Combinatorial entropy and succinct data structures

Gilles Schaeffer

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

Analysis of Algorithms, 2009

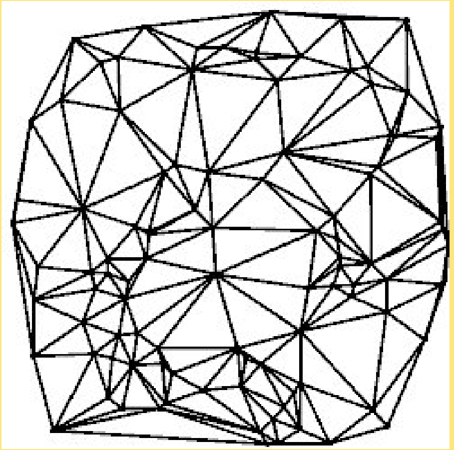
LIX

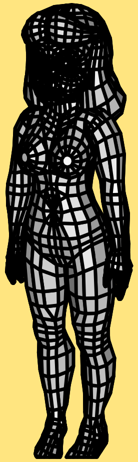
ÉCOLE POLYTECHNIQUE

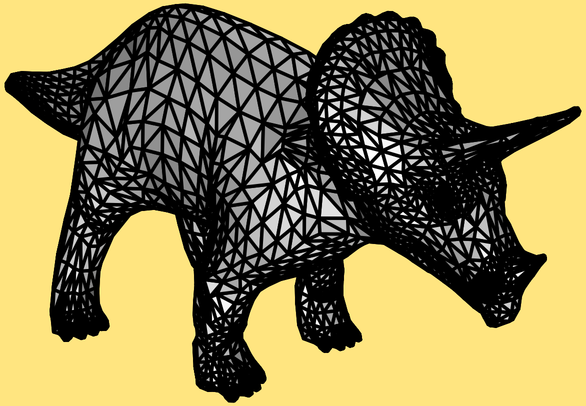
cnrs dépasser les frontières

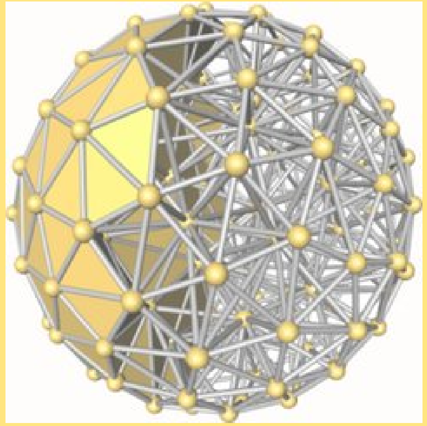
Before we start... Geometric data ; meshes

Among data structures for geometric data, I pick meshes...

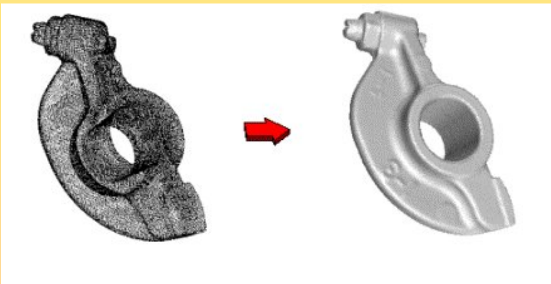




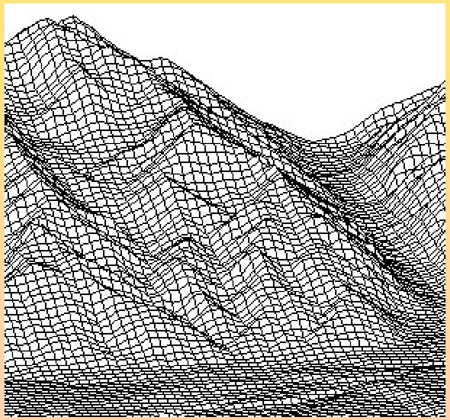




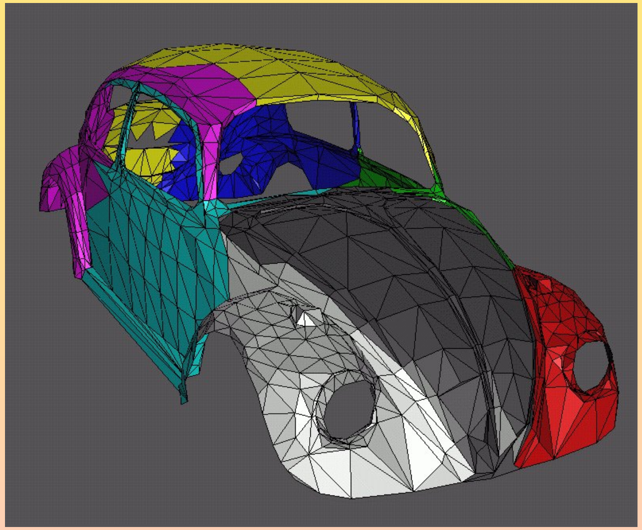
Surface recontruction
from sampling



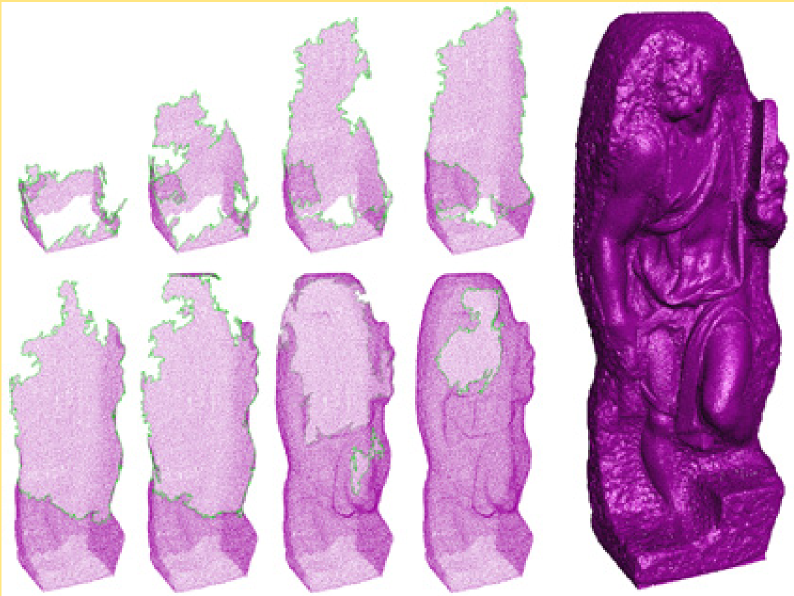
Geographic information
systems



Surface modelling



Before we start... $\exists$ 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

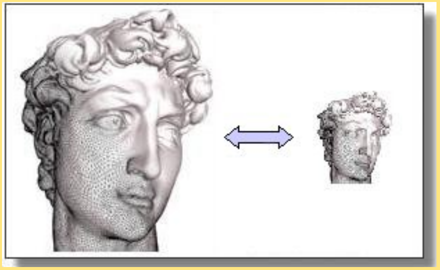
David statue (Stanford's Digital Michelangelo Project, 2000)

2 billions polygons
32 Giga bytes (without compression)

No existing algorithm nor data structure for dealing with the entire model

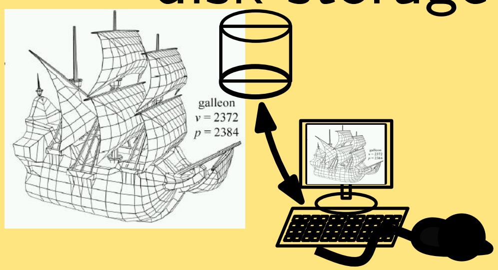
Before we start... What we are aiming at

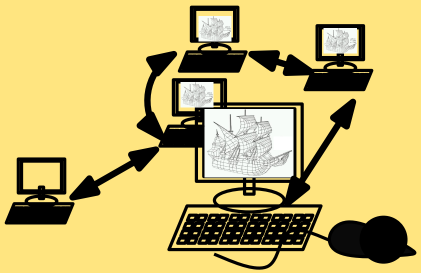
Mesh compression



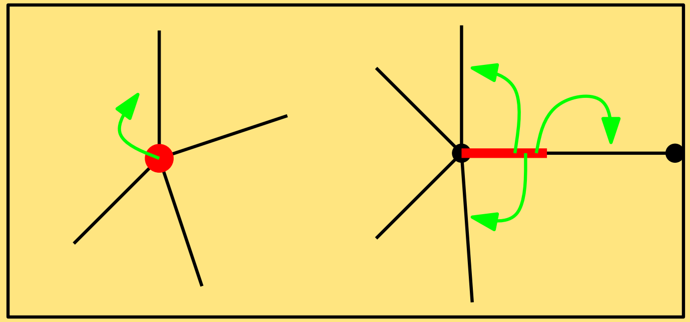
Transmission

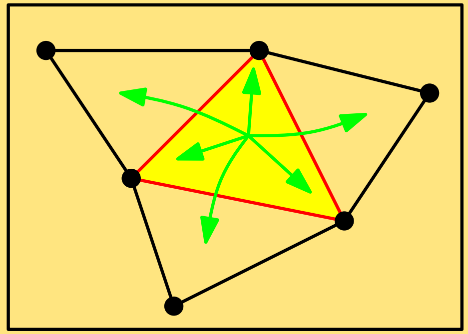
disk storage





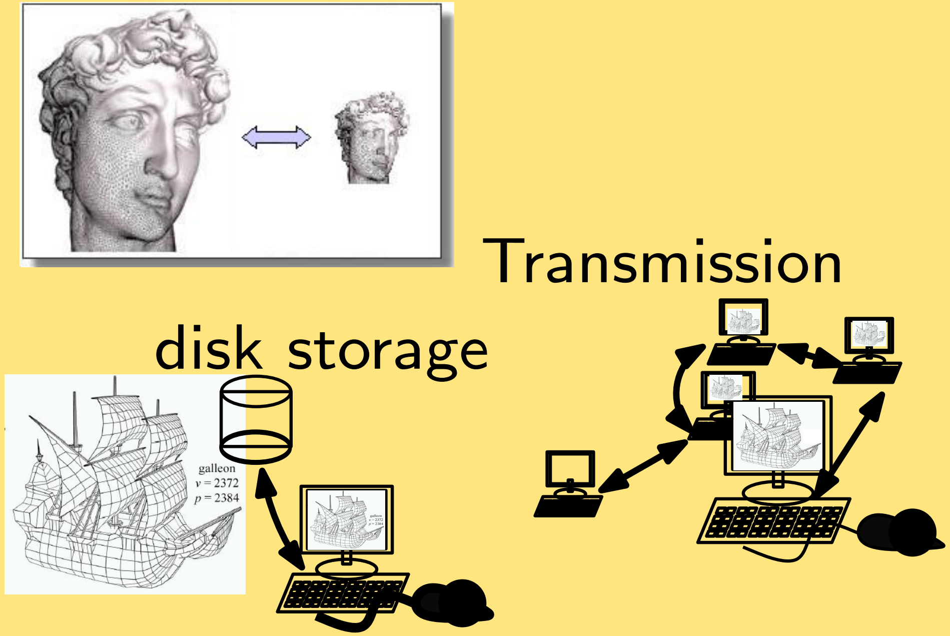
Geometric data structures



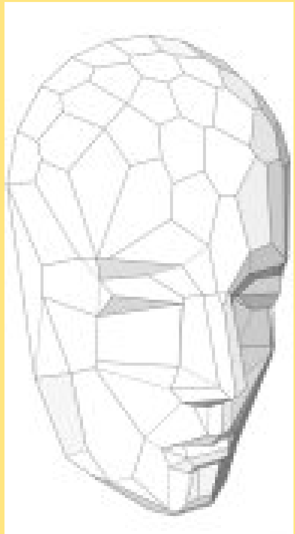


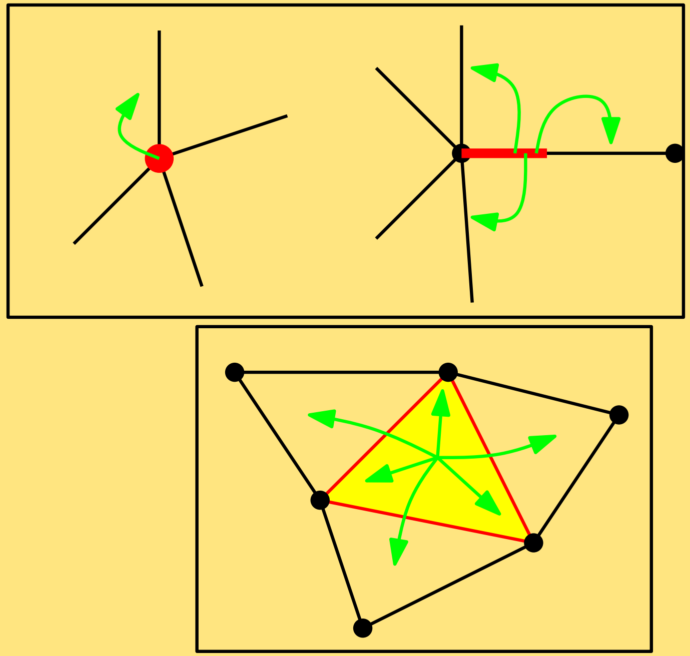
Before we start... What we are aiming at

Mesh compression

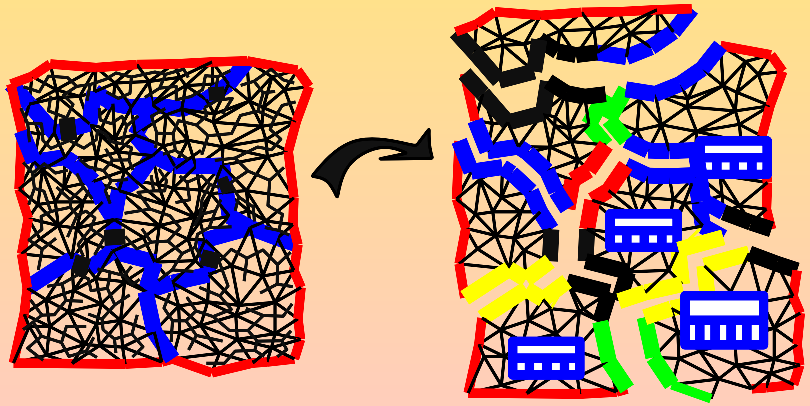


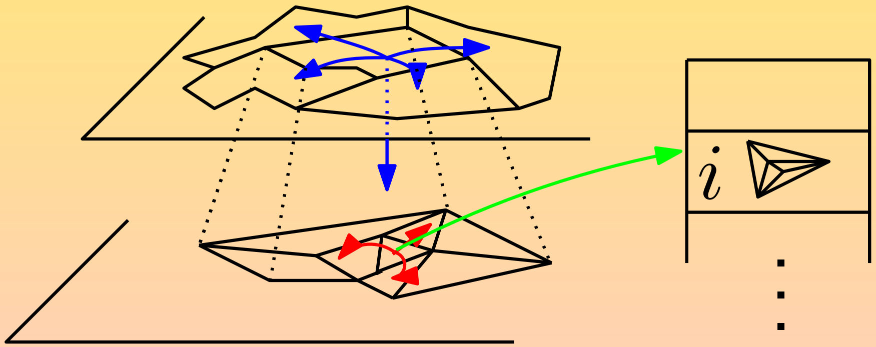
Geometric data structures





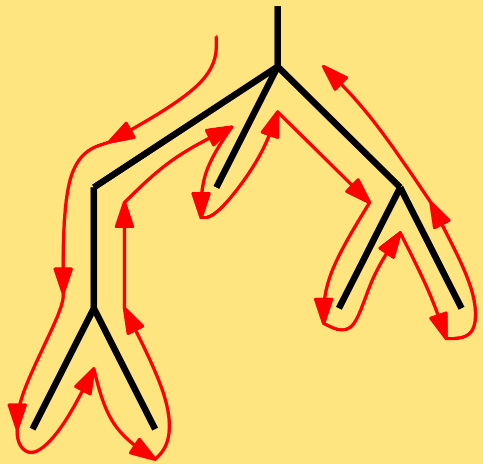
MERGE INTO: Compact representations of geometric data structures



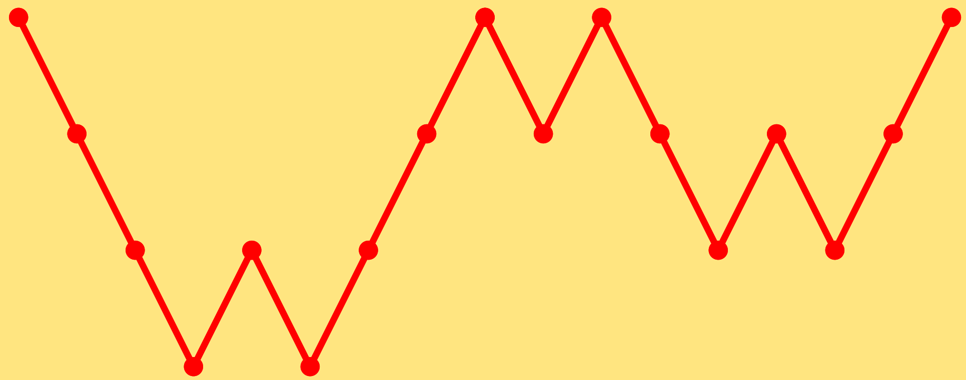


Starter: the encoding of plane trees

ordered tree with n edges



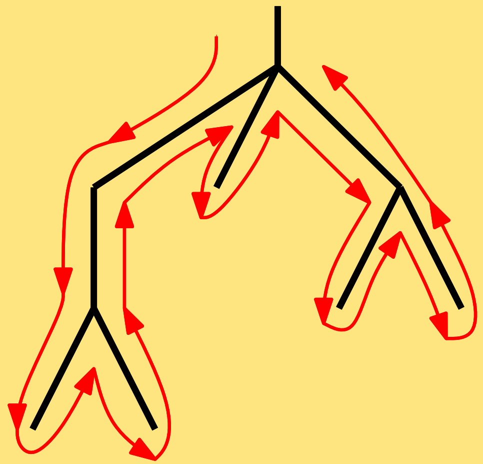
balanced parenthesis word of length $2n$



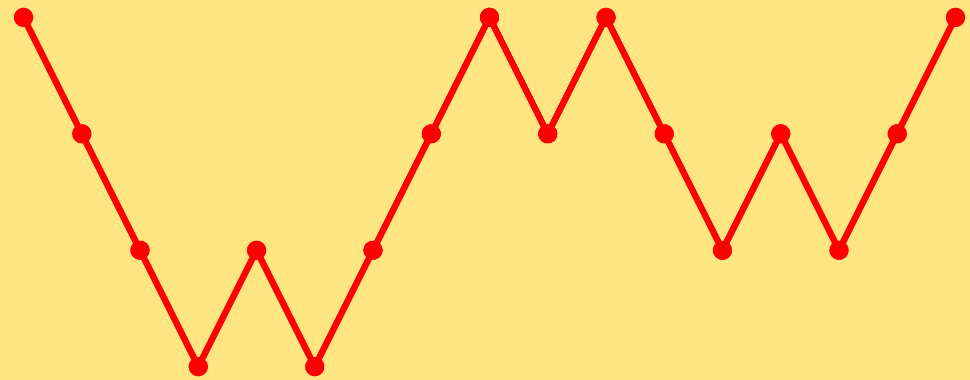
1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0

Starter: the encoding of plane trees

ordered tree with n edges



balanced parenthesis word of length $2n$

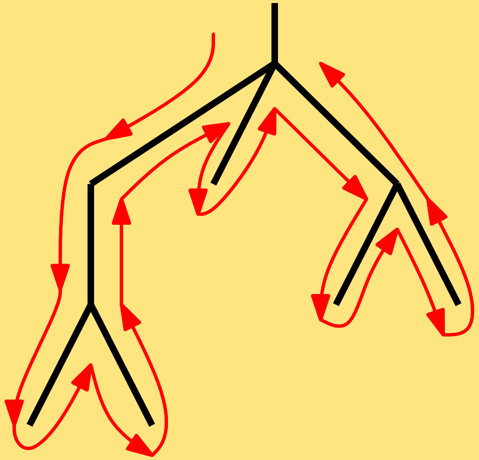


$1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0$

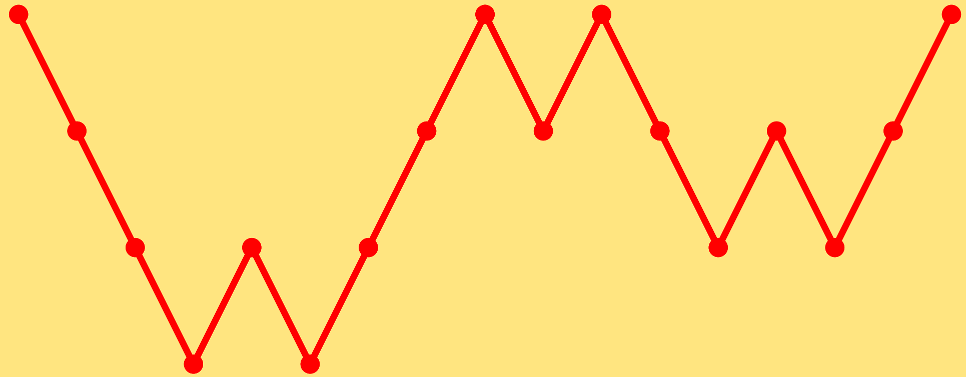
$\Rightarrow 2n$ bits for encoding an ordered tree with n edges

Starter: the encoding of plane trees

ordered tree with n edges



balanced parenthesis word of length $2n$



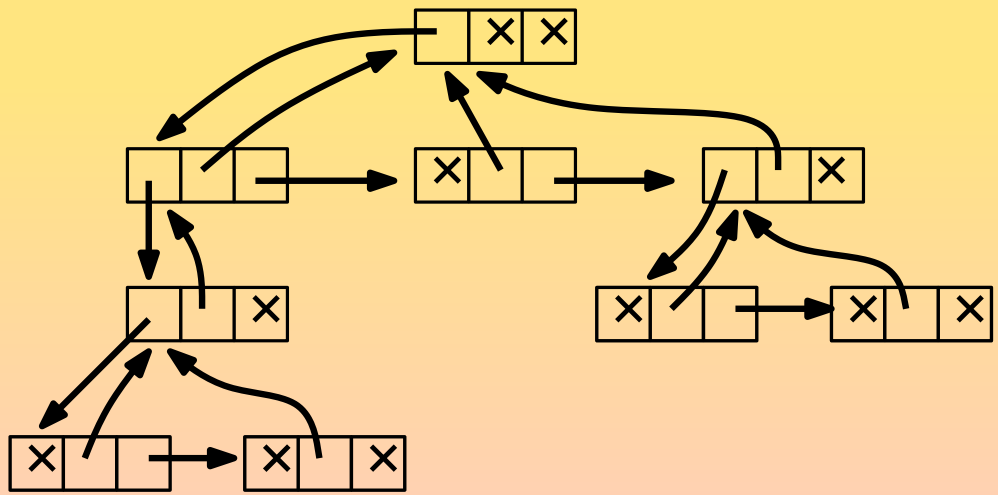
$1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0$

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

Compare to the standard explicit represention:

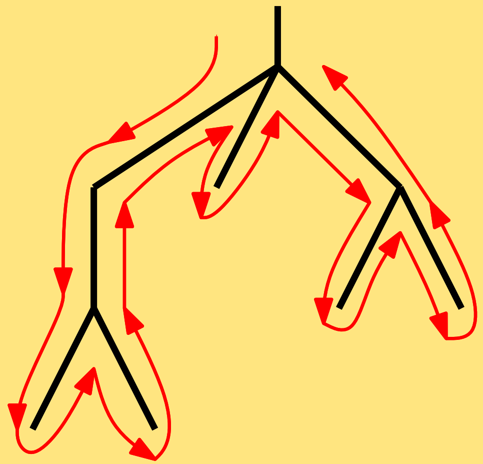
$3n$ pointers $\approx$ 96 bits

$3n \log n$ in theory

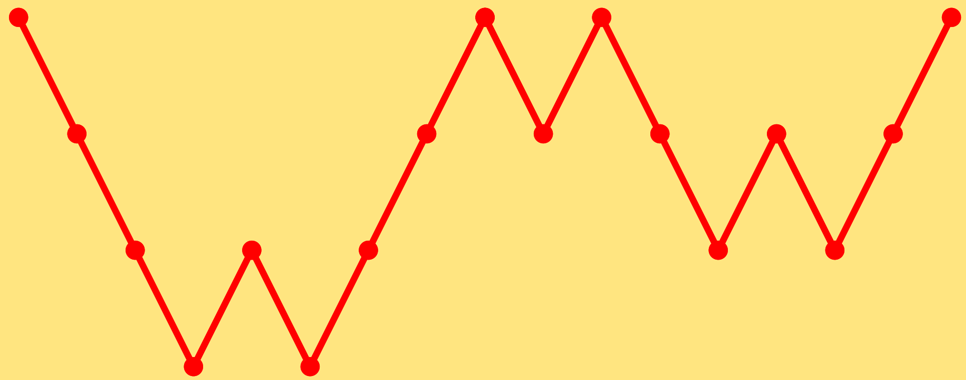


Starter: the encoding of plane trees

ordered tree with n edges



balanced parenthesis word of length $2n$



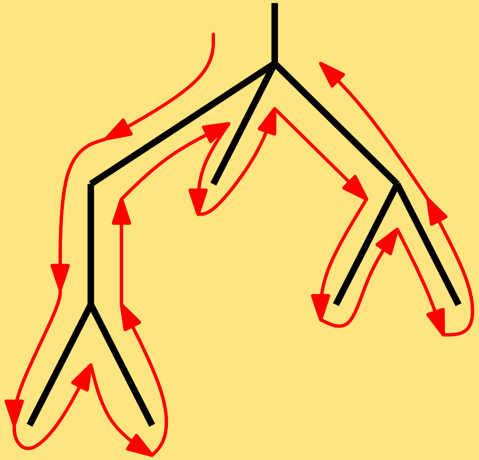
1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0

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

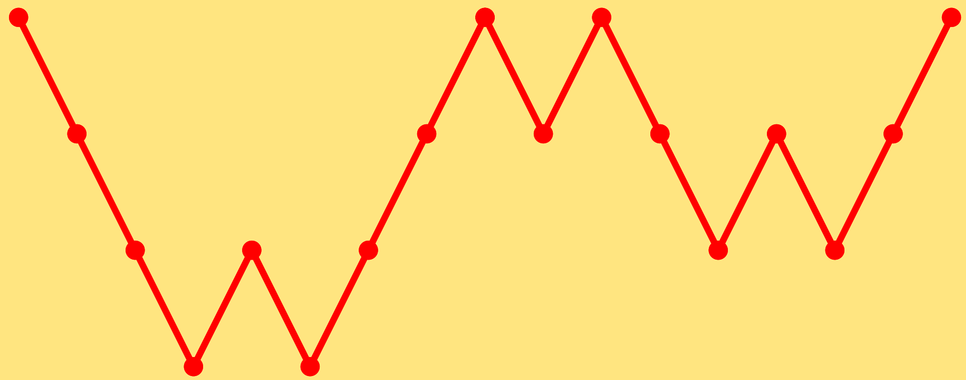
enumeration: $\|\mathcal{B}_n\| = \frac{1}{n+1}\binom{2n}{n} \approx 2^{2n} n^{-\frac{3}{2}}$

Starter: the encoding of plane trees

ordered tree with n edges



balanced parenthesis word of length $2n$



$1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0$

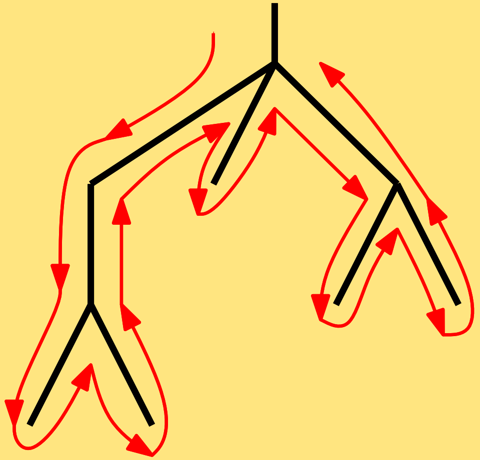
$\Rightarrow 2n$ bits for encoding an ordered tree with n edges

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

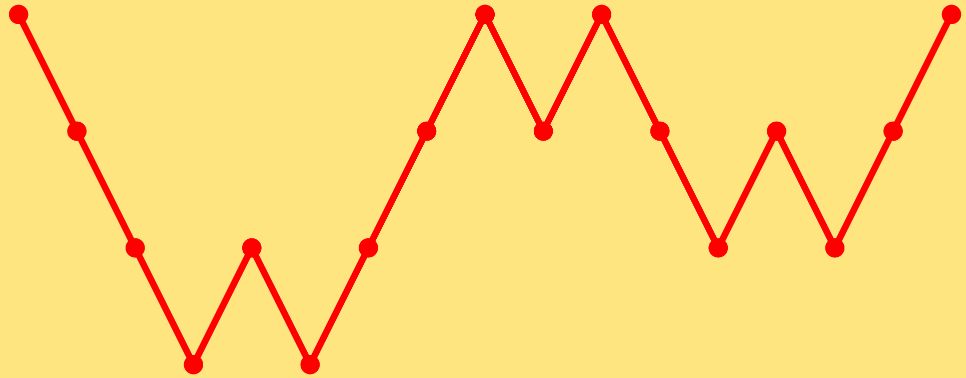
$$\log_2 \|\mathcal{B}_n\| = 2n + O(\lg n) \text{ bpv}$$

Starter: the encoding of plane trees

ordered tree with n edges



balanced parenthesis word of length $2n$



$1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0$

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

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

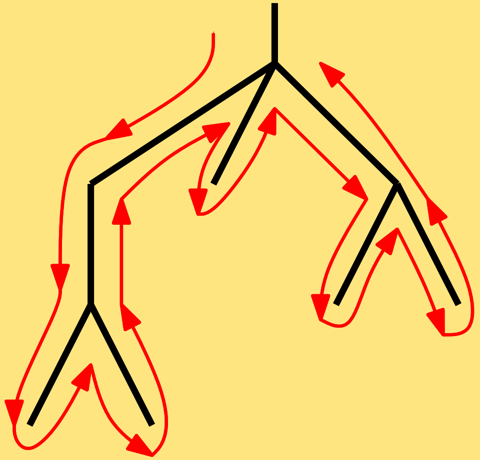
$$\log_2 \|\mathcal{B}_n\| = 2n + O(\lg n) \text{ bpv}$$

This is an optimal encoding!

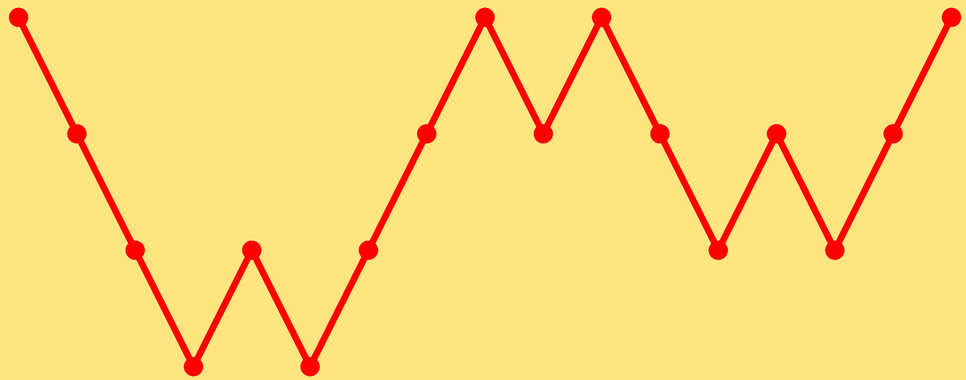
it matches asymptotically the information-theory lower bound

Starter: the encoding of plane trees

ordered tree with n edges



balanced parenthesis word of length $2n$



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

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

exponential growth rate

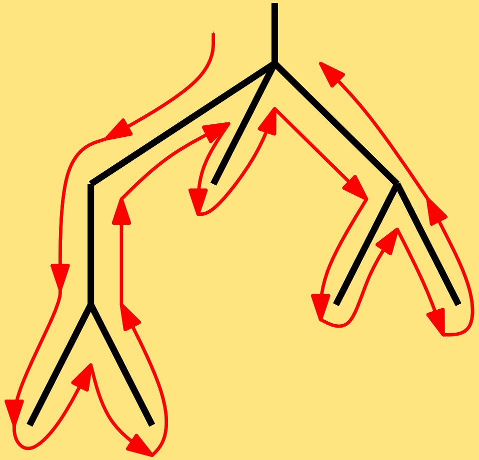
$$\log_2 \|\mathcal{B}_n\| = 2n + O(\lg n) \text{ bpv}$$

This is an optimal encoding!

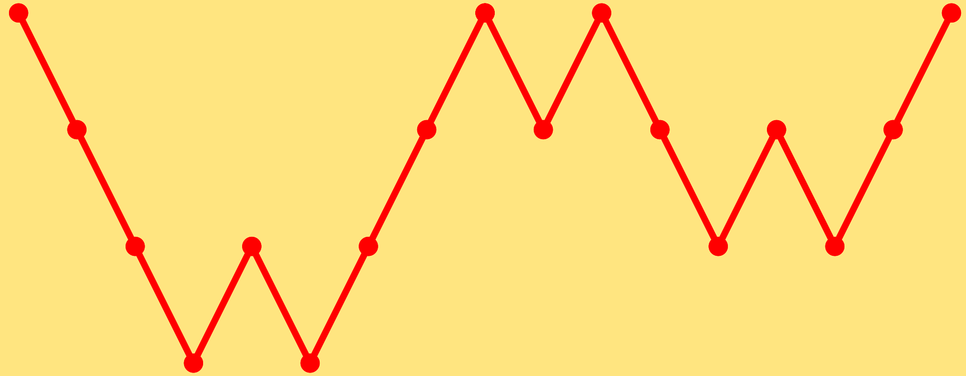
it matches asymptotically the information-theory lower bound

Starter: the encoding of plane trees

ordered tree with n edges



balanced parenthesis word of length $2n$



$$1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0$$

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

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

$$\log_2 \|\mathcal{B}_n\| = 2n + O(\lg n) \text{ bpv}$$

exponential growth rate

$\Updownarrow$

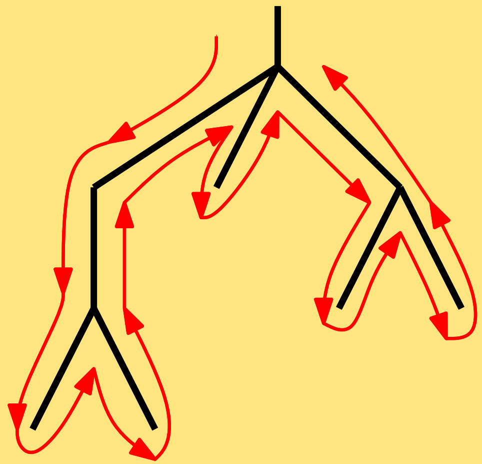
combinatorial entropy

This is an optimal encoding!

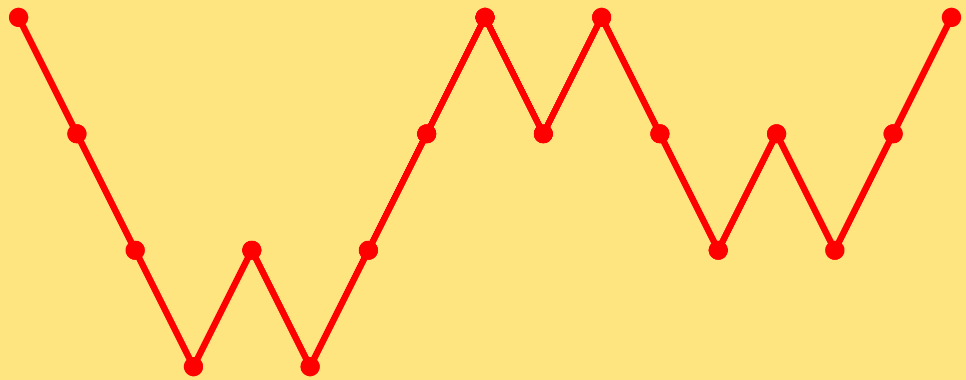
it matches asymptotically the information-theory lower bound

Starter: linear space data structures for plane trees?

ordered tree with n edges



balanced parenthesis word of length $2n$

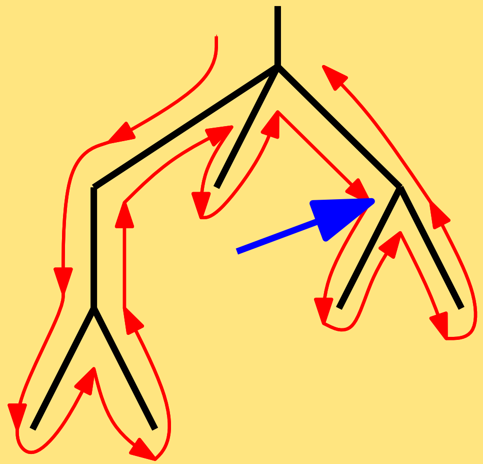


$1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0$

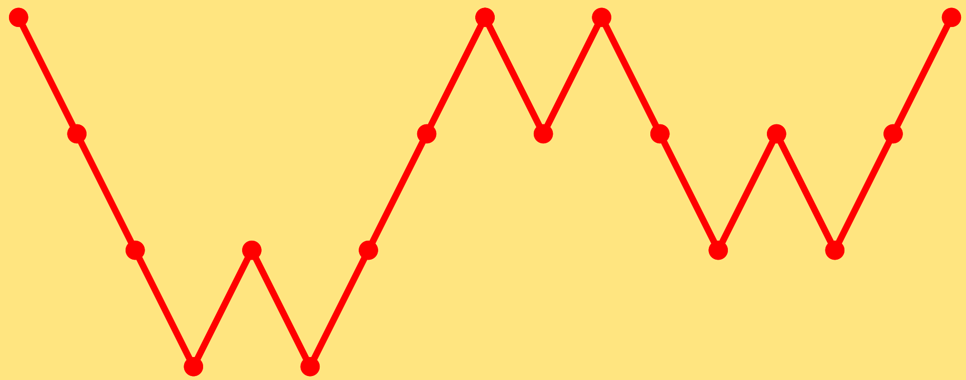
Navigation in the tree: handlers

Starter: linear space data structures for plane trees?

ordered tree with n edges



balanced parenthesis word of length $2n$

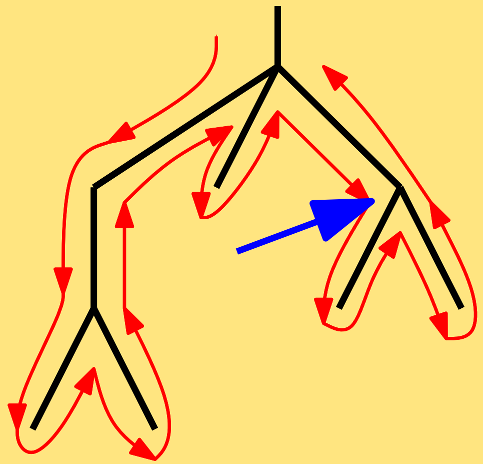


$1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0$

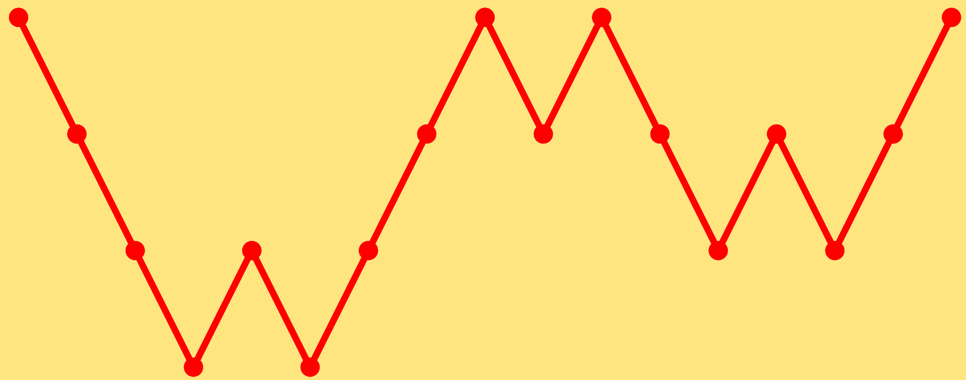
Navigation in the tree: handlers

Starter: linear space data structures for plane trees?

ordered tree with n edges

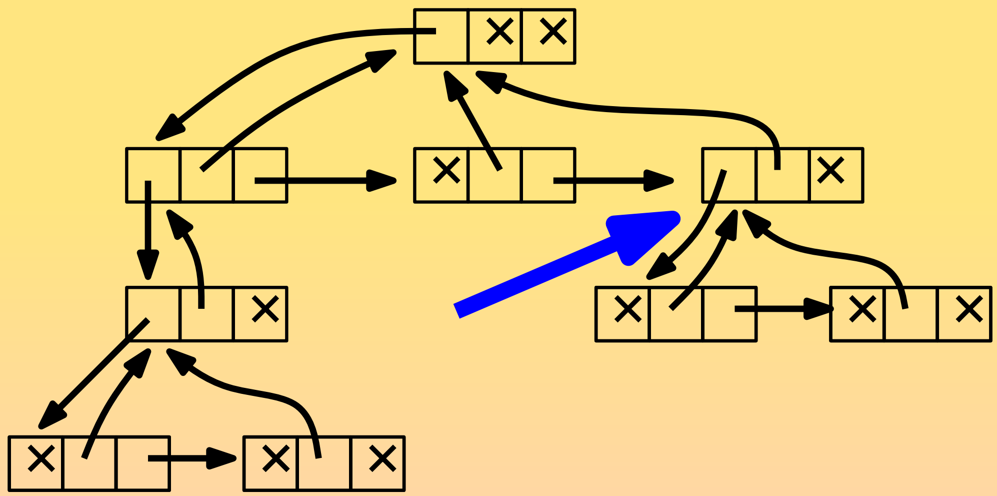


balanced parenthesis word of length $2n$



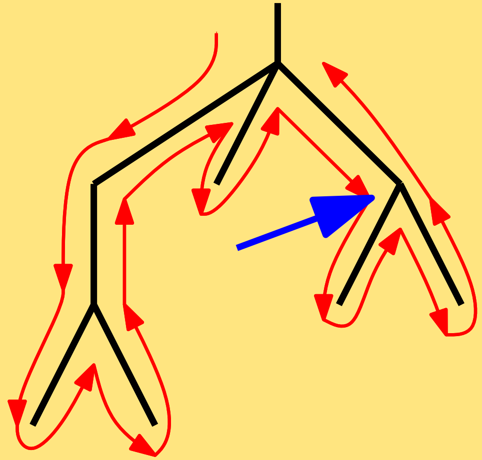
1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0

Navigation in the tree: handlers

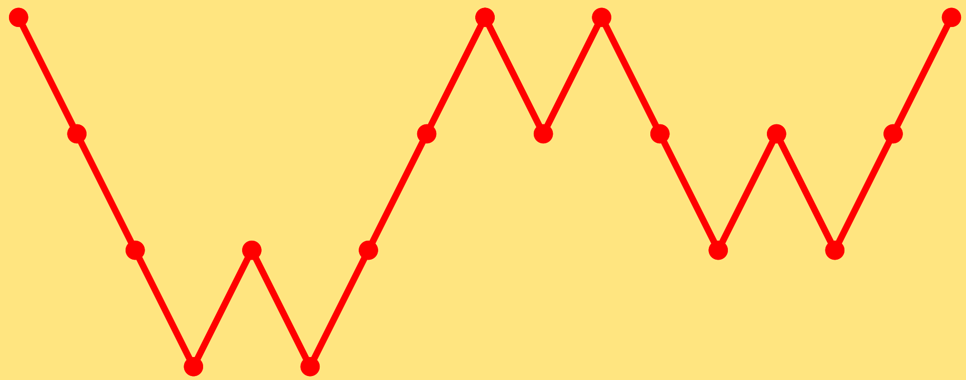


Starter: linear space data structures for plane trees?

ordered tree with n edges



balanced parenthesis word of length $2n$



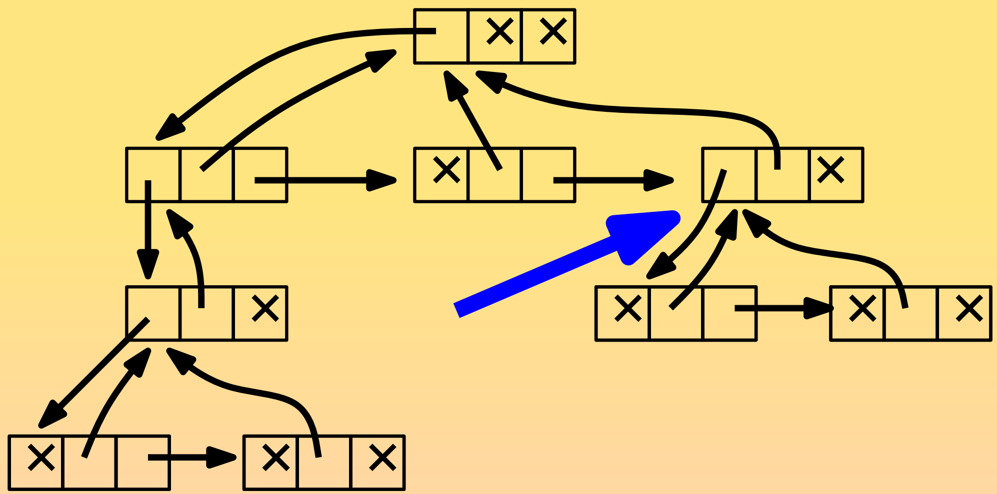
$1\,1\,1\,0\,1\,0\,0\,0\,1\,0\,1\,1\,0\,1\,0\,0$

Navigation in the tree: handlers

move the handler to first son

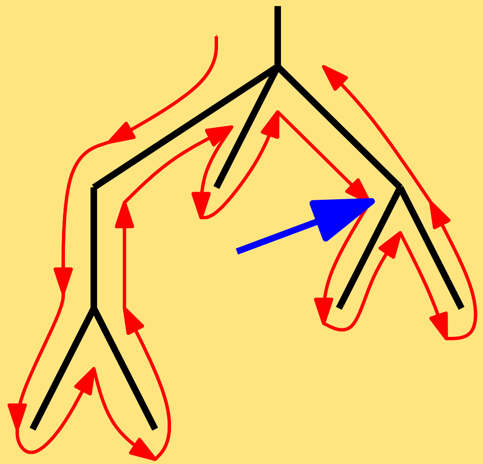
move the handler to next brother

move the handler to father

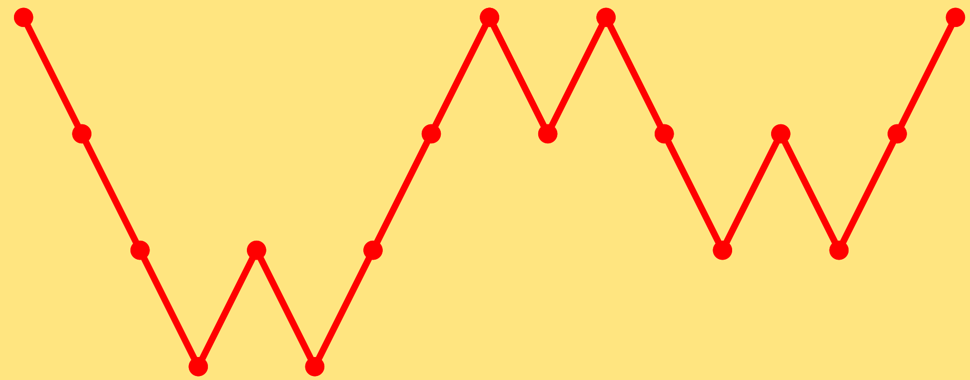


Starter: linear space data structures for plane trees?

ordered tree with n edges



balanced parenthesis word of length $2n$



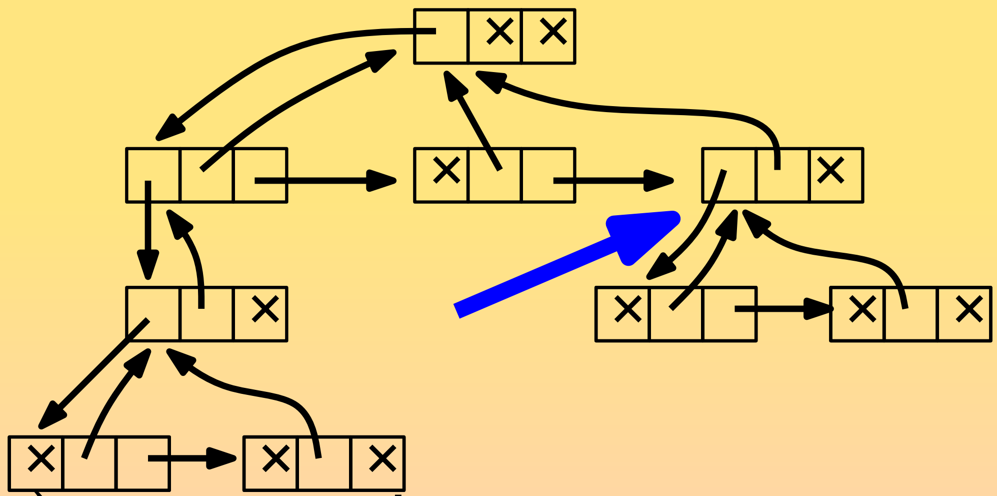
1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0

Navigation in the tree: handlers

move the handler to first son

move the handler to next brother

move the handler to father

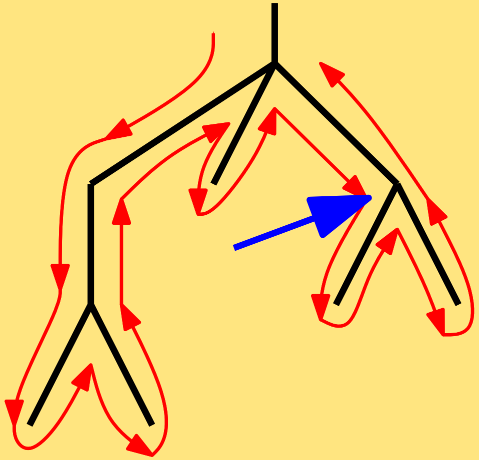


Constant time with standard (pointer) representation

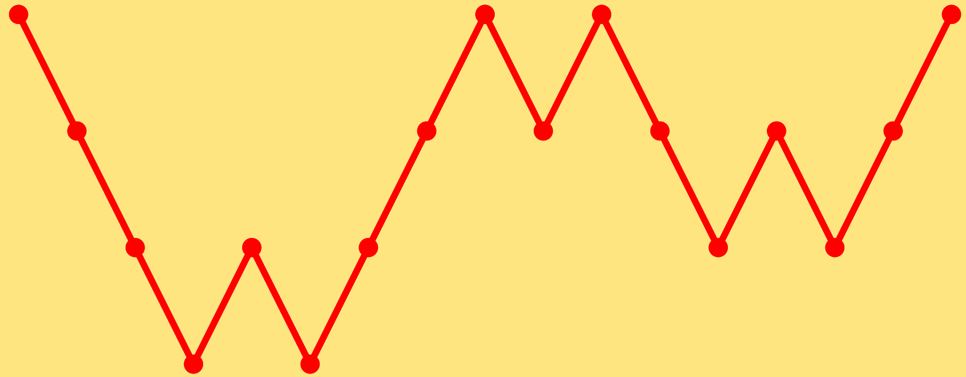
but the pointer based representation uses $\Theta(n \log n)$ bits

Starter: linear space data structures for plane trees?

ordered tree with n edges



balanced parenthesis word of length $2n$



1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0

Navigation in the tree: handlers

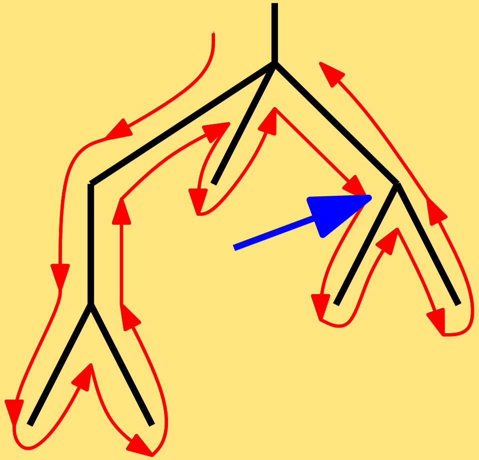
handler = index of opening bracket

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

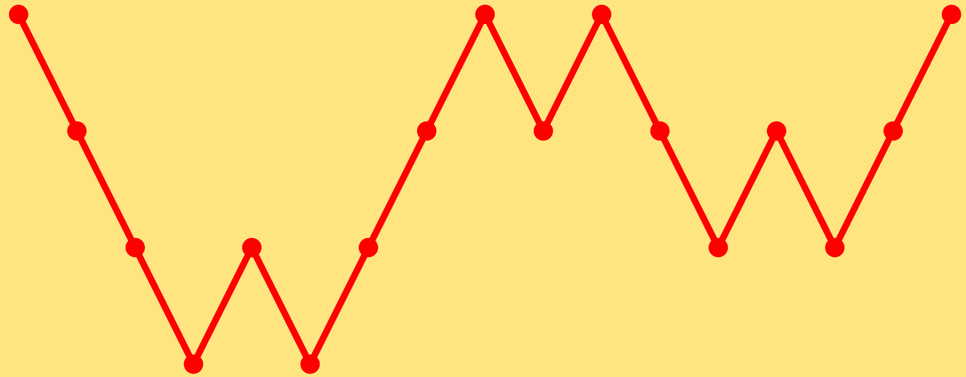
Constant time with standard (pointer) representation
but the pointer based representation uses $\Theta(n \log n)$ bits

Starter: linear space data structures for plane trees?

ordered tree with n edges



balanced parenthesis word of length $2n$



1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0

Navigation in the tree: handlers

handler = index of opening bracket

move the handler to first son

move the handler to next brother

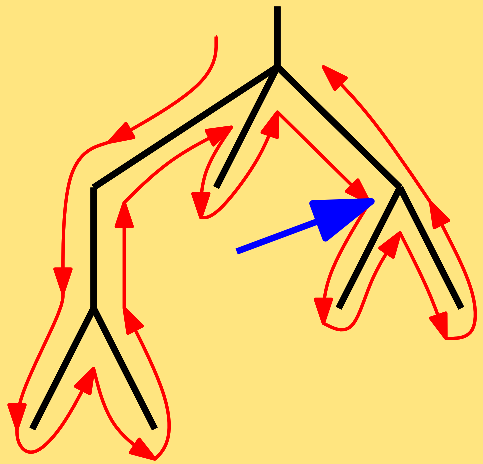
move the handler to father

Constant time with standard (pointer) representation

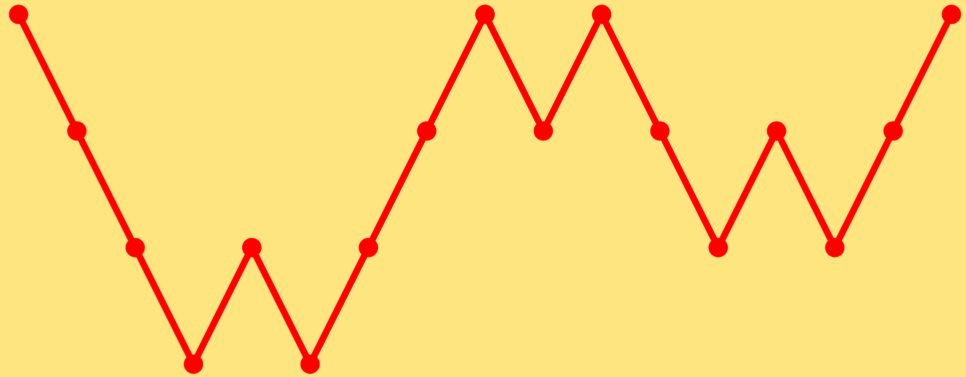
but the pointer based representation uses $\Theta(n \log n)$ bits

Starter: linear space data structures for plane trees?

ordered tree with n edges



balanced parenthesis word of length $2n$



$1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ \boxed{1}\ 1\ 0\ 1\ 0\ 0$

Navigation in the tree: handlers

handler $=$ index of opening bracket

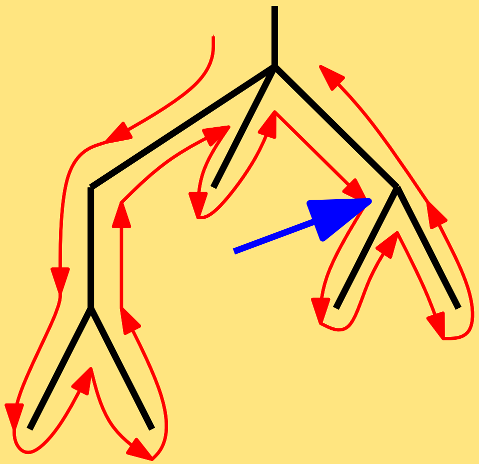
- move the handler to first son — index $\rightarrow$ index$+1$
- move the handler to next brother
- move the handler to father

Constant time with standard (pointer) representation

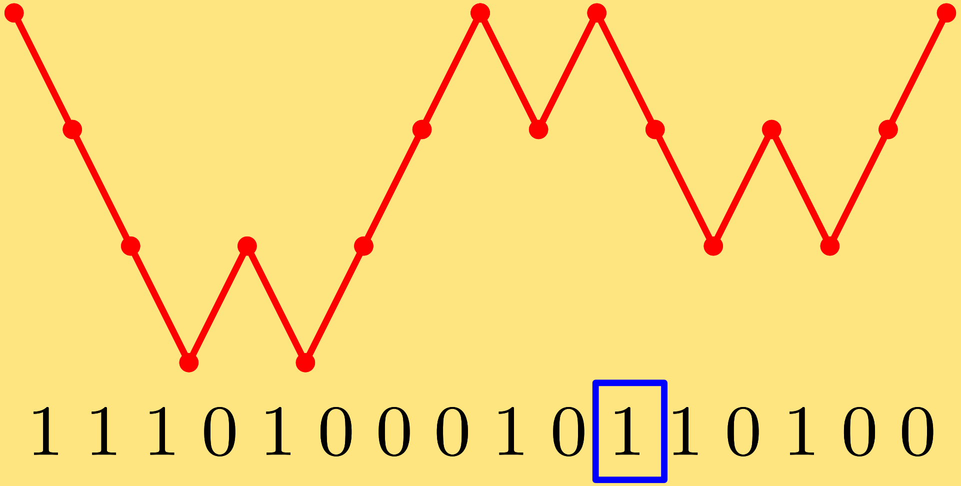
but the pointer based representation uses $\Theta(n \log n)$ bits

Starter: linear space data structures for plane trees?

ordered tree with n edges



balanced parenthesis word of length $2n$



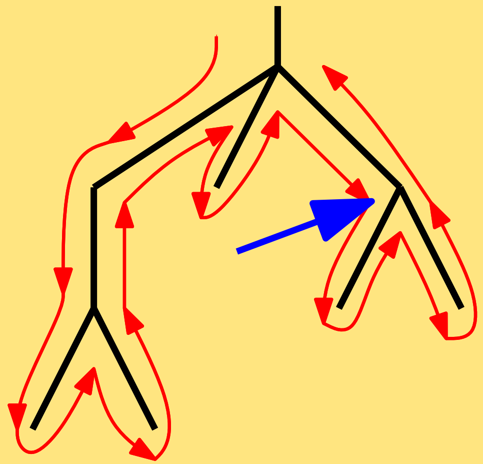
Navigation in the tree: handlers — handler $=$ index of opening bracket

- move the handler to first son — index $\rightarrow$ index$+1$
- move the handler to next brother — index $\rightarrow$ matching(index)$+1$
- move the handler to father

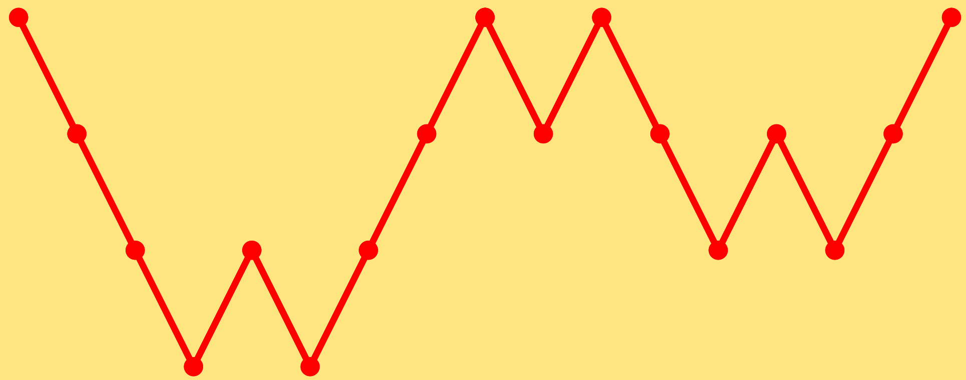
Constant time with standard (pointer) representation
but the pointer based representation uses $\Theta(n \log n)$ bits

Starter: linear space data structures for plane trees?

ordered tree with n edges



balanced parenthesis word of length $2n$



$1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ \boxed{1}\ 1\ 0\ 1\ 0\ 0$

Navigation in the tree: handlers — handler $=$ index of opening bracket

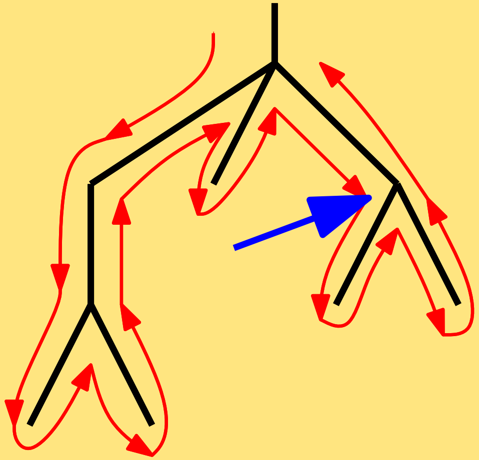
move the handler to first son	index $\rightarrow$ index$+1$
move the handler to next brother	index $\rightarrow$ matching(index)$+1$
move the handler to father	index $\rightarrow$ outer(index)

Constant time with standard (pointer) representation

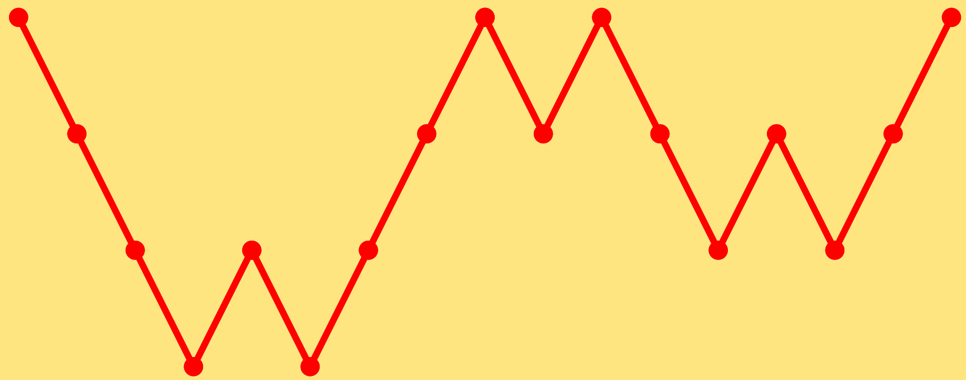
but the pointer based representation uses $\Theta(n \log n)$ bits

Starter: linear space data structures for plane trees?

ordered tree with n edges



balanced parenthesis word of length $2n$



1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0

Navigation in the tree: handlers — handler $=$ index of opening bracket

move the handler to first son	index $\rightarrow$ index$+1$
move the handler to next brother	index $\rightarrow$ matching(index)$+1$
move the handler to father	index $\rightarrow$ outer(index)

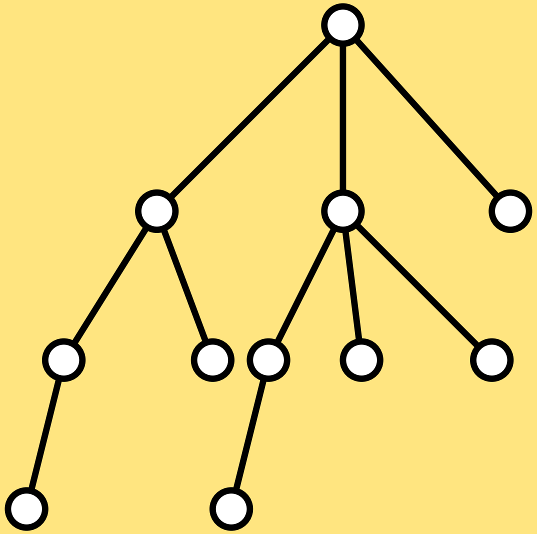
up to linear time!

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

(Jacobson, Focs89)



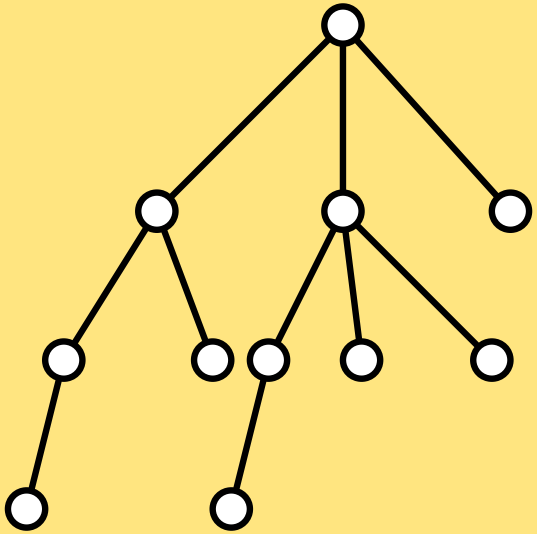
Decompose into m small blocks of size ε

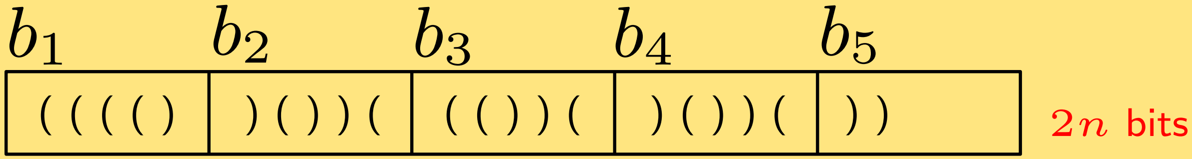
b_1	b_2	b_3	b_4	b_5	
(((( )	) ( )) (	(( )) (	) ( )) (	))	$2n$ bits

Starter: linear space data structures for plane trees

(Jacobson, Focs89)

Decompose into m small blocks of size ε

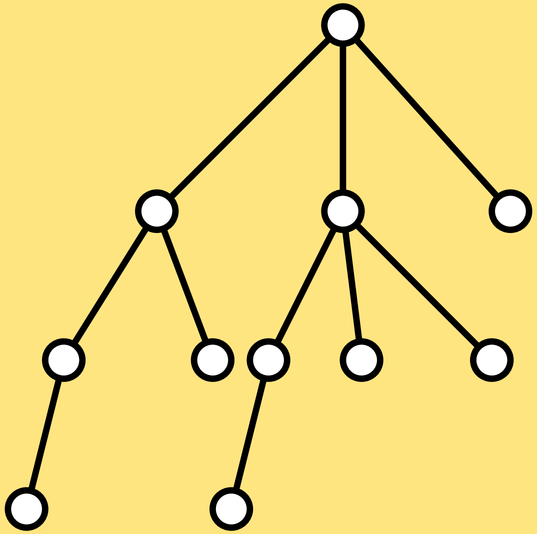




matching(index): go slowly inside block

Starter: linear space data structures for plane trees

(Jacobson, Focs89)



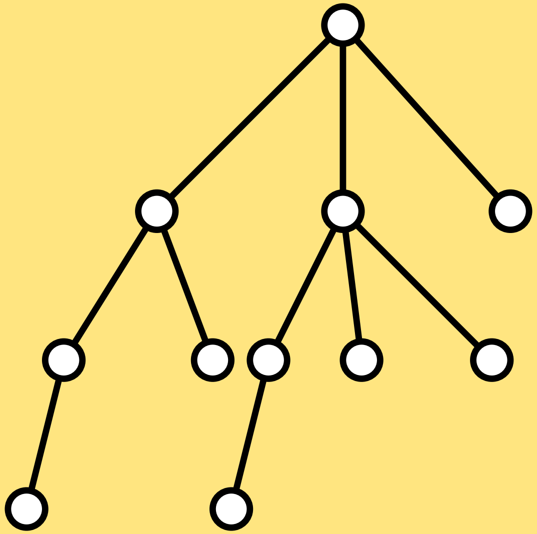
Decompose into m small blocks of size ε

b_1	b_2	b_3	b_4	b_5	
(((( )	) ( )) (	(( )) (	) ( )) (	))	$2n$ bits

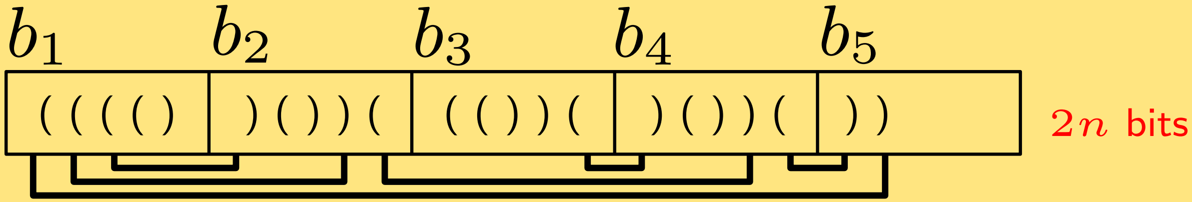
matching(index): go slowly inside block   if border reached: interblock

Starter: linear space data structures for plane trees

(Jacobson, Focs89)



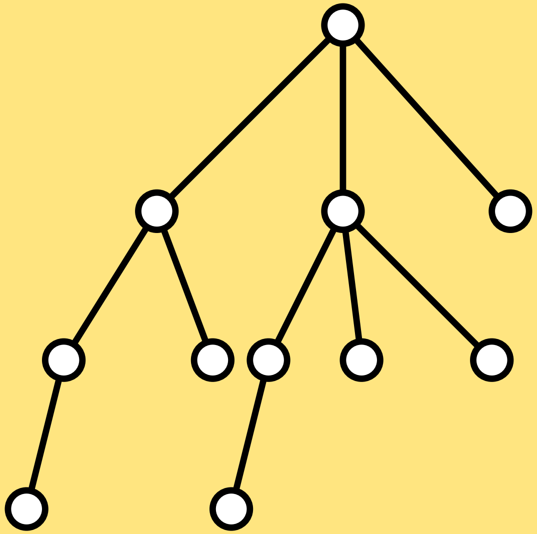
Decompose into m small blocks of size ε



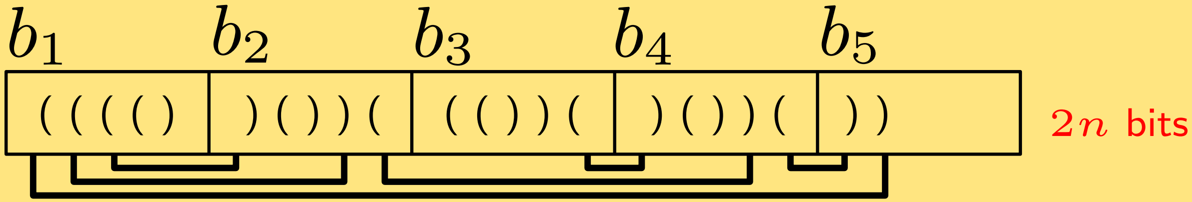
matching(index): go slowly inside block   if border reached: interblock

Starter: linear space data structures for plane trees

(Jacobson, Focs89)



Decompose into m small blocks of size ε

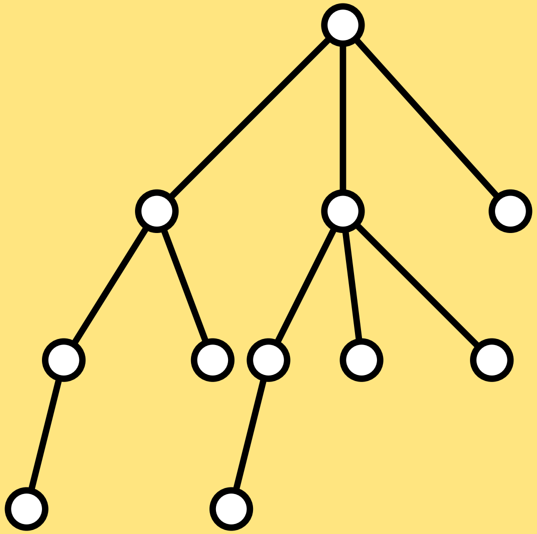


matching(index): go slowly inside block if border reached: interblock

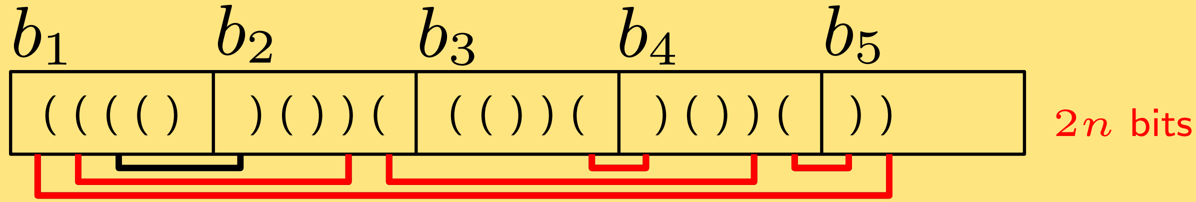
encode interblock explicitely: up to n edges $\Rightarrow$ space $n \log n$

Starter: linear space data structures for plane trees

(Jacobson, Focs89)



Decompose into m small blocks of size ε



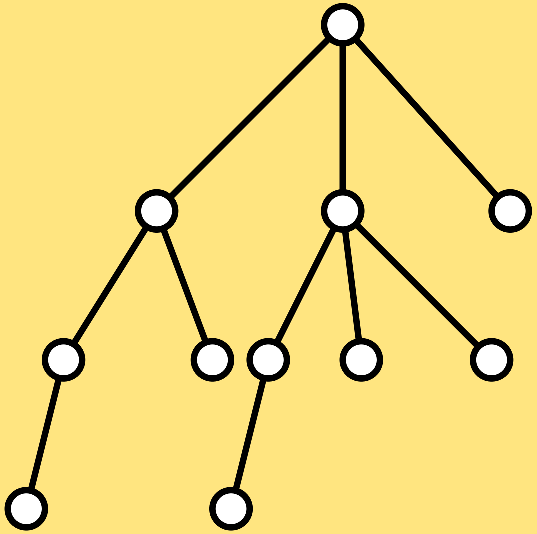
matching(index): go slowly inside block if border reached: interblock

encode interblock explicitely: up to n edges $\Rightarrow$ space $n \log n$

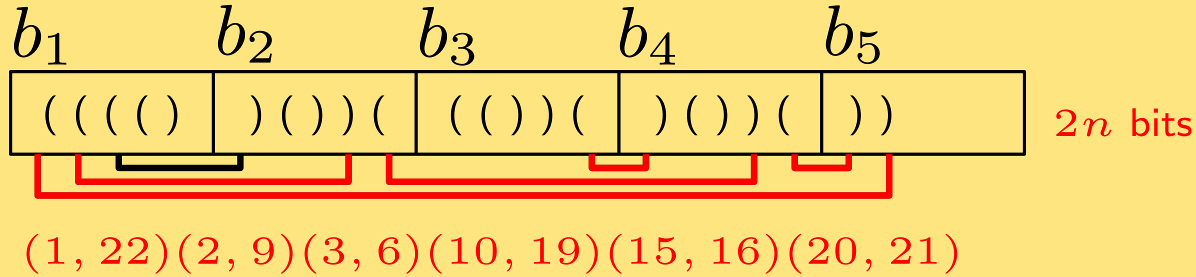
encode $\leq m$-1 pioneers (outermost between blocks) $\Rightarrow$ space $m \log n$

Starter: linear space data structures for plane trees

(Jacobson, Focs89)



Decompose into m small blocks of size ε



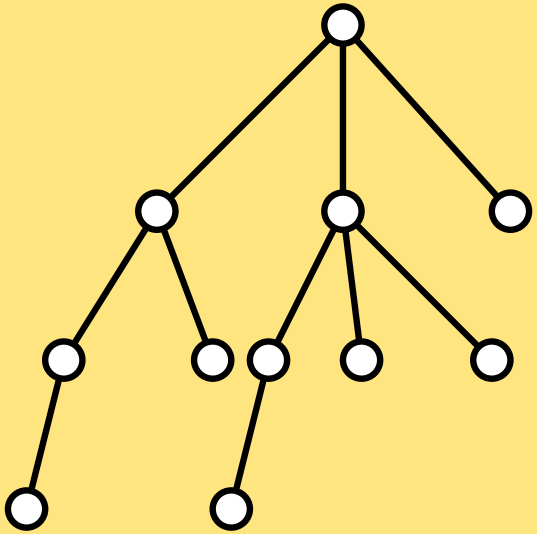
matching(index): go slowly inside block if border reached: interblock

encode interblock explicitely: up to n edges $\Rightarrow$ space $n \log n$

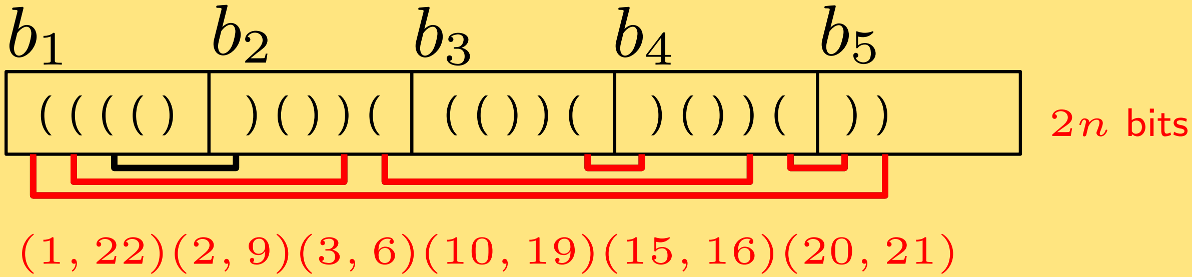
encode $\leq m$-1 pioneers (outermost between blocks) $\Rightarrow$ space $m \log n$

Starter: linear space data structures for plane trees

(Jacobson, Focs89)



Decompose into m small blocks of size ε



matching(index): go slowly inside block if border reached: interblock

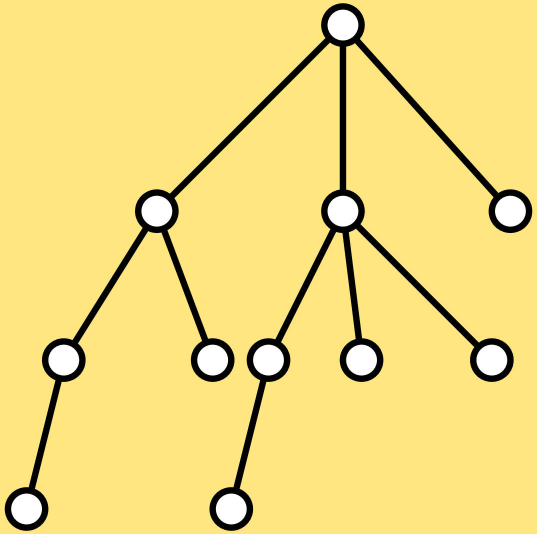
encode interblock explicitely: up to n edges $\Rightarrow$ space $n \log n$

encode $\leq m$-1 pioneers (outermost between blocks) $\Rightarrow$ space $m \log n$

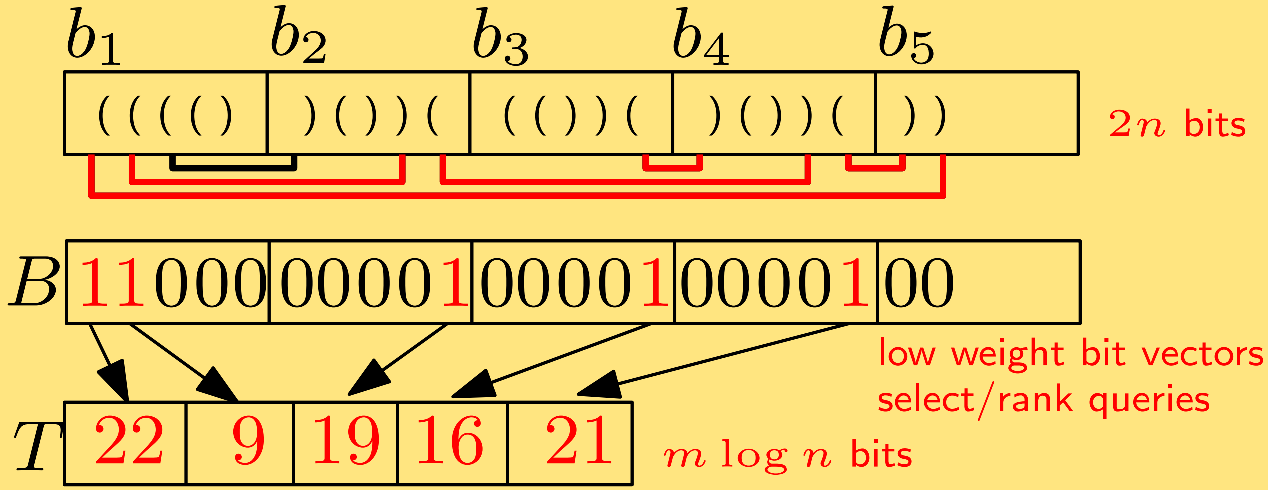
the explicit representation must allow navigation...

Starter: linear space data structures for plane trees

(Jacobson, Focs89)



Decompose into m small blocks of size ε



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

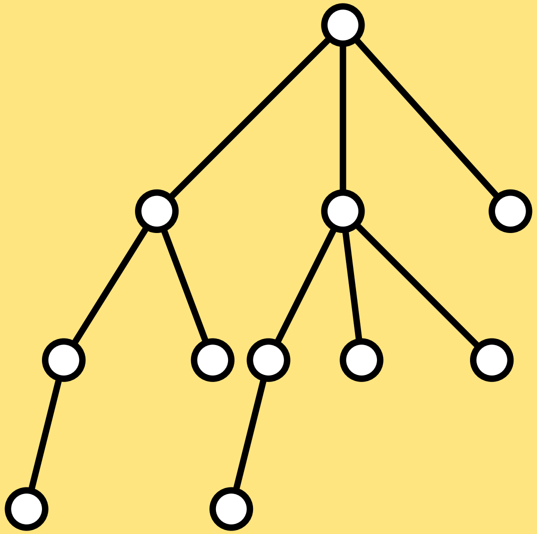
encode interblock explicitely: up to n edges $\Rightarrow$ space $n \log n$

encode $\leq m$-1 pioneers (outermost between blocks) $\Rightarrow$ space $m \log n$

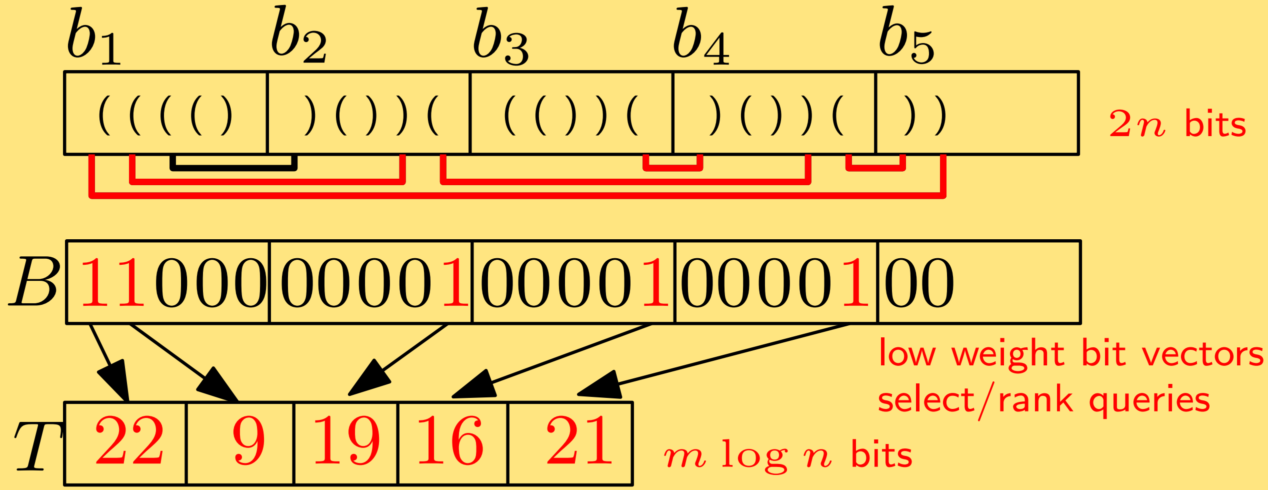
the explicit representation must allow navigation...

Starter: linear space data structures for plane trees

(Jacobson, Focs89)



Decompose into m small blocks of size ε



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 $n \log n$

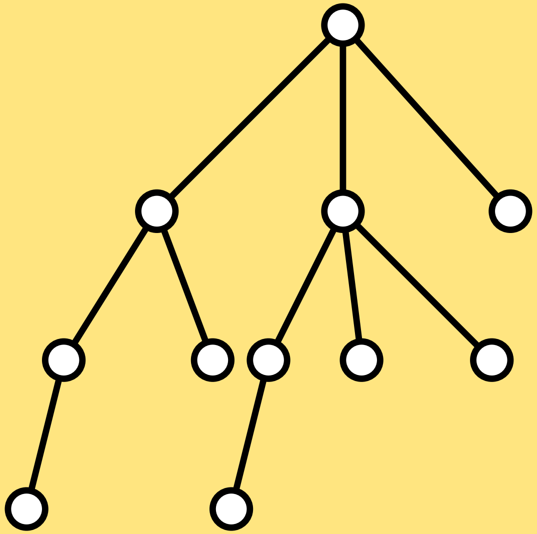
encode $\leq m$-1 pioneers (outermost between blocks) $\Rightarrow$ space $m \log n$

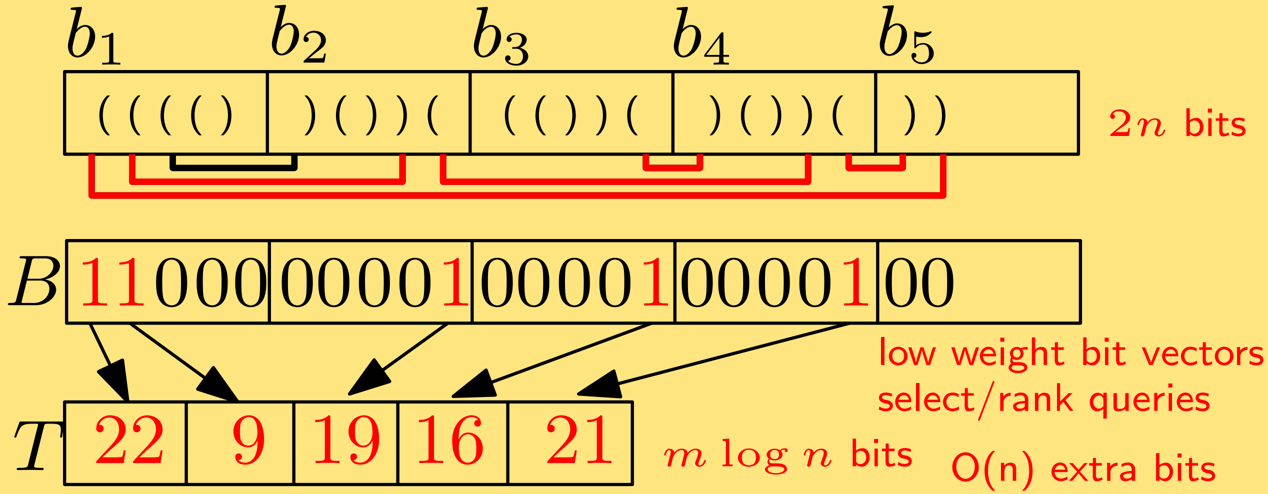
the explicit representation must allow navigation...

Starter: linear space data structures for plane trees

(Jacobson, Focs89)

Decompose into m small blocks of size ε





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

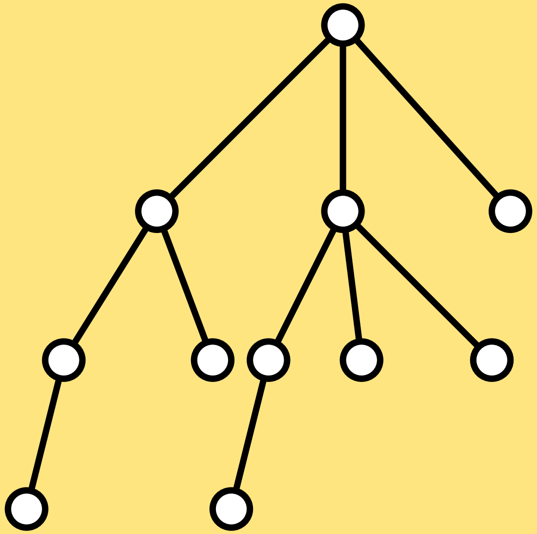
encode interblock explicitely: up to n edges $\Rightarrow$ space $n \log n$

encode $\leq m$-1 pioneers (outermost between blocks) $\Rightarrow$ space $m \log n$

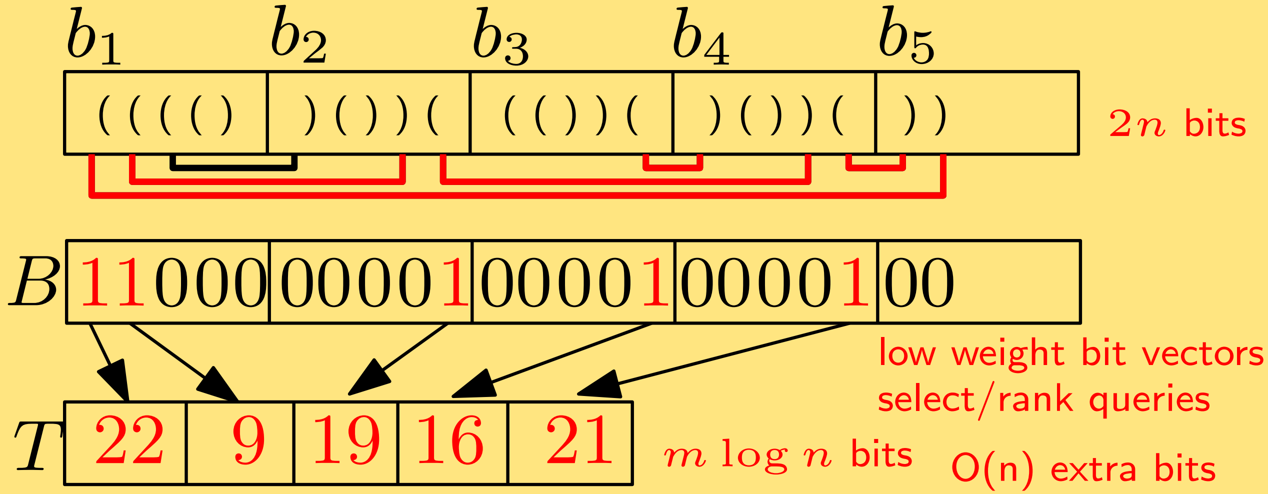
Taking $\varepsilon = \Theta(\log n)$: space $m \log n = O(n)$, queries in $O(\log n)$

Starter: linear space data structures for plane trees

(Jacobson, Focs89)



Decompose into m small blocks of size ε



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 $m \log n$

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 α bits per size unit

with linear time encoding/decoding procedures

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 α 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)

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 α 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

$\mathcal{A}_n$: 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 α 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 — entropy 2bpv

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

Entropy reduction and parametrized classes

ordered trees with n vertices	entropy 2bpv
degree 2 and 0 only: complete binary trees ($2n+1$ vertices: n nodes, $n+1$ leaves)	1bpv
degree 3 and 0 only: complete ternary ($3n+1$ vertices: n nodes, $2n+1$ leaves)	$\frac{1}{3}\log_2\frac{27}{2} \approx 1.25$ bpv

Entropy reduction and parametrized classes

ordered trees with n vertices — entropy 2bpv

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

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

more generally, n_i vertices of degree i

Entropy reduction and parametrized classes

ordered trees with n vertices — entropy 2bpv

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

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

more generally, n_i vertices of degree i

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

Entropy reduction and parametrized classes

ordered trees with n vertices — entropy 2bpv

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

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

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 $n = \sum n_i = 1 + \sum i n_i$

$\log_2 \binom{n}{n_0,n_1,\ldots,n_k}^{\frac{1}{n}}$

$\log_2 \prod_i \alpha_i^{-\alpha_i}$
if $n_i = \alpha_i n$

Entropy reduction and parametrized classes

ordered trees with n vertices — entropy 2bpv

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

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

more generally, n_i vertices of degree i

Old Thm: $|\mathcal{T}(n_0,\ldots,n_k)| = \frac{1}{n}\binom{n}{n_0,n_1,\ldots,n_k}$ — $\log_2\binom{n}{n_0,n_1,\ldots,n_k}^{\frac{1}{n}}$

if $n = \sum n_i = 1 + \sum i n_i$ — $\log_2 \prod_i \alpha_i^{-\alpha_i}$
if $n_i = \alpha_i n$

encode tree by degree list in prefix order

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

Entropy reduction and parametrized classes

ordered trees with n vertices — entropy 2bpv

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

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

more generally, n_i vertices of degree i — $\log_2 \binom{n}{n_0, n_1, \ldots, n_k}^{\frac{1}{n}}$

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

if $n = \sum n_i = 1 + \sum i n_i$

$\log_2 \prod_i \alpha_i^{-\alpha_i}$
if $n_i = \alpha_i n$

encode tree by degree list in prefix order

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

Question: what is the maximum entropy, for which degrees?

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes

given degree distribution

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes

given degree distribution $\sum \alpha_i \log_2 1/\alpha_i$

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
given degree distribution	$\sum \alpha_i \log_2 1/\alpha_i$	yes		

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
given degree distribution	$\sum \alpha_i \log_2 1/\alpha_i$	yes	yes (soda'07)	?

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
given degree distribution	$\sum \alpha_i \log_2 1/\alpha_i$	yes	yes (soda'07)	?
bipartite: p black, q white				

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
given degree distribution	$\sum \alpha_i \log_2 1/\alpha_i$	yes	yes (soda'07)	?
bipartite: p black, q white	4 if $p = \frac{n}{2} + O(\sqrt{n})$			

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
given degree distribution	$\sum \alpha_i \log_2 1/\alpha_i$	yes	yes (soda'07)	?
bipartite: p black, q white	4 if $p = \frac{n}{2} + O(\sqrt{n})$ otherwise $\binom{p+q}{p}^{\frac{2}{n}}$		use basic result	

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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	
	otherwise $\binom{p+q}{p}^{\frac{2}{n}}$	yes	probably	?

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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	
	otherwise $\binom{p+q}{p}^{\frac{2}{n}}$	yes	probably	?
height h				

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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	
	otherwise $\binom{p+q}{p}^{\frac{2}{n}}$	yes	probably	?
height h	known	?		

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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	
	otherwise $\binom{p+q}{p}^{\frac{2}{n}}$	yes	probably	?
height h	known	?		

positive natural embedding

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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		
	otherwise $\binom{p+q}{p}^{\frac{2}{n}}$	yes	probably	?
height h	known	?		
positive natural embedding	4	use basic result		

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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	
	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				

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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		
	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

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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		
	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?	?		

ordinary decomposable structures

(multitype ordered trees)

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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	
	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?	?		

ordinary decomposable structures

(multitype ordered trees) computable ? use frequecies ?

link with multivariable Lagrange inversion?

Entropy quizz

	entropy	compression	succinct d.s.	dynamic
ordered trees	4	yes	yes	yes
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	
	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?	?		

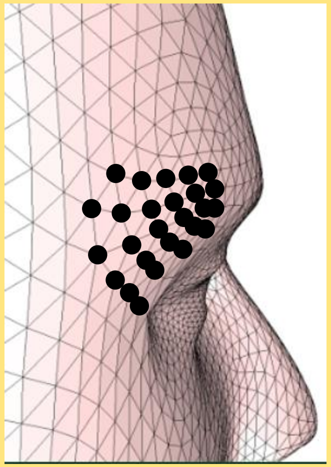
ordinary decomposable structures (multitype ordered trees) computable ? use frequecies ?

link with multivariable Lagrange inversion?

entropy measures diversity of local structure

Geometric information vs Combinatorial information

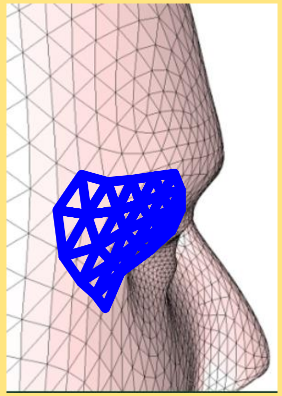
Geometry

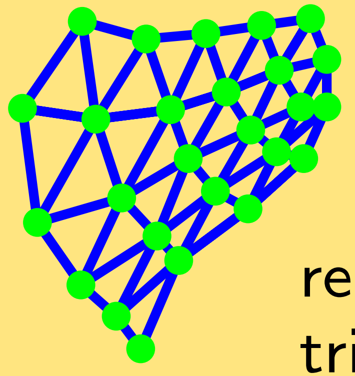


vertex
coordinates

between 30 et 96 bits/vertex

"Connectivity": the underlying triangulation





adjacency
relations between
triangles, vertices

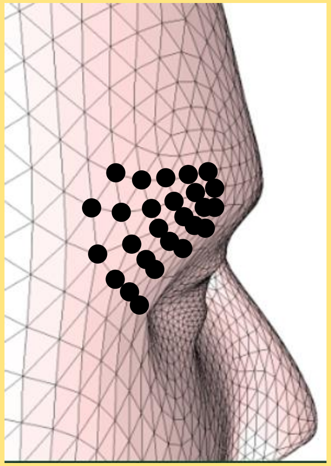
vertex 1 reference to a triangle

triangle 3 references to vertices
3 references to triangles

$13n \log n$ or $416n$ bits

Geometric information vs Combinatorial information

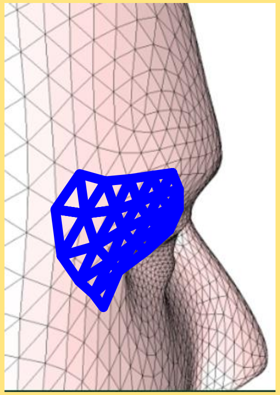
Geometry

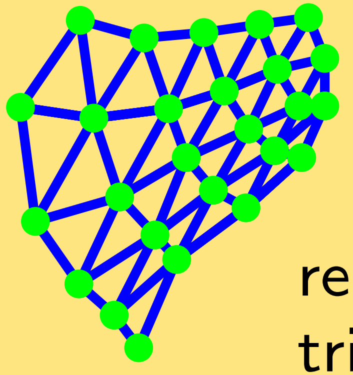


vertex coordinates

between 30 et 96 bits/vertex

"Connectivity": the underlying triangulation





adjacency relations between triangles, vertices

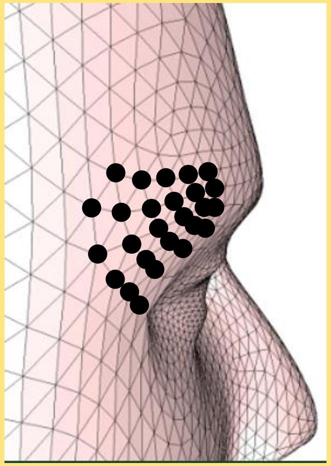
vertex	1 reference to a triangle
triangle	3 references to vertices 3 references to triangles

$13n \log n$ or $416n$ bits

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

Geometric information vs Combinatorial information

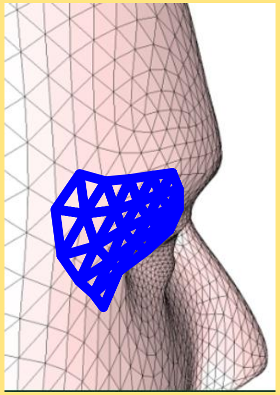
Geometry

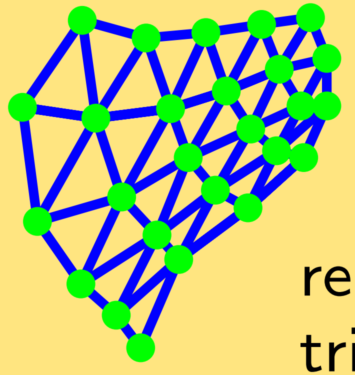


vertex coordinates

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

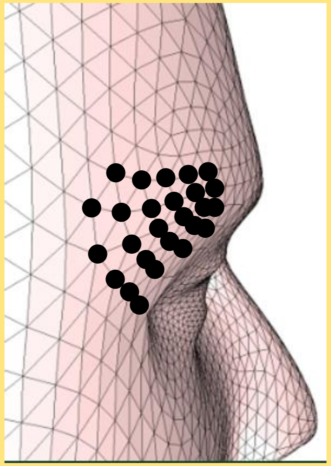
$13n \log n$ or $416n$ bits

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

$$\Rightarrow \quad \text{entropy} = \log_2 \frac{256}{27} \approx 3.24 \text{ bpv.}$$

Geometric information vs Combinatorial information

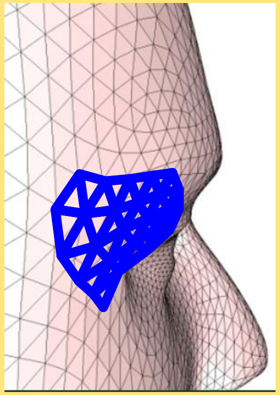
Geometry

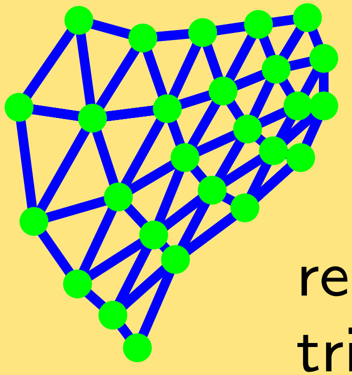


vertex coordinates

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

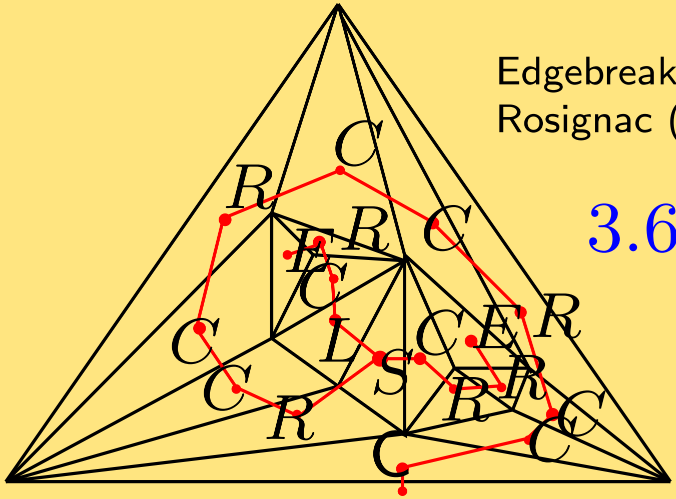
$13n \log n$ or $416n$ bits

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

$\Rightarrow$ entropy $= \log_2 \frac{256}{27} \approx 3.24$ bpv. Room for improvement!

Triangulation encodings: trees decompositions

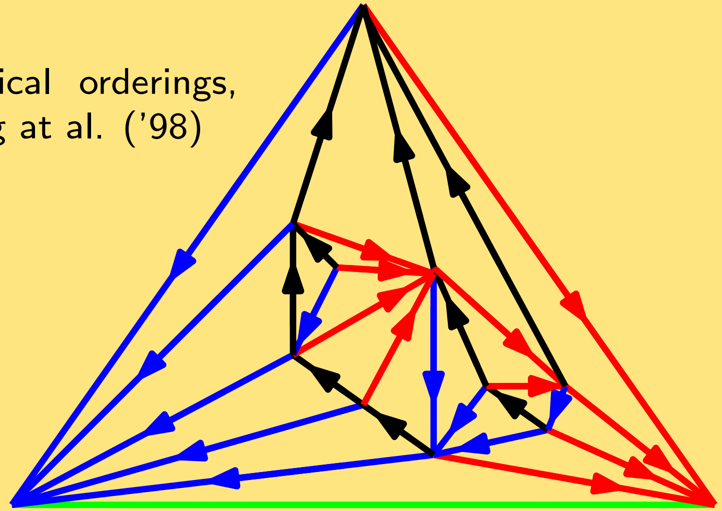
Common visual framework (Isenburg Snoeyink'05)



Edgebreaker,
Rosignac ('99)

$3.67n$

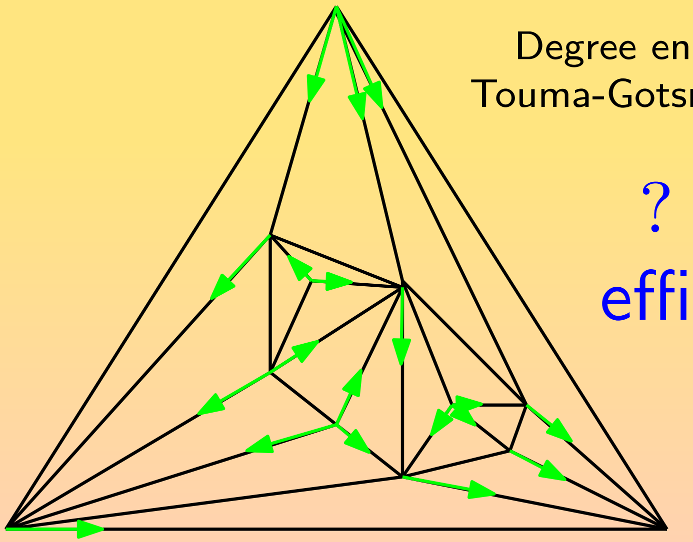
$CCCRCCRCCRECRRELCRE$



Canonical orderings,
Chiang at al. ('98)

$4n$

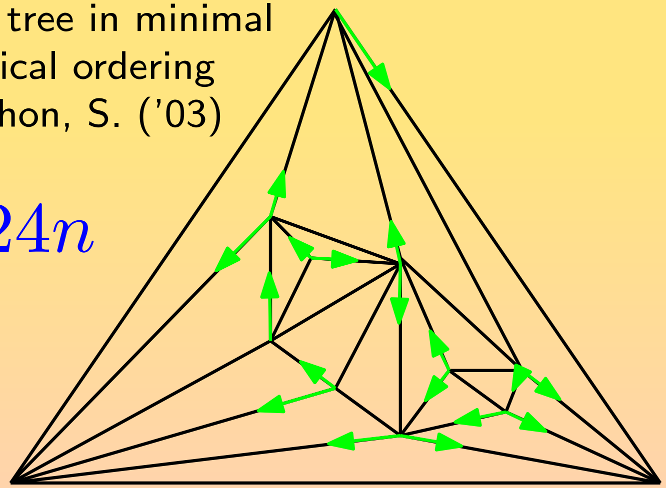
([[[ ) ( ] ( ] ( ] [[[ ) [ ) ( ]] [ ) . . .



Degree encoding,
Touma-Gotsman ('98)

? but
efficient

$V_5V_5V_6V_5V_4V_5V_8V_5V_5V_4S_4V_3V_4$



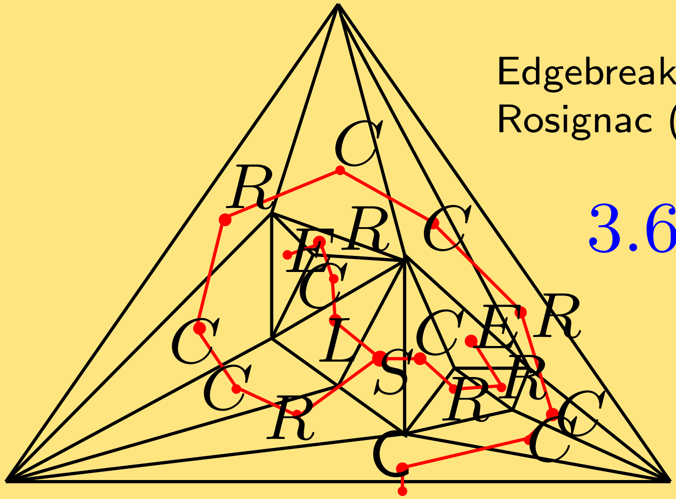
Leftmost tree in minimal
canonical ordering
Poulalhon, S. ('03)

$3.24n$

110100011000001001000001100100000000

Triangulation encodings: trees decompositions

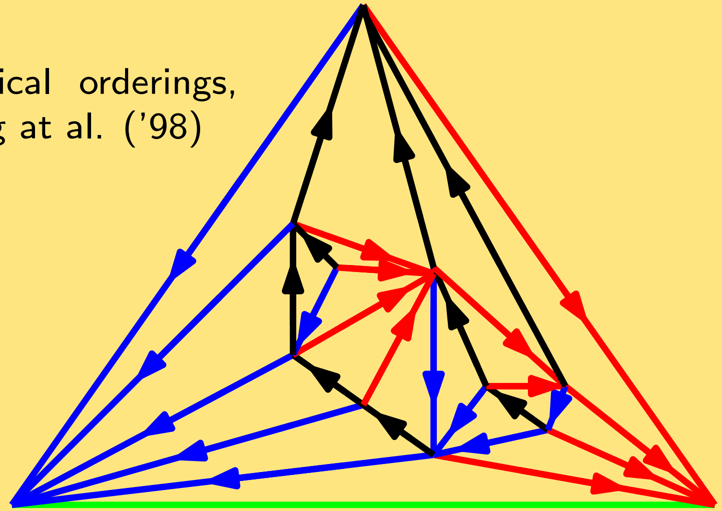
Common visual framework (Isenburg Snoeyink'05)



Edgebreaker, Rosignac ('99)

$3.67n$

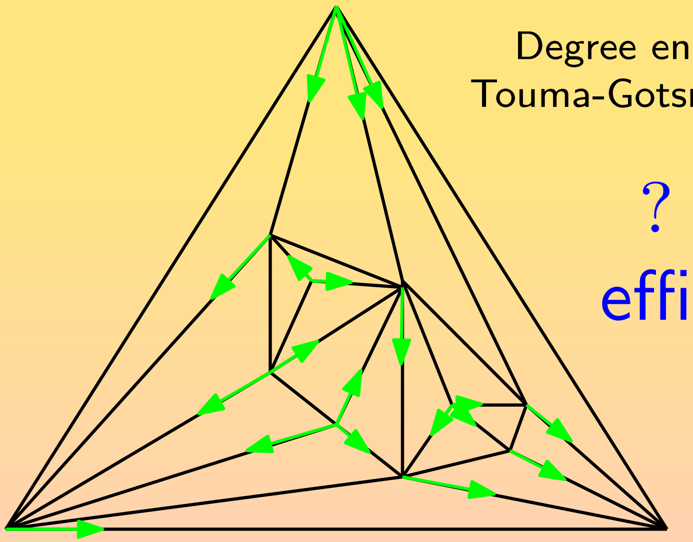
$CCCRCCRCCRECRRELCRE$



Canonical orderings, Chiang at al. ('98)

$4n$

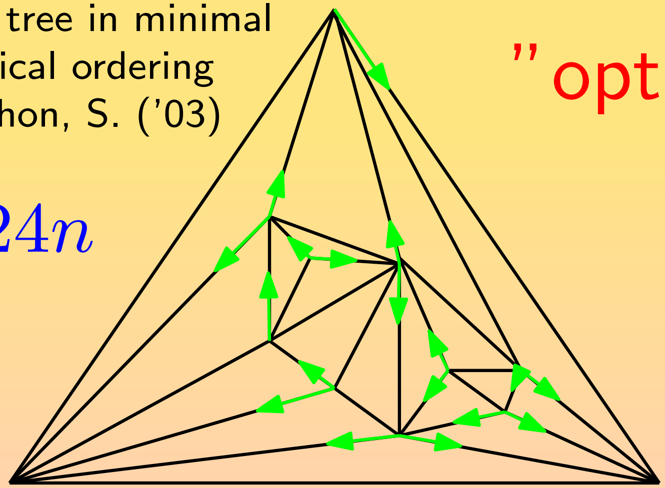
([[[ ) ( ] ( ] ( ] [[[ ) [ ) ( ]] [ ) . . .



Degree encoding, Touma-Gotsman ('98)

? but efficient

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



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

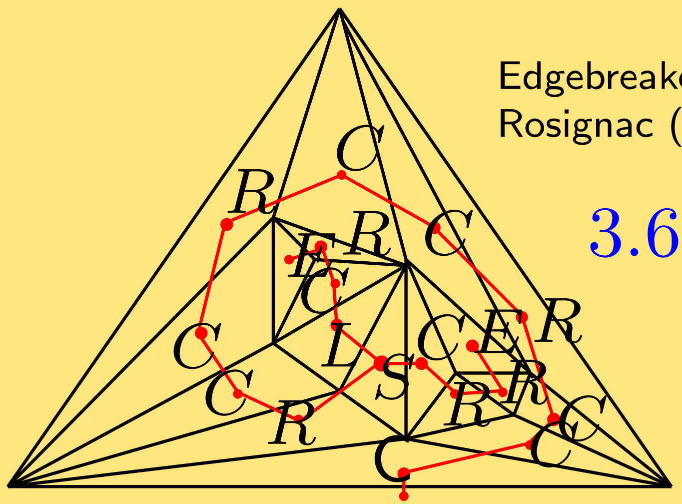
$3.24n$

"optimal"

110100011000001001000001100100000000

Triangulation encodings: trees decompositions

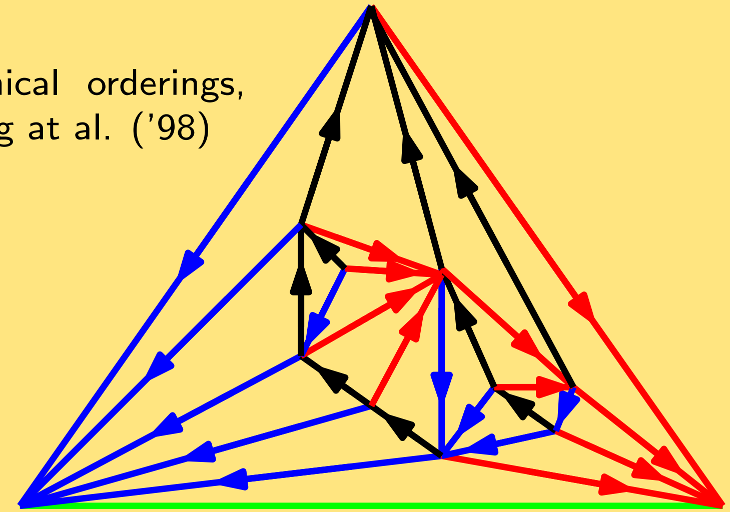
Common visual framework (Isenburg Snoeyink'05)



Edgebreaker,
Rosignac ('99)

$3.67n$

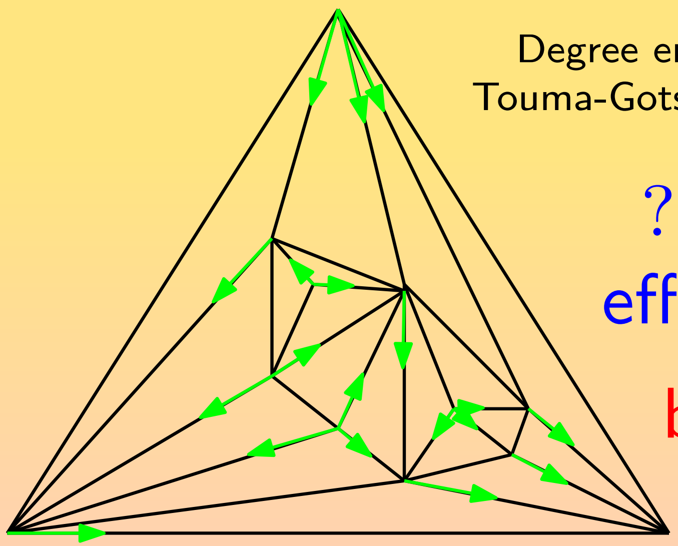
$CCCRCCRCCRECRRELCRE$



Canonical orderings,
Chiang at al. ('98)

$4n$

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

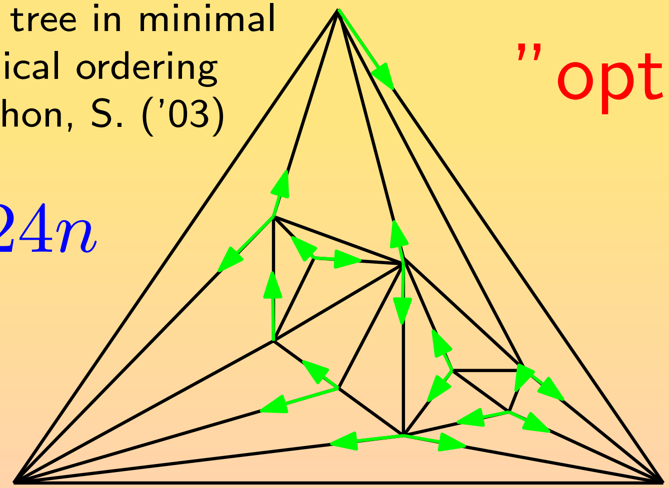


Degree encoding,
Touma-Gotsman ('98)

? but
efficient

better?!

$V_5V_5V_6V_5V_4V_5V_8V_5V_5V_4S_4V_3V_4$



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

$3.24n$

"optimal"

11010001100000100100000110010000000000

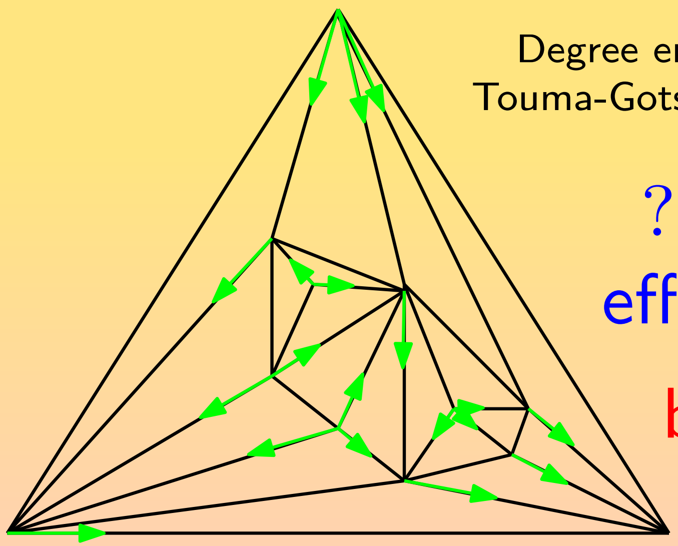
Triangulation encodings: trees decompositions

Common visual framework (Isenburg Snoeyink'05)

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

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

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



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

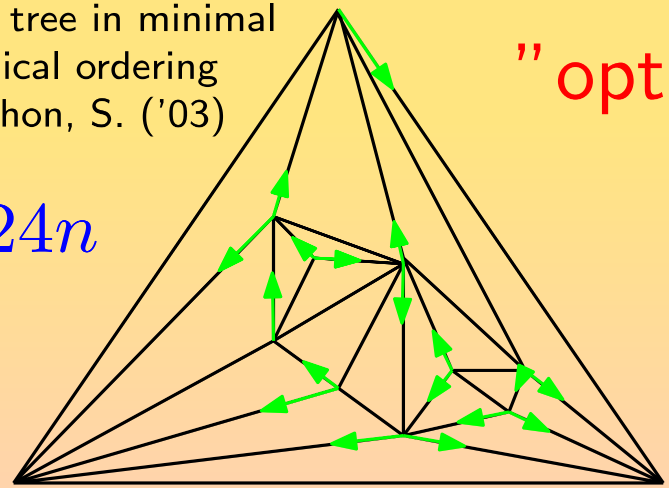
Degree encoding,
Touma-Gotsman ('98)

? but efficient

better?!

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

$3.24n$



110100011000001001000001100100000000

"optimal"

Triangulation encodings: trees decompositions

Common visual framework (Isenburg Snoeyink'05)

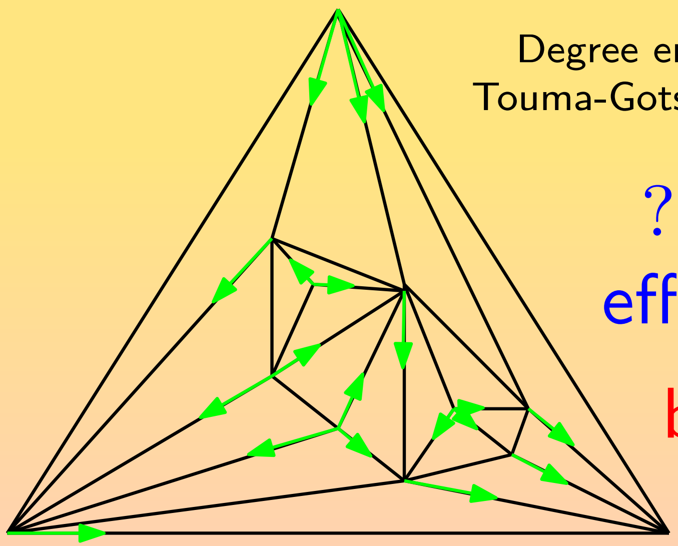
The (non-optimal) degree encoder gives much better codes for low entropy triangulations!

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

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 **bpv** < 3.245 **bpv!**



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

Degree encoding, Touma-Gotsman ('98)

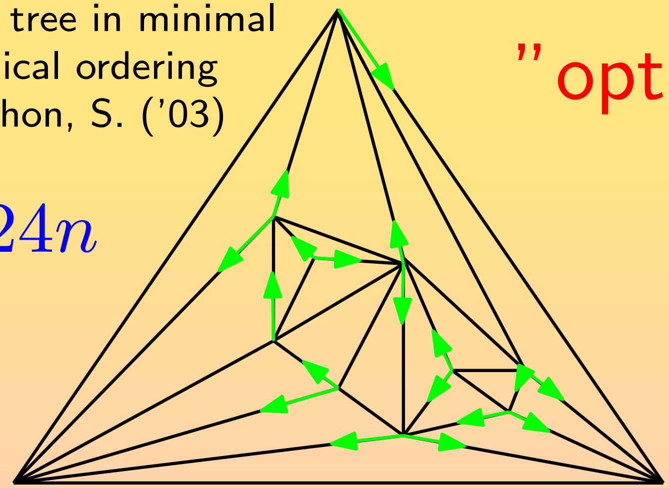
? but efficient

better?!

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

$3.24n$

"optimal"



1101000110000001001000000110010000000000

Mesh compression

Computer graphics

Edgebreaker

Rossignac ('99)

Lope et al. ('03)

Lewiner et al. ('04)

. . . . . . (many many others)

Valence (degree)

Touma and Gotsman ('98)

Alliez and Debrun

Isenburg

Khodakovsky

. . . . . . (many others)

Cut − border machine

Gumhold et al. (Siggraph '98)

Gumhold (Soda '05)

Graph encoding

Graph theory / combinatorics

Turan ('84)

Keeler Westbrook ('95)

He et al. ('99)

Chuang et al. (Icalp98)

Poulalhon S.(Icalp03)

Fusy et al. (Soda05)

Castelli Aleardi, Fusy, Lewiner (SoCG08)

Succinct representations

Algorithms and DS

Jacobson (Focs89)

Munro and Raman (Focs97)

Chiang et al. (Soda01)

Castelli Aleardi, Devillers and S. (Wads05, CCCG05, SoCG06)

Barbay et al. (Isaac07)

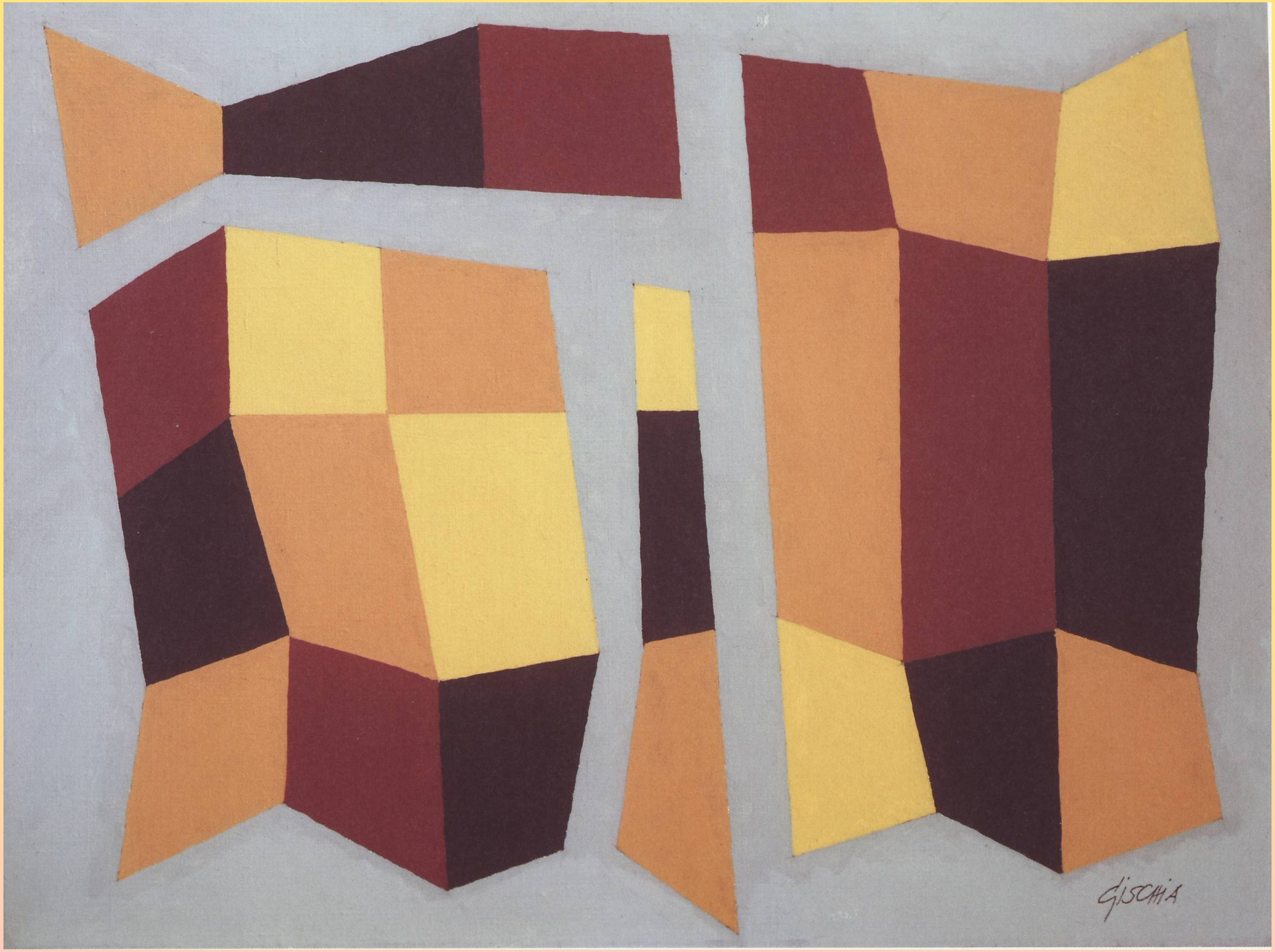
Nakano et al. (2008)

Blandford Blelloch (Soda03)

A more generic approach?

First idea (following Luca Castelli Aleardi)

Decomposition of quadrangulations...by the french artist Léon Gischia (1903-1991)



2nd idea (following Luca Castelli Aleardi)

Literary digression (La leçon, Eugène Ionesco, 1951)

During a private lesson, a very young student, preparing herself for the total doctorate, talks about arithmetics with her teacher

(the young student cannot understand how to subtract integers)

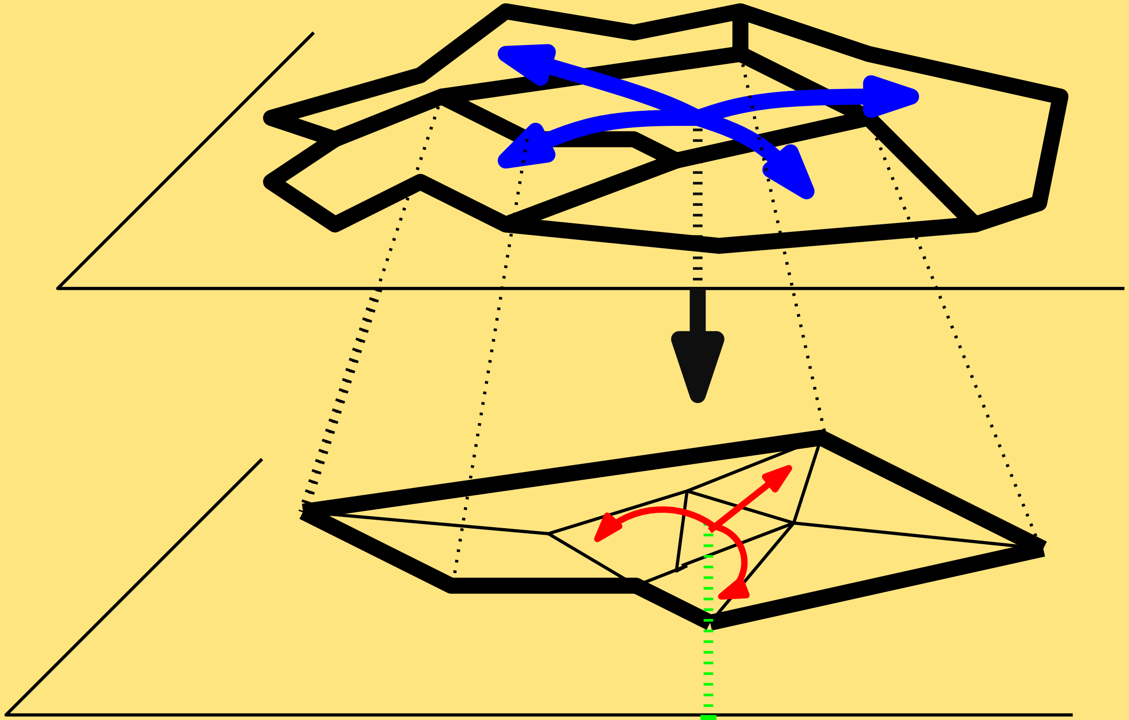
Teacher Listen to me, If you cannot deeply understand these principles, these arithmetic archetypes, you will never perform correctly a "polytechnicien" job... you will never obtain a teaching position at "Ecole Polytechnique". For example, what is $3.755.918.261$ multiplied by $5.162.303.508$?

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.



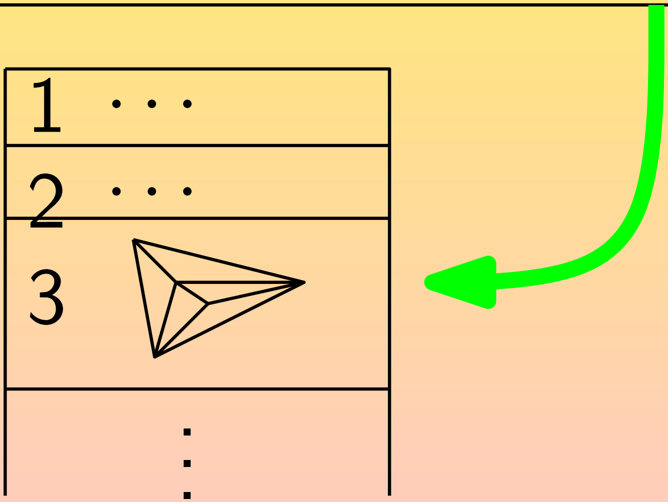
Level 1:

- $\Theta(\frac{n}{\log^2 n})$ 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

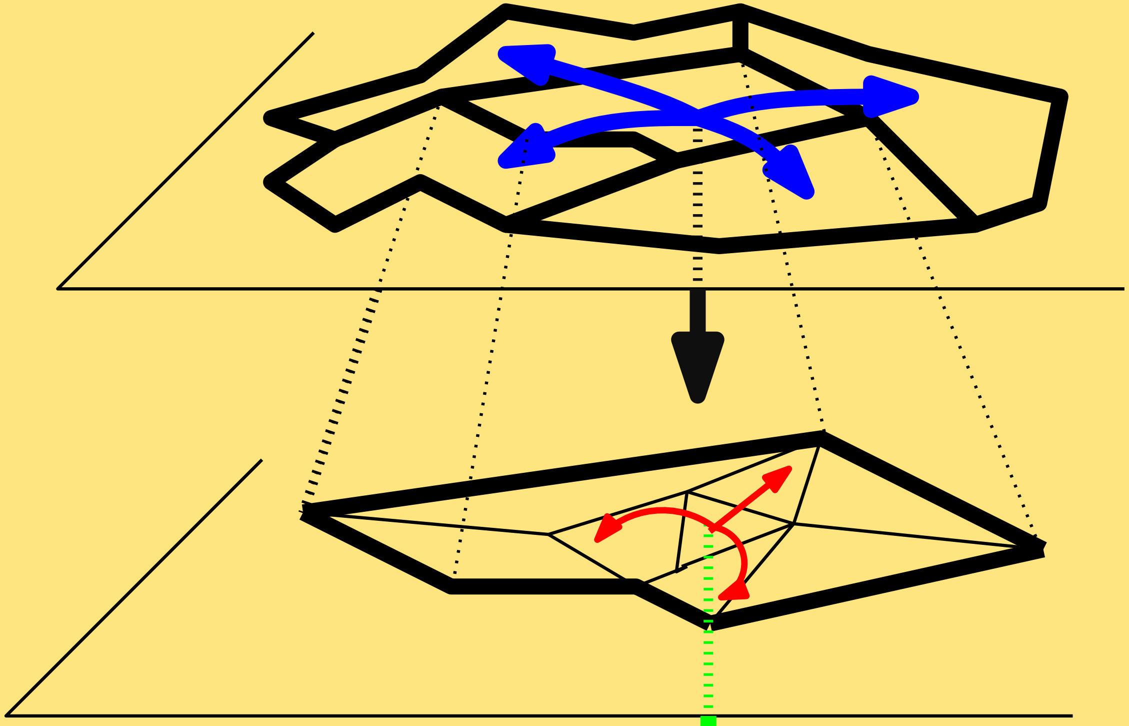
- $\Theta(\log n)$ regions of size $C \log n$, represented by pointers to level 3



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.

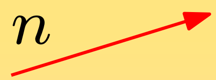


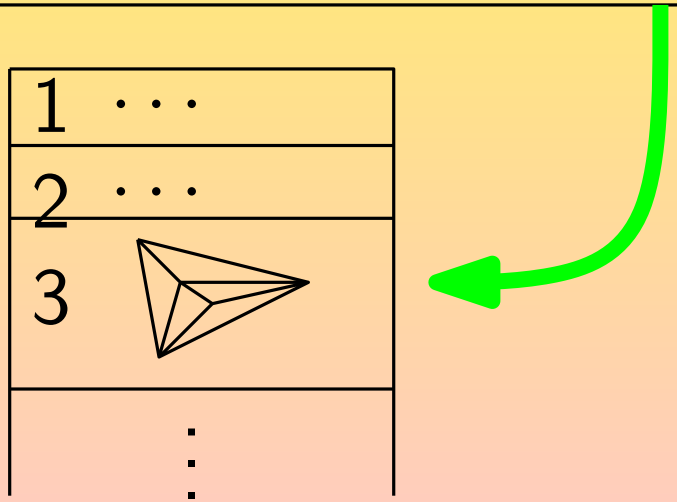
Level 1:

- $\Theta(\frac{n}{\log^2 n})$ 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

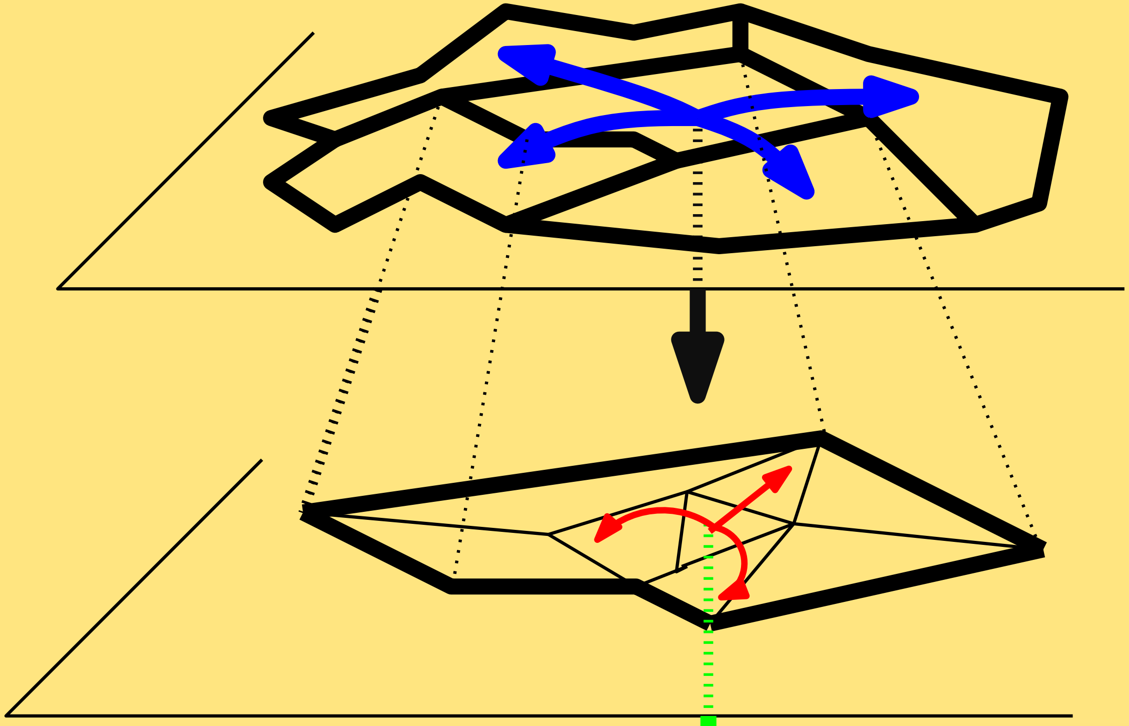
- $\Theta(\log n)$ regions of size $C \log n$,
  represented by pointers to level 3
- local pointers of size $\log \log n$ 



Level 3: exhaustive catalog of all different regions of size $i < C \log n$:

- complete explicit representation.

A hierarchical approach, with a dictionary at bottom.



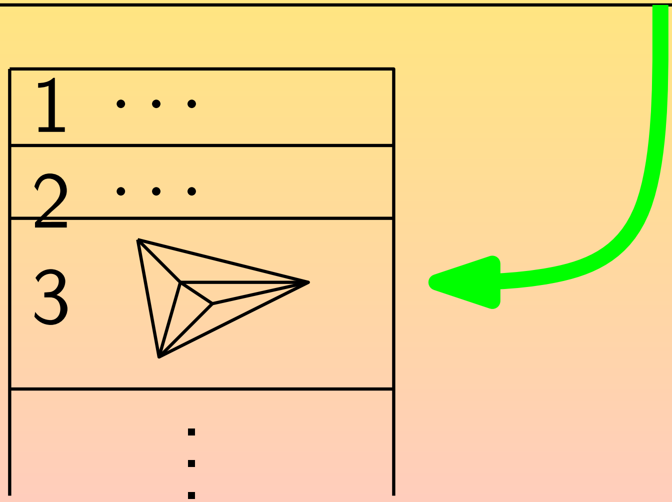
Level 1:

- $\Theta(\frac{n}{\log^2 n})$ 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 $\log \log n$ 

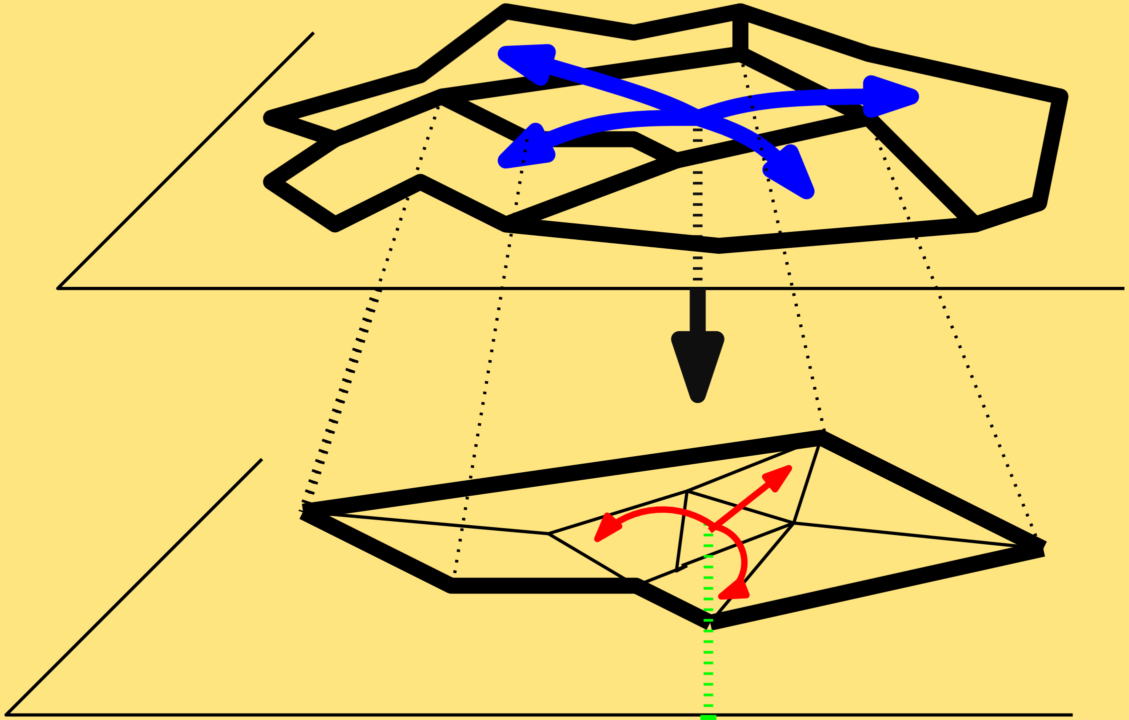


Level 3: exhaustive catalog of all different regions of size $i < C \log n$:

- complete explicit representation.

Dictionnary space is $o(n)$ if C small enough.

A hierarchical approach, with a dictionary at bottom.



Level 1:

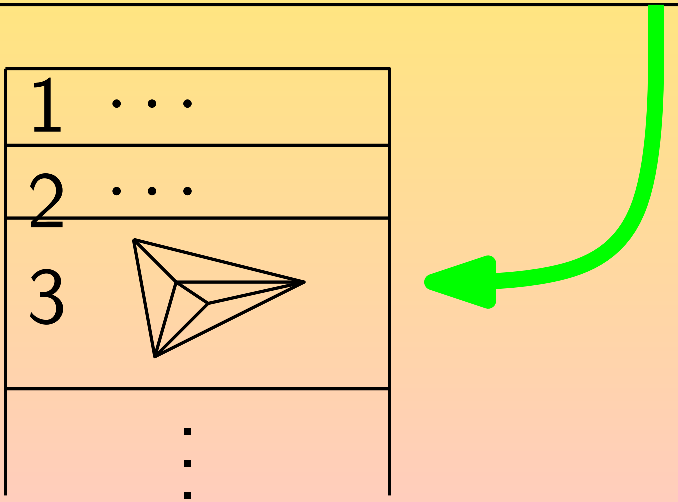
- $\Theta(\frac{n}{\log^2 n})$ regions of size $\Theta(\log^2 n)$, represented by pointers to level 2
- global pointers of size $\log n$

space $O(\frac{n}{\log^2 n} \cdot \log n) = o(n)$

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 $\log \log n$

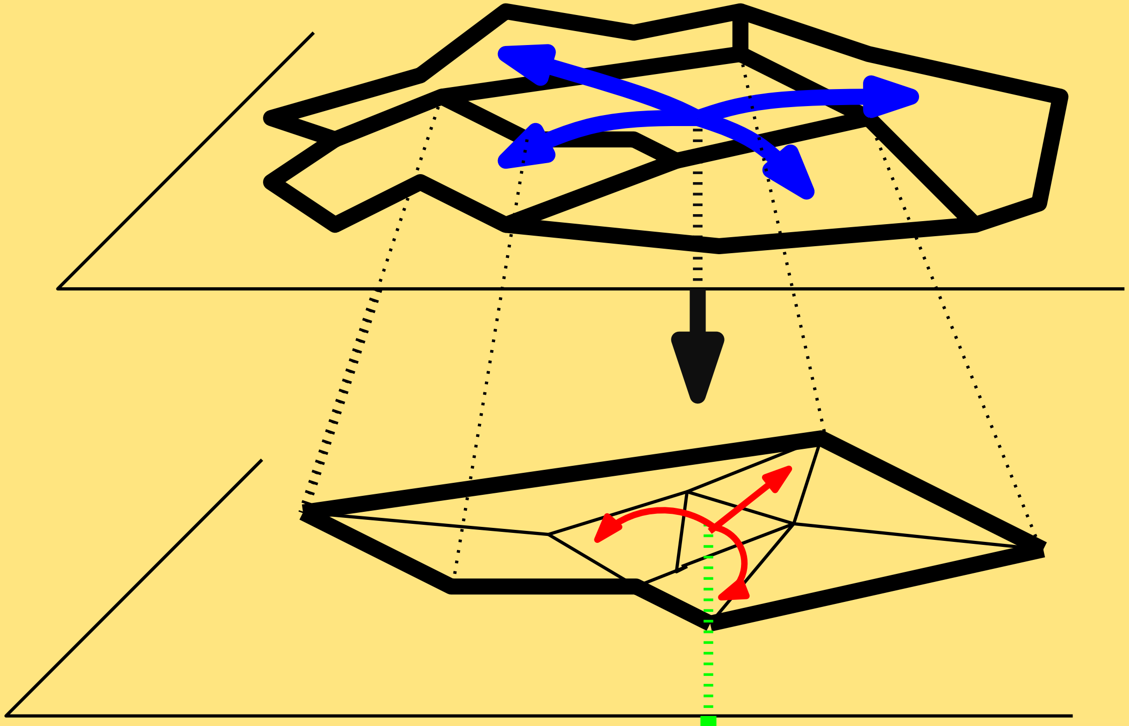


Level 3: exhaustive catalog of all different regions of size $i < C \log n$:

- complete explicit representation.

Dictionnary space is $o(n)$ if C small enough.

A hierarchical approach, with a dictionary at bottom.



Level 1:

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

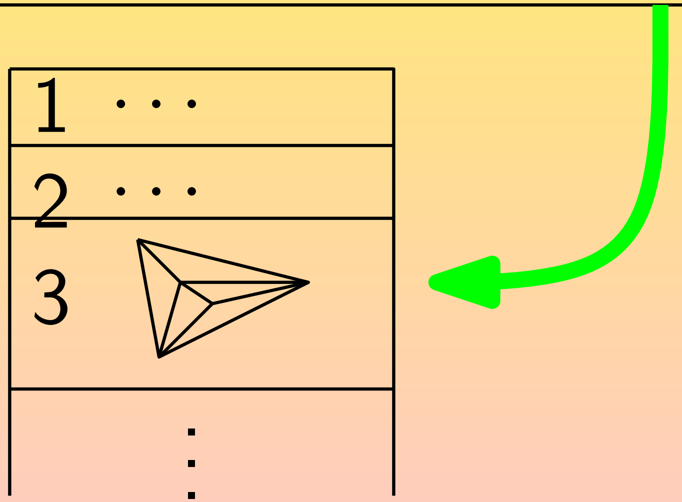
space $O(\frac{n}{\log^2 n} \cdot \log n) = o(n)$

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



Level 3: exhaustive catalog of all different regions of size $i < C \log n$:

- complete explicit representation.

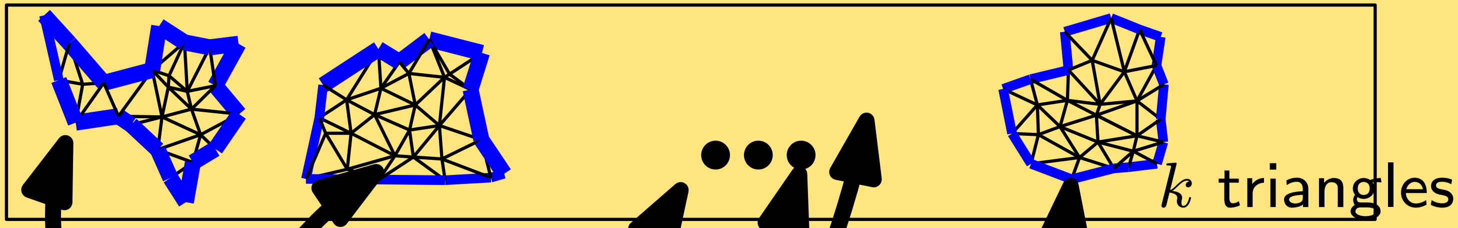
Dictionnary space is $o(n)$ if C small enough.

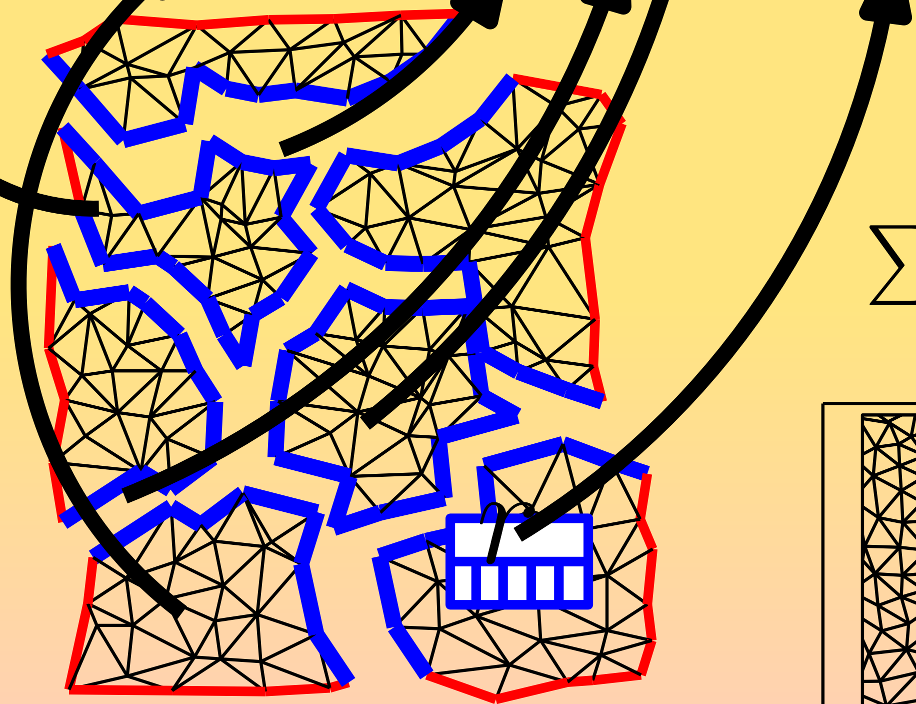
A hierarchical approach, with a dictionary at bottom.

Dominant term?

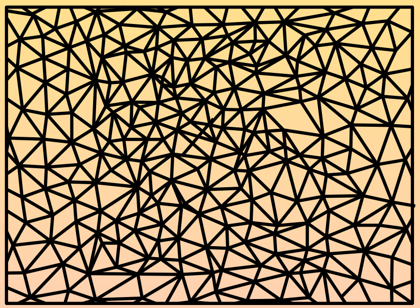
The dominant term is given by the sum of references to the dictionary

references on objects of $\mathcal{T}_k$ have size $\log_2 \mathcal{T}_k \sim 2.175k$ if $k \to \infty$





$\sum_j 2.175k_j = 2.175m$ bits



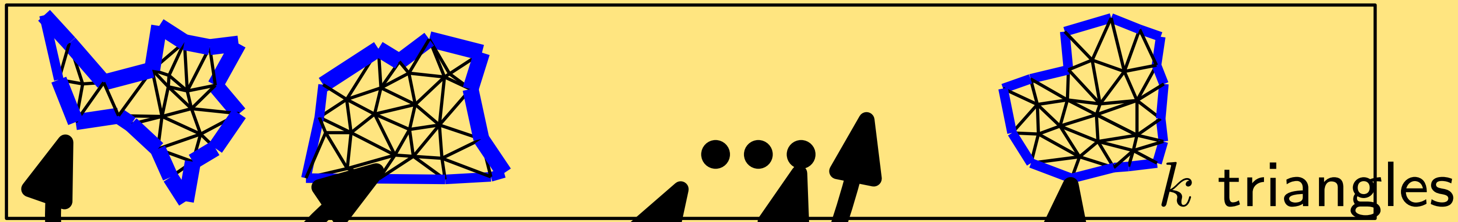
2.175bpt is entropy of triangulations **with a boundary**

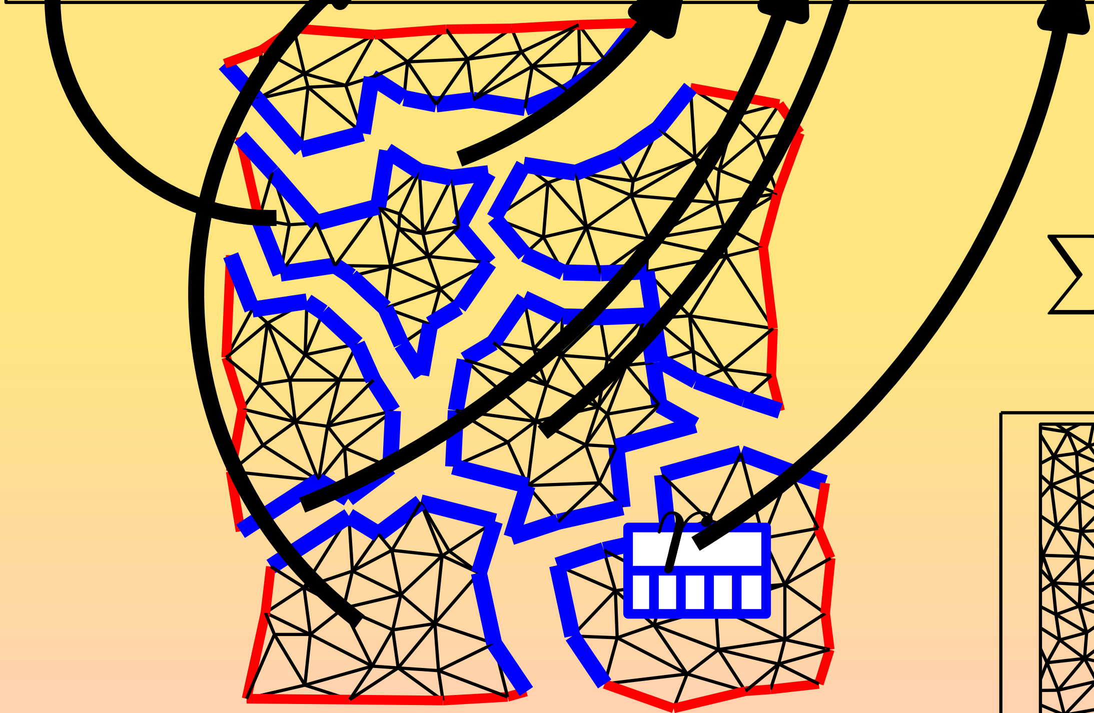
A hierarchical approach, with a dictionary at bottom.

Dominant term?

The dominant term is given by the sum of references to the dictionary

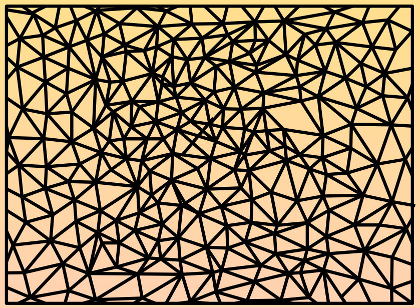
references on objects of $\mathcal{T}_k$ have size $\log_2 \mathcal{T}_k \sim 2.175k$ if $k \to \infty$





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

$\sum_j 2.175k_j = 2.175m$ bits



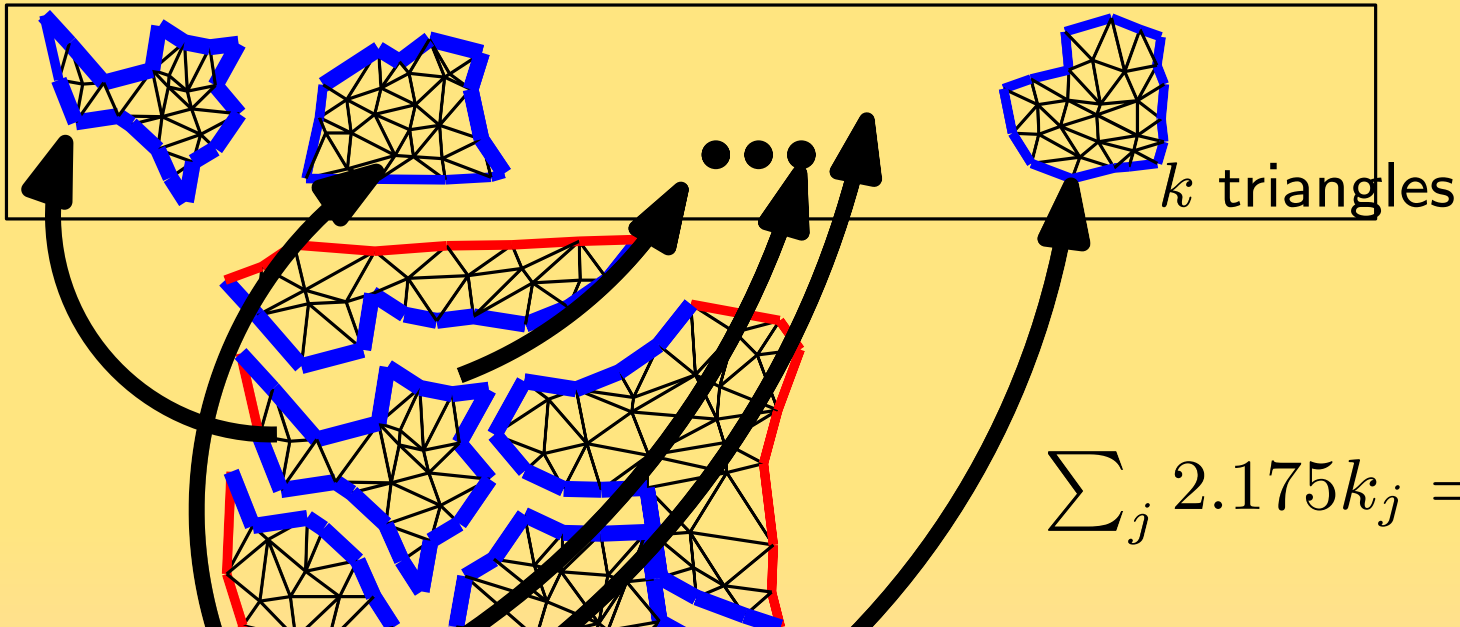
2.175bpt is entropy of triangulations **with a boundary**

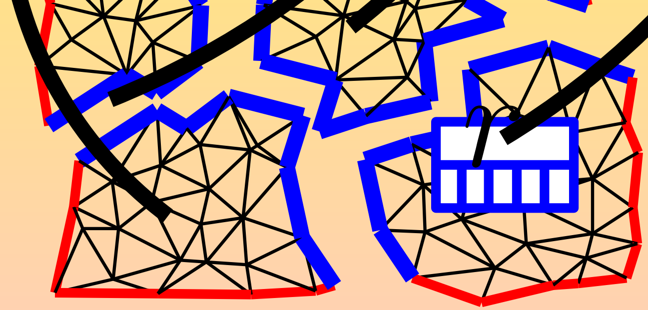
A hierarchical approach, with a dictionary at bottom.

Dominant term?

The dominant term is given by the sum of references to the dictionary

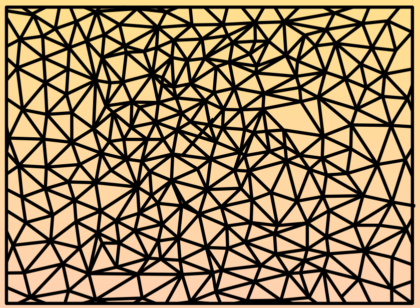
references on objects of $\mathcal{T}_k$ have size $\log_2 \mathcal{T}_k \sim 2.175k$ if $k \to \infty$





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

$\sum_j 2.175k_j = 2.175m$ bits



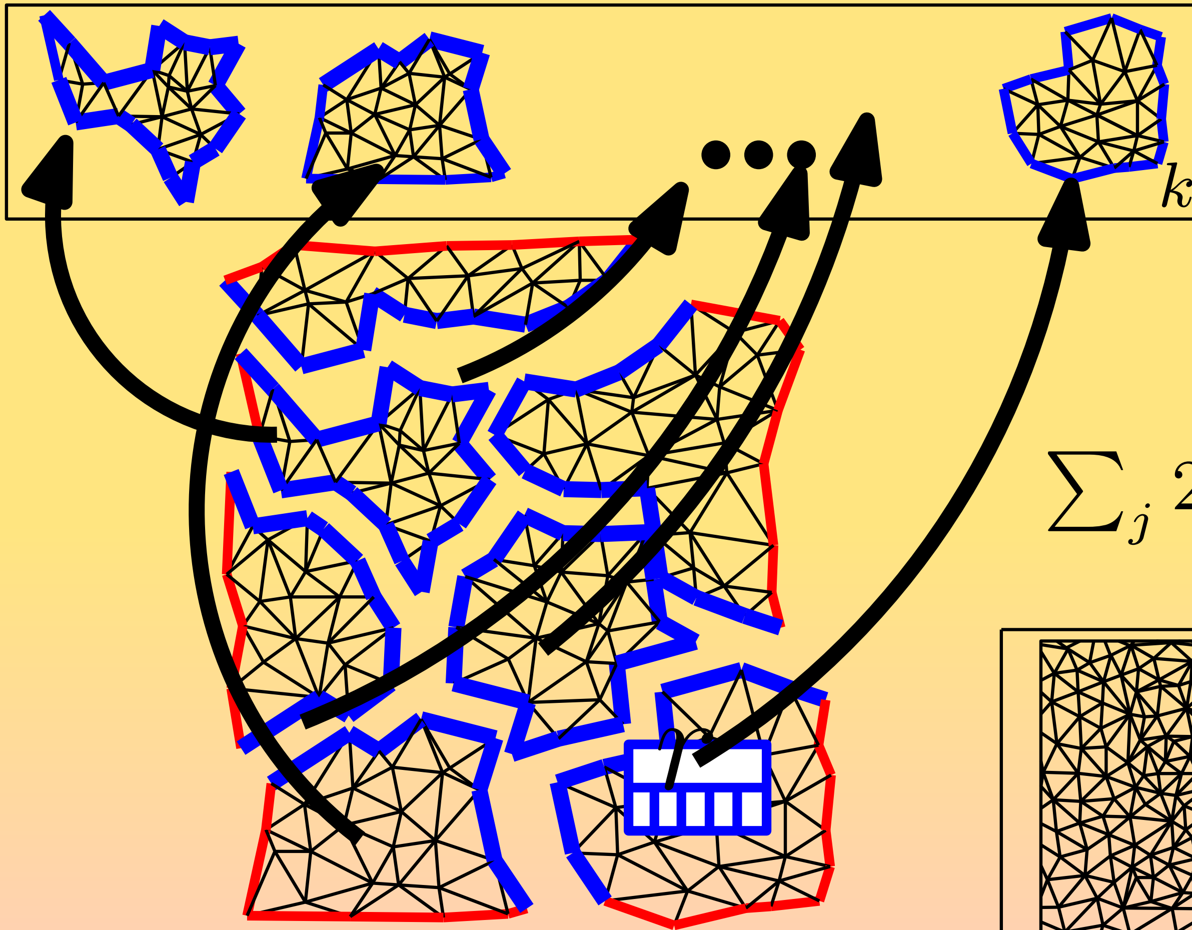
2.175bpt is entropy of triangulations **with a boundary** larger than previous $\frac{1}{2} \cdot 3.24$bpt

A hierarchical approach, with a dictionary at bottom.

Dominant term?

The dominant term is given by the sum of references to the dictionary

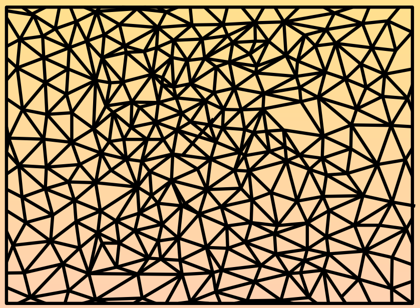
references on objects of $\mathcal{T}_k$ have size $\log_2 \mathcal{T}_k \sim 2.175k$ if $k \to \infty$



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

Adaptative to "reasonable" entropy reduction

$\sum_j 2.175k_j = 2.175m$ bits



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

A 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...

  $\Rightarrow$ 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

  $\Rightarrow$ find an optimal adaptative encoder for triangulations with given degrees

  $\Rightarrow$ find other parameters of trees or maps that allow for simple adaptative compression or sds (depth?)