \) vertices: \( n \) nodes, \( n + 1 \) leaves)

degree 3 and 0 only: complete ternary
(\( 3n + 1 \) vertices: \( n \) nodes, \( 2n + 1 \) leaves)

more generally, \( n_i \) vertices of degree \( i \)

**Old Thm:** \(|T(n_0, \ldots, n_k)| = \frac{1}{n} \binom{n}{n_0, n_1, \ldots, n_k} \)

if \( n = \sum n_i = 1 + \sum i n_i \)

encode tree by degree list in prefix order

observe that: \( \text{entropy(trees)} = \text{entropy of text} \)

compress optimally with arithmetic coder

\[
\text{entropy} \ 2\text{bpv} \\
\text{1bpv} \\
\frac{1}{3} \log_2 \frac{27}{2} \approx 1.25 \text{ bpv} \\
\log_2 \left( \binom{n}{n_0, n_1, \ldots, n_k} \right) \frac{1}{n} \\
\log_2 \prod_i \alpha_i^{-\alpha_i} \text{ if } n_i = \alpha_i n
\]
Entropy reduction and parametrized classes

ordered trees with \( n \) vertices

degree 2 and 0 only: complete binary trees
\((2n + 1 \text{ vertices: } n \text{ nodes, } n + 1 \text{ leaves})\)

degree 3 and 0 only: complete ternary
\((3n + 1 \text{ vertices: } n \text{ nodes, } 2n + 1 \text{ leaves})\)

more generally, \( n_i \) vertices of degree \( i \)

**Old Thm:** \(|\mathcal{T}(n_0, \ldots, n_k)| = \frac{1}{n} \binom{n}{n_0,n_1,\ldots,n_k} \)
\(\text{if } n = \sum n_i = 1 + \sum i n_i \)

encode tree by degree list in prefix order

observe that: \( \text{entropy(trees)} = \text{entropy of text} \)
compress optimally with arithmetic coder

**Question:** what is the maximum entropy, for which degrees?
## Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
## Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

given degree distribution
### Entropy quizz

<table>
<thead>
<tr>
<th>Ordered trees</th>
<th>Entropy</th>
<th>Compression</th>
<th>Succinct d.s.</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

Given degree distribution:

\[ \sum \alpha_i \log_2 \frac{1}{\alpha_i} \]
## Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

given degree distribution \[ \sum \alpha_i \log_2 \frac{1}{\alpha_i} \] yes
## Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

given degree distribution \[ \sum \alpha_i \log_2 \frac{1}{\alpha_i} \] yes \((soda'07)\) ?
## Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

| given degree distribution         | $\sum \alpha_i \log_2 \frac{1}{\alpha_i}$ | yes         | yes           | ?       |

| bipartite:                        | $p$ black, $q$ white                        |             | (soda’07)     |         |
### Entropy quizz

<table>
<thead>
<tr>
<th></th>
<th>Entropy</th>
<th>Compression</th>
<th>Succinct d.s.</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>ordered trees</td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>given degree distribution</strong></td>
<td>$\sum \alpha_i \log_2 \frac{1}{\alpha_i}$</td>
<td>yes</td>
<td>yes (soda'07)</td>
<td>?</td>
</tr>
<tr>
<td><strong>bipartite:</strong> $p$ black, $q$ white</td>
<td>4 if $p = \frac{n}{2} + O(\sqrt{n})$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

| given degree distribution | $\sum \alpha_i \log_2 1/\alpha_i$ | yes | yes \(^{(soda'07)}\) | ? |

| bipartite: $p$ black, $q$ white | 4 if $p = \frac{n}{2} + O(\sqrt{n})$ | use basic result |
### Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

| given degree distribution | \[ \sum \alpha_i \log_2 \frac{1}{\alpha_i} \] yes | yes | ? |

<table>
<thead>
<tr>
<th>bipartite: p black, q white</th>
<th>4 if ( p = \frac{n}{2} + O(\sqrt{n}) ) use basic result</th>
</tr>
</thead>
<tbody>
<tr>
<td>otherwise ( \binom{p+q}{p} \frac{2}{n} )</td>
<td></td>
</tr>
<tr>
<td>Ordered Trees</td>
<td>Entropy</td>
</tr>
<tr>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td>Given Degree Distribution</td>
<td>$\sum \alpha_i \log_2 \frac{1}{\alpha_i}$</td>
</tr>
<tr>
<td>Bipartite: $p$ black, $q$ white</td>
<td>$4$ if $p = \frac{n}{2} + O(\sqrt{n})$</td>
</tr>
</tbody>
</table>
## Entropy quizz

<table>
<thead>
<tr>
<th>Ordered trees</th>
<th>Entropy</th>
<th>Compression</th>
<th>Succinct d.s.</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>given degree distribution</td>
<td>$\sum \alpha_i \log_2 \frac{1}{\alpha_i}$</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>bipartite: $p$ black, $q$ white</td>
<td>4 if $p = \frac{n}{2} + O(\sqrt{n})$</td>
<td>use basic result</td>
<td></td>
<td>?</td>
</tr>
<tr>
<td></td>
<td>otherwise $\binom{p+q}{p} \frac{2}{n}$</td>
<td>yes</td>
<td>probably</td>
<td>?</td>
</tr>
<tr>
<td>height $h$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Entropy quizz

<table>
<thead>
<tr>
<th>Ordered Trees</th>
<th>Entropy</th>
<th>Compression</th>
<th>Succinct D.S.</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Given Degree Distribution</td>
<td>$\sum \alpha_i \log_2 \frac{1}{\alpha_i}$</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Bipartite: $p$ black, $q$ white</td>
<td>$4$ if $p = \frac{n}{2} + O(\sqrt{n})$</td>
<td>use basic result</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Otherwise $\left(\frac{p+q}{n}\right)^2$</td>
<td>yes</td>
<td>probably</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Height $h$</td>
<td>known</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>given degree distribution</td>
<td>$\sum \alpha_i \log_2 1/\alpha_i$</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>bipartite: $p$ black, $q$ white</td>
<td>$4$ if $p = \frac{n}{2} + O(\sqrt{n})$</td>
<td>use basic result</td>
<td>probably</td>
<td>?</td>
</tr>
<tr>
<td>height $h$</td>
<td>known</td>
<td>?</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

positive natural embedding
### Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>given degree distribution</th>
<th>$\sum \alpha_i \log_2 1/\alpha_i$</th>
<th>yes</th>
<th>yes</th>
<th>?</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>bipartite: $p$ black, $q$ white</th>
<th>$4$ if $p = \frac{n}{2} + O(\sqrt{n})$</th>
<th>use basic result</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>otherwise $\left(\frac{p+q}{p}\right) \frac{2}{n}$</td>
<td>yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>height $h$</th>
<th>known</th>
<th>?</th>
</tr>
</thead>
</table>

| positive natural embedding | 4 | use basic result |
## Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

| given degree distribution | $\sum \alpha_i \log_2 \frac{1}{\alpha_i}$ | yes | yes | ? |

| bipartite: $p$ black, $q$ white | $4$ if $p = \frac{n}{2} + O(\sqrt{n})$ | use basic result |
|                                 | otherwise $\left(\frac{p+q}{p}\right)^{\frac{2}{n}}$ | yes | probably | ? |

| height $h$ | known | ? |

| positive natural embedding | 4 | use basic result |

| all leaves at same depth | | |

(soda’07)
## Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

| given degree distribution | \[ \sum \alpha_i \log_2 \frac{1}{\alpha_i} \] yes | yes | (soda'07) |

| bipartite: \( p \) black, \( q \) white | 4 if \( p = \frac{n}{2} + O(\sqrt{n}) \) use basic result | otherwise \( \binom{p+q}{p} \frac{2}{n} \) yes | probably |

| height \( h \) | known | ? |

| positive natural embedding | 4 | use basic result |

| all leaves at same depth | known? | ? |
### Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

| given degree distribution | \[
\sum \alpha_i \log_2 \frac{1}{\alpha_i} \]
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>yes</td>
</tr>
<tr>
<td></td>
<td>yes</td>
</tr>
</tbody>
</table>

\[
\text{bipartite: } \begin{cases}
4 \text{ if } p = \frac{n}{2} + O(\sqrt{n}) \\
\text{otherwise } \left(\frac{p+q}{p}\right) \frac{2}{n}
\end{cases}
\]

<table>
<thead>
<tr>
<th>height ( h )</th>
<th>known</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>positive natural embedding</th>
<th>4</th>
<th>use basic result</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>all leaves at same depth</th>
<th>known?</th>
</tr>
</thead>
</table>

### ordinary decomposable structures

(multitype ordered trees)
# Entropy quizz

<table>
<thead>
<tr>
<th>ordered trees</th>
<th>entropy</th>
<th>compression</th>
<th>succinct d.s.</th>
<th>dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

| given degree distribution | \[ \sum \alpha_i \log_2 \frac{1}{\alpha_i} \] yes | yes | ? |

| bipartite: p black, q white | 4 if \( p = \frac{n}{2} + O(\sqrt{n}) \) | use basic result |
|                            | otherwise \( \left( \frac{p+q}{p} \right)^{\frac{2}{n}} \) | yes | probably | ? |

| height \( h \) | known | ? |

| positive natural embedding | 4 | use basic result |

| all leaves at same depth | known? | ? |

| ordinary decomposable structures (multitype ordered trees) | computable | ? use frequencies | ? |

link with multivariable Lagrange inversion?
## Entropy quiz

<table>
<thead>
<tr>
<th>Ordered Trees</th>
<th>Entropy</th>
<th>Compression</th>
<th>Succinct D.S.</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Given Degree Distribution</td>
<td>$\sum \alpha_i \log_2 \frac{1}{\alpha_i}$</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Bipartite: $p$ black, $q$ white</td>
<td>$4$ if $p = \frac{n}{2} + O(\sqrt{n})$</td>
<td>use basic result</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Otherwise</td>
<td>$\binom{p+q}{p} \frac{2}{n}$</td>
<td>yes</td>
<td>probably</td>
<td></td>
</tr>
<tr>
<td>Height $h$</td>
<td>known</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Positive Natural Embedding</td>
<td>$4$</td>
<td>use basic result</td>
<td></td>
<td></td>
</tr>
<tr>
<td>All Leaves at Same Depth</td>
<td>known?</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Ordinary Decomposable Structures**

*(Multitype Ordered Trees)*

- Computable
- Use Frequencies?
- Link with Multivariable Lagrange Inversion?

**Entropy Measures Diversity of Local Structure**
**Geometry information vs Combinatorial information**

**Geometry**
- 30 to 96 bits/vertex
- Vertex coordinates

"Connectivity": the underlying triangulation
- 1 reference to a triangle
- 3 references to vertices
- 3 references to triangles
- $13n \log n$ or $416n$ bits

Adjacency relations between triangles, vertices
**Geometric information vs Combinatorial information**

**Geometry**

- vertex coordinates
- between 30 et 96 bits/vertex

**”Connectivity”: the underlying triangulation**

- vertex: 1 reference to a triangle
- triangle: 3 references to vertices, 3 references to triangles

\[ \#\{\text{triangulations}\} = \frac{2(4n + 1)!}{(3n + 2)!(n + 1)!} \approx \frac{16}{27} \sqrt{\frac{3}{2\pi}} n^{-5/2} \left(\frac{256}{27}\right)^n \]

or

\[ 13n \log n \]

or

\[ 416n \text{ bits} \]
**Geometric information vs Combinatorial information**

**Geometry**
- vertex coordinates
- between 30 et 96 bits/vertex

**"Connectivity"**: the underlying triangulation
- vertex
  - 1 reference to a triangle
- triangle
  - 3 references to vertices
  - 3 references to triangles

\[
\#\{\text{triangulations}\} = \frac{2(4n + 1)!}{(3n + 2)!(n + 1)!} \approx \frac{16}{27} \sqrt{\frac{3}{2\pi}} n^{-5/2} \left(\frac{256}{27}\right)^n
\]

\[\Rightarrow \text{entropy} = \log_2 \frac{256}{27} \approx 3.24 \text{ bpv.}\]
**Geometric information vs Combinatorial information**

**Geometry**

- vertex coordinates
- between 30 et 96 bits/vertex

**"Connectivity": the underlying triangulation**

- adjacency relations between triangles, vertices

- vertex: 1 reference to a triangle
- triangle: 3 references to vertices, 3 references to triangles

\[
\#\{\text{triangulations}\} = \frac{2(4n + 1)!}{(3n + 2)!(n + 1)!} \approx \frac{16}{27} \sqrt{\frac{3}{2\pi}} n^{-5/2} \left(\frac{256}{27}\right)^n
\]

\[
\Rightarrow \text{entropy} = \log_2 \frac{256}{27} \approx 3.24 \text{ bpv.} \quad \text{Room for improvement!}
\]
Triangulation encodings: trees decompositions

Common visual framework (Isenburg Snoeyink’05)

Edgebreaker, Rosignac (’99) - 3.67n

Canonical orderings, Chiang at al. (’98) - 4n

Degree encoding, Touma-Gotsman (’98) - but efficient

Leftmost tree in minimal canonical ordering, Poulalhon, S. (’03) - 3.24n

V5V5V6V5V4V5V8V5V5V4S4V3V4

1101000110000010010000011001000000000
Triangulation encodings: trees decompositions

Common visual framework (Isenburg Snoeyink’05)

Edgebreaker, Rosignac ('99)

Canonical orderings, Chiang at al. ('98)

Degree encoding, Touma-Gotsman ('98)

Leftmost tree in minimal canonical ordering Poulalhon, S. ('03)

"optimal"

3.67n

4n

? but efficient

3.24n

$V_5 V_5 V_6 V_5 V_4 V_5 V_8 V_5 V_5 V_4 S_4 V_3 V_4$

1101000110000010010000011001000000000
Triangulation encodings: trees decompositions

Common visual framework (Isenburg Snoeyink’05)

Edgebreaker, Rosignac ('99)

Canonical orderings, Chiang at al. ('98)

Degree encoding, Touma-Gotsman ('98)

Leftmost tree in minimal canonical ordering Poulalhon, S. ('03)

```
( [ [ ] ] [ ] [ ] [ ] [ ] [ ] ) . . .
```

```
1101000110000010010000011001000000000
```

3.67n

4n

3.24n

"optimal"
Triangulation encodings: trees decompositions
Common visual framework (Isenburg Snoeyink’05)
The (non-optimal) degree encoder gives much better codes for low entropy triangulations!
Patch of triangular grids \(\Rightarrow 6,6,6,6,6,6,5,6,6,6,6,6,7\ldots\)

Alliez Desbrun (Eurographics ’01): could a degree encoder be optimal?

The (non-optimal) degree encoder gives much better codes for low entropy triangulations!

Patch of triangular grids \(\Rightarrow 6,6,6,6,6,6,5,6,6,6,6,6,6,7\ldots\)

Alliez Desbrun (Eurographics ’01): could a degree encoder be optimal?
Triangulation encodings: trees decompositions

Common visual framework (Isenburg Snoeyink’05)

The (non-optimal) degree encoder gives much better codes for low entropy triangulations!

Patch of triangular grids ⇒ 6,6,6,6,6,6,5,6,6,6,6,6,6,6,6,7…

Alliez Desbrun (Eurographics ’01): could a degree encoder be optimal?

Gotsman (’06): No. Under constraints $\sum p_1 = 1$ and $\sum ip_i = 6$ on the proportion of vertices of degree $p_i$, the max entropy of degree sequence is $3.236 \text{ bpv} < 3.245 \text{ bpv}!$

Degree encoding, Touma-Gotsman (’98)

Leftmost tree in minimal canonical ordering Poulalhon, S. (’03)

? but efficient better?!

$3.24n$

110100011000001001000001100100000000
<table>
<thead>
<tr>
<th>Mesh compression</th>
<th>Graph encoding</th>
<th>Succinct representations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer graphics</td>
<td>Graph theory / combinatorics</td>
<td>Algorithms and DS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Edgebreaker</strong></td>
<td><strong>Turan</strong> ('84)</td>
<td>Jacobson (Focs89)</td>
</tr>
<tr>
<td>Rossignac ('99)</td>
<td><strong>Keeler Westbrook</strong> ('95)</td>
<td>Munro and Raman (Focs97)</td>
</tr>
<tr>
<td>Lope et al. ('03)</td>
<td><strong>He et al.</strong> ('99)</td>
<td></td>
</tr>
<tr>
<td>Lewiner et al. ('04)</td>
<td><strong>Chuang et al.</strong> (Icalp98)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chiang et al. (Soda01)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Valence (degree)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Touma and Gotsman ('98)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alliez and Debrun</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Isenburg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Khodakovsky</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Cut – border machine</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gumhold et al. (Siggraph '98)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gumhold (Soda '05)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A more generic approach?
First idea (following Luca Castelli Aleardi)

Decomposition of quadrangulations...by the french artist Léon Gischia (1903-1991)
Teacher Listen to me, If you cannot deeply understand these principles, these arithmetic archetypes, you will never perform correctly a "polytechnicien" job... you will never obtain a teaching position at "Ecole Polytechnique". For example, what is 3.755.918.261 multiplied by 5.162.303.508?

Student (very quickly) the result is 193891900145...

Teacher (very astonished) yes ... the product is really... But, how have you computed it, if you do not know the principles of arithmetic reasoning?

Student: it is simple: I have learned by heart all possible results of all possible different multiplications.
A hierarchical approach, with a dictionary at bottom.

Level 1:
- $\Theta\left(\frac{n}{\log^2 n}\right)$ regions of size $\Theta(\log^2 n)$, represented by pointers to level 2

Level 2:
- in each of the $\frac{n}{\log^2 n}$ regions
  - $\Theta(\log n)$ regions of size $C \log n$, represented by pointers to level 3

Level 3: exhaustive catalog of all different regions of size $i < C \log n$:
- complete explicit representation.
A hierarchical approach, with a dictionary at bottom.

Level 1:
- $\Theta\left(\frac{n}{\log^2 n}\right)$ regions of size $\Theta(\log^2 n)$, represented by pointers to level 2
- Global pointers of size $\log n$

Level 2:
in each of the $\frac{n}{\log^2 n}$ regions
- $\Theta(\log n)$ regions of size $C \log n$, represented by pointers to level 3
- Local pointers of size $\log \log n$

Level 3: exhaustive catalog of all different regions of size $i < C \log n$:
- Complete explicit explicit representation.
A hierarchical approach, with a dictionary at bottom.

Level 1:
- $\Theta\left(\frac{n}{\log^2 n}\right)$ regions of size $\Theta(\log^2 n)$, represented by pointers to level 2
- global pointers of size $\log n$

Level 2:
in each of the $\frac{n}{\log^2 n}$ regions
- $\Theta(\log n)$ regions of size $C \log n$, represented by pointers to level 3
- local pointers of size $\log \log n$

Level 3: exhaustive catalog of all different regions of size $i < C \log n$:
- complete explicit representation.

Dictionary space is $o(n)$ if $C$ small enough.
A hierarchical approach, with a dictionary at bottom.

Level 1:
- $\Theta\left(\frac{n}{\log^2 n}\right)$ regions of size $\Theta(\log^2 n)$, represented by pointers to level 2
- global pointers of size $\log n$

Space $O\left(\frac{n}{\log^2 n} \cdot \log n\right) = o(n)$

Level 2:
in each of the $\frac{n}{\log^2 n}$ regions
- $\Theta(\log n)$ regions of size $C \log n$, represented by pointers to level 3
- local pointers of size $\log \log n$

Level 3: exhaustive catalog of all different regions of size $i < C \log n$:
- complete explicit representation.

Dictionary space is $o(n)$ if $C$ small enough.
A hierarchical approach, with a dictionary at bottom.

Level 1:
• $\Theta\left(\frac{n}{\log^2 n}\right)$ regions of size $\Theta(\log^2 n)$, represented by pointers to level 2
• global pointers of size $\log n$

Space $O\left(\frac{n}{\log^2 n} \cdot \log n\right) = o(n)$

Level 2:
in each of the $\frac{n}{\log^2 n}$ regions
• $\Theta(\log n)$ regions of size $C \log n$, represented by pointers to level 3
• local pointers of size $\log \log n$

Space $O\left(\frac{n}{\log n} \cdot \log \log n\right) = o(n)$

Level 3: exhaustive catalog of all different regions of size $i < C \log n$:
• complete explicit representation.

Dictionary space is $o(n)$ if $C$ small enough.
A hierarchical approach, with a dictionary at bottom.

Dominant term?

The dominant term is given by the sum of references to the dictionary references on objects of $\mathcal{T}_k$ have size $\log_2 \mathcal{T}_k \sim 2.175k$ if $k \to \infty$.

$$\sum_j 2.175k_j = 2.175m \text{ bits}$$

2.175bpt is entropy of triangulations with a boundary.
A hierarchical approach, with a dictionary at bottom.

Dominant term?

The dominant term is given by the sum of references to the dictionary references on objects of $\mathcal{T}_k$ have size $\log_2 \mathcal{T}_k \sim 2.175k$ if $k \to \infty$

we should take all $k$ s.t. $\frac{1}{12} \log n < k < \frac{1}{2} \log n$

$$\sum_j 2.175k_j = 2.175m \text{ bits}$$

2.175bpt is entropy of triangulations with a boundary
A hierarchical approach, with a dictionary at bottom.

Dominant term?

The dominant term is given by the sum of references to the dictionary references on objects of $T_k$ have size $\log_2 T_k \sim 2.175k$ if $k \to \infty$

we should take all $k$ s.t. $\frac{1}{12} \log n < k < \frac{1}{2} \log n$

$\sum_j 2.175k_j = 2.175m$ bits

2.175bpt is entropy of triangulations with a boundary larger than previous $\frac{1}{2} \cdot 3.24$ bpt
A hierarchical approach, with a dictionary at bottom.

Dominant term?

The dominant term is given by the sum of references to the dictionary references on objects of $\mathcal{T}_k$ have size $\log_2 \mathcal{T}_k \sim 2.175k$ if $k \to \infty$

we should take all $k$ s.t. $\frac{1}{12} \log n < k < \frac{1}{2} \log n$

$\sum_j 2.175k_j = 2.175m$ bits

A hierarchical approach, with a dictionary at bottom.

Dominant term?

The dominant term is given by the sum of references to the dictionary references on objects of $\mathcal{T}_k$ have size $\log_2 \mathcal{T}_k \sim 2.175k$ if $k \to \infty$

we should take all $k$ s.t. $\frac{1}{12} \log n < k < \frac{1}{2} \log n$

$\sum_j 2.175k_j = 2.175m$ bits

2.175bpt is entropy of triangulations with a boundary larger than previous $\frac{1}{2} \cdot 3.24bpt$
A word of conclusion

- A relatively generic method to get adaptative s.d.s:
  - triangulations with boundary, trees, polyhedral maps...
  - but complex hierarchical structure, unpractical subleading terms...
  - develop "elegant" succinct data structures:
    - a non asymptotic $2n + O(\log n)$ bits sds for plane trees with $n$ vertices?

- Some examples of nice optimal encodings
  - but not so adaptative and no query support
  - find an optimal adaptative encoder for triangulations with given degrees
  - find other parameters of trees or maps that allow for simple adaptative compression or sds (depth?)