Coding, counting, and sampling triangulations and other planar graphs

Gilles SCHAEFFER

CNRS, École Polytechnique

An overview of the talk

I. 3-c planar graphs

II. Binary trees and a combinatorial approach

III. From trees to dissections, counting and sampling.

IV. Minimal α-orientations, coding.

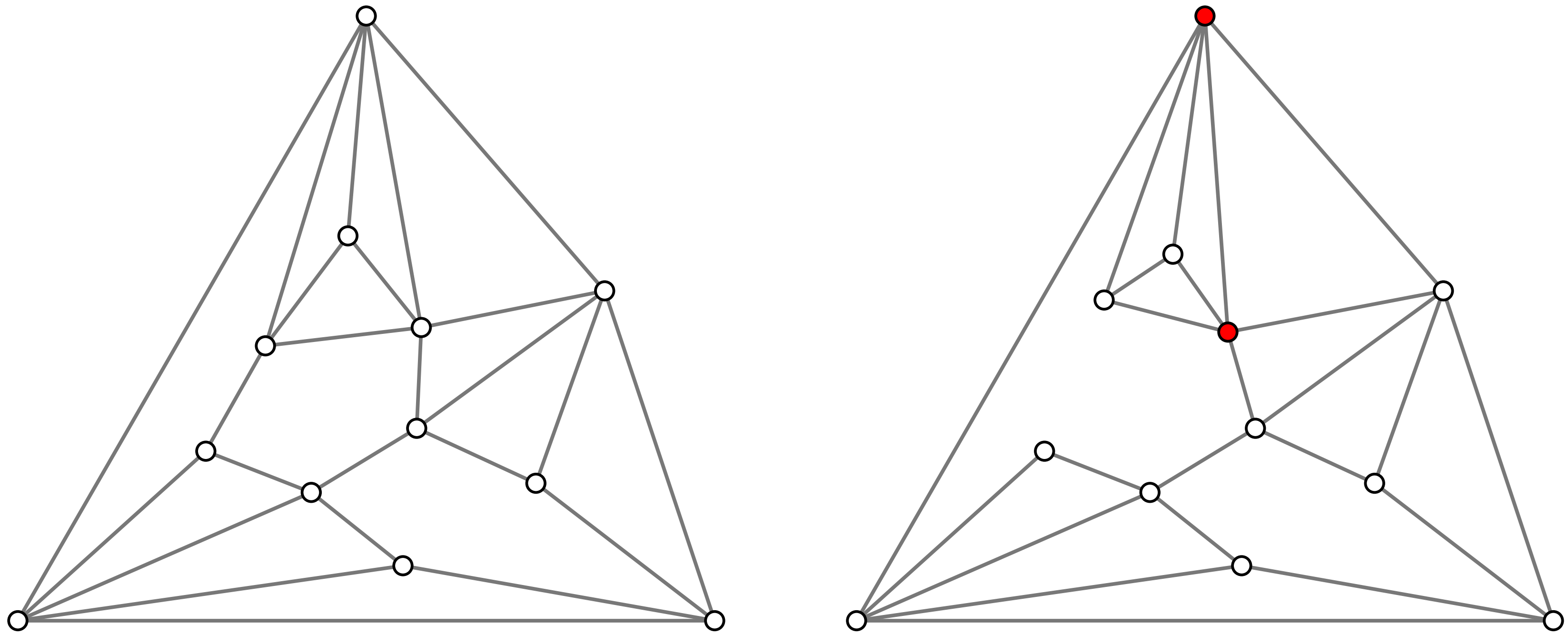
V. Trees and orientations everywhere.

Part 1. Some combinatorial structures.

CASTING for this part :

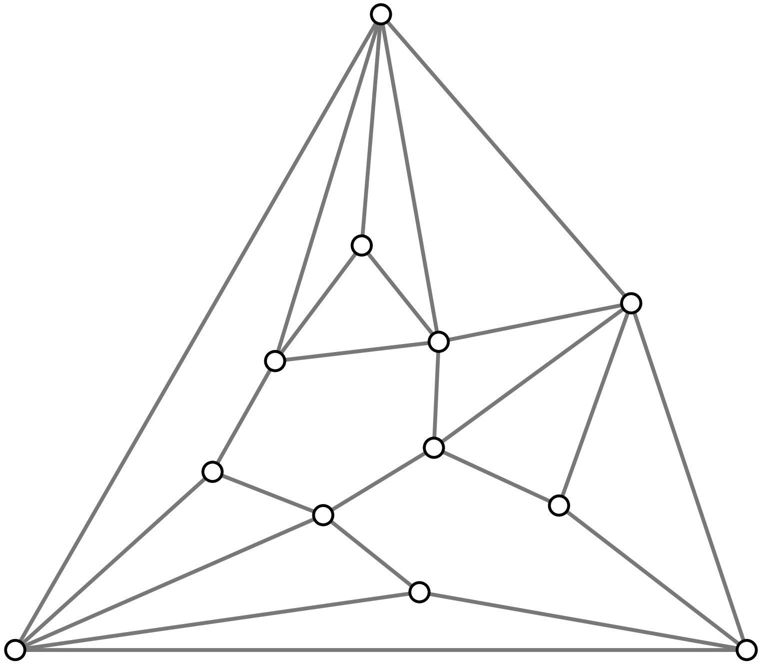
- 3-connected planar graphs
- polyhedral graphs, irreducible dissections
- lion, triceratops

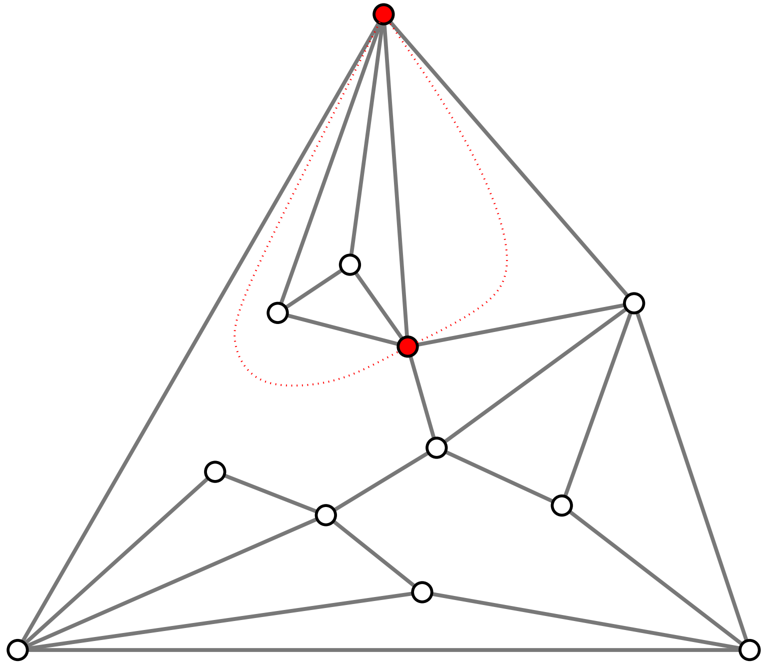
3-CONNECTED PLANAR GRAPHS



A planar graph is 3-connected if there is no 2-separator.

3-CONNECTED PLANAR GRAPHS



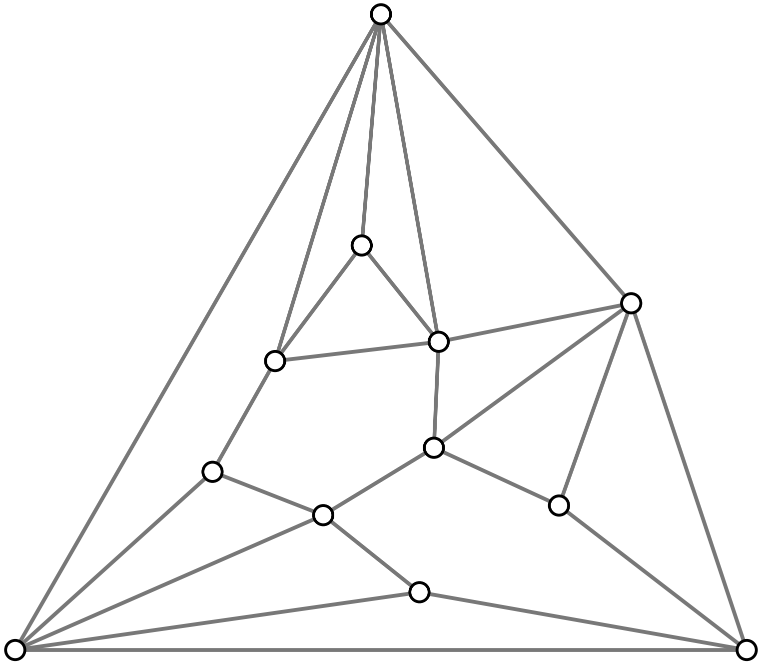


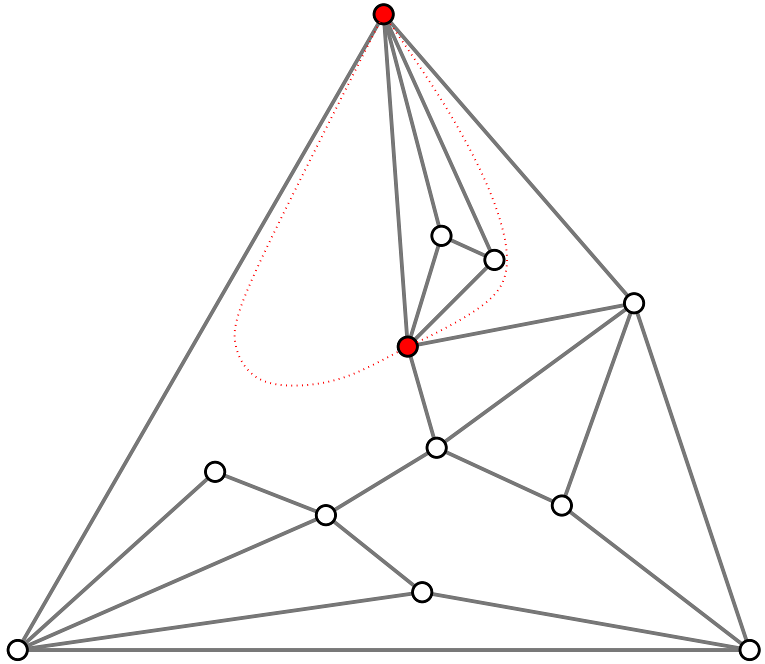
A planar graph is 3-connected if there is no 2-separator.

Whitney : 3-connected planar graphs have a unique embedding up to homeomorphisms of the non-oriented sphere, i.e. a unique planar map.

2-separators give raise to different maps for the same graph

3-CONNECTED PLANAR GRAPHS



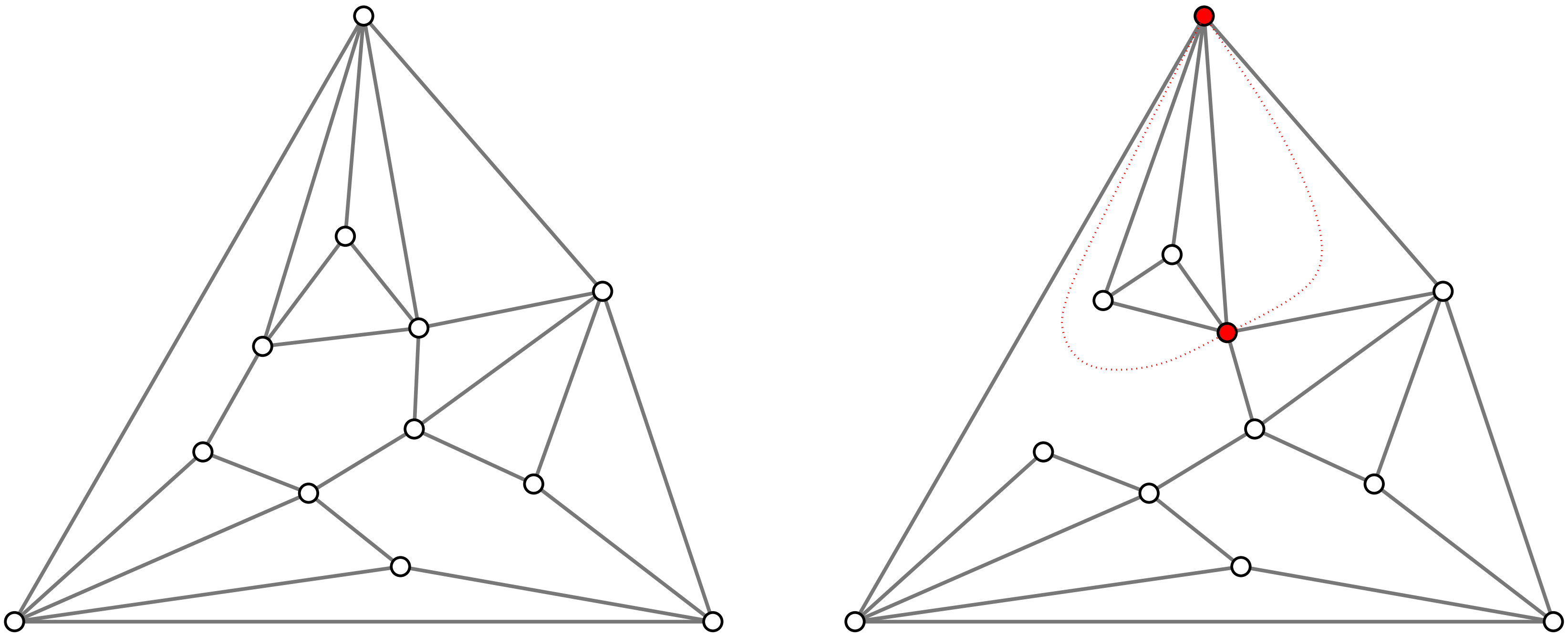


A planar graph is 3-connected if there is no 2-separator.

Whitney : 3-connected planar graphs have a unique embedding up to homeomorphisms of the non-oriented sphere, i.e. a unique planar map.

2-separators give raise to different maps for the same graph

3-CONNECTED PLANAR GRAPHS

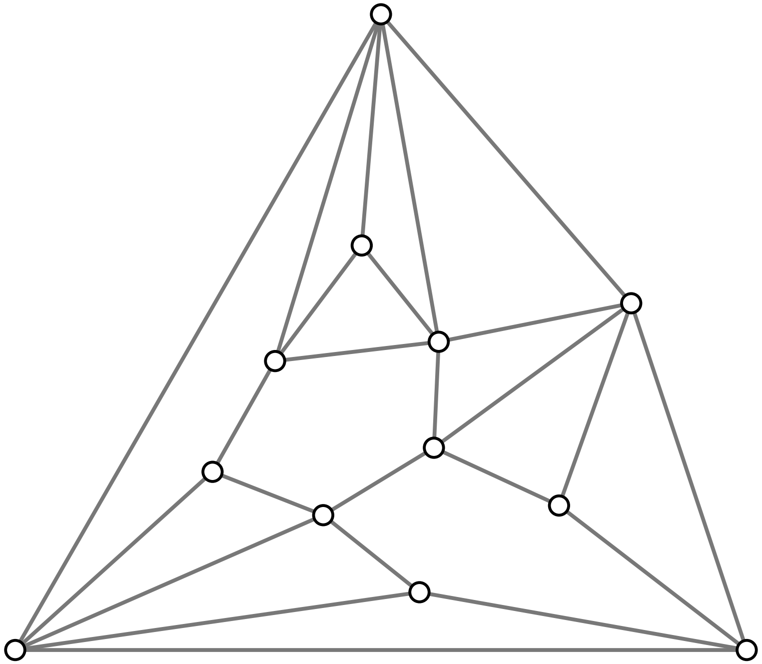


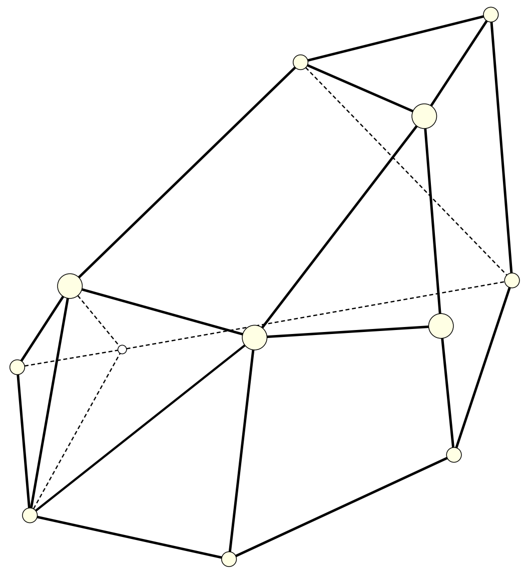
A planar graph is 3-connected if there is no 2-separator.

Whitney : 3-connected planar graphs have a unique embedding up to homeomorphisms of the non-oriented sphere, i.e. a unique planar map.

2-separators give raise to different maps for the same graph

3-CONNECTED PLANAR GRAPHS



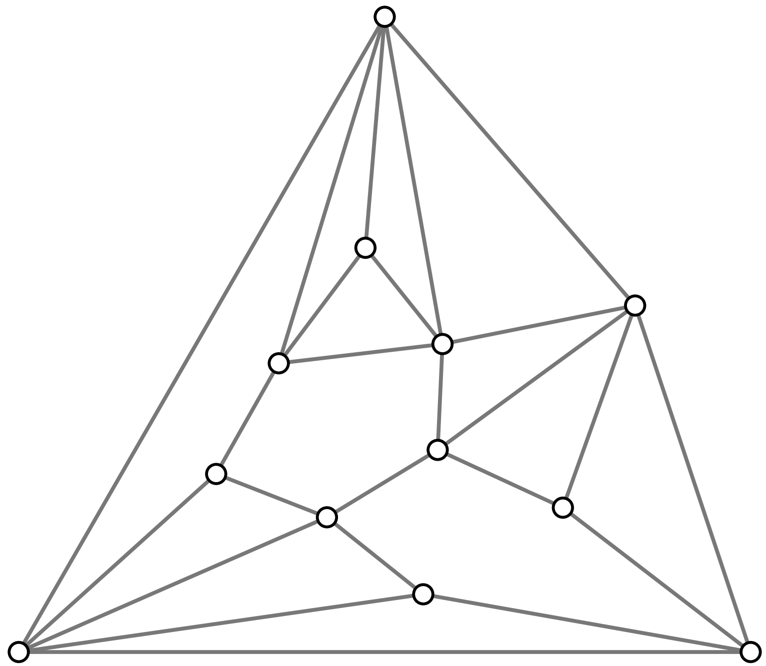


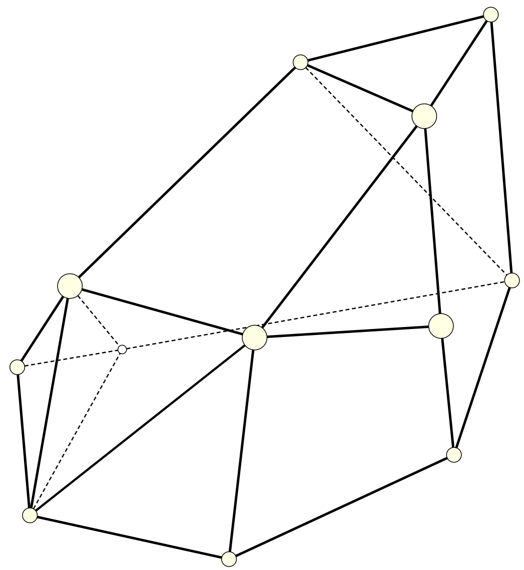
A planar graph is 3-connected if there is no 2-separator.

Whitney : 3-connected planar graphs have a unique embedding up to homeomorphisms of the non-oriented sphere, i.e. a unique planar map.

2-separators give raise to different maps for the same graph

3-CONNECTED PLANAR GRAPHS





A planar graph is 3-connected if there is no 2-separator.

Whitney : 3-connected planar graphs have a unique embedding up to homeomorphisms of the non-oriented sphere, i.e. a unique planar map.

2-separators give raise to different maps for the same graph

Steinitz : They are the 2-skeletons of 3d convex polyhedra.

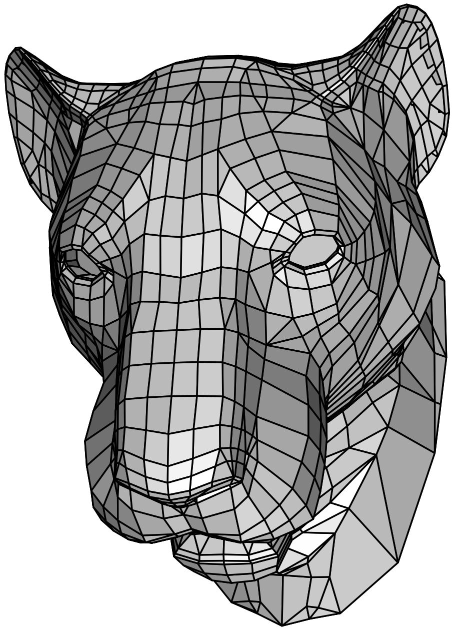
What do we do with 3-connected planar graphs ?

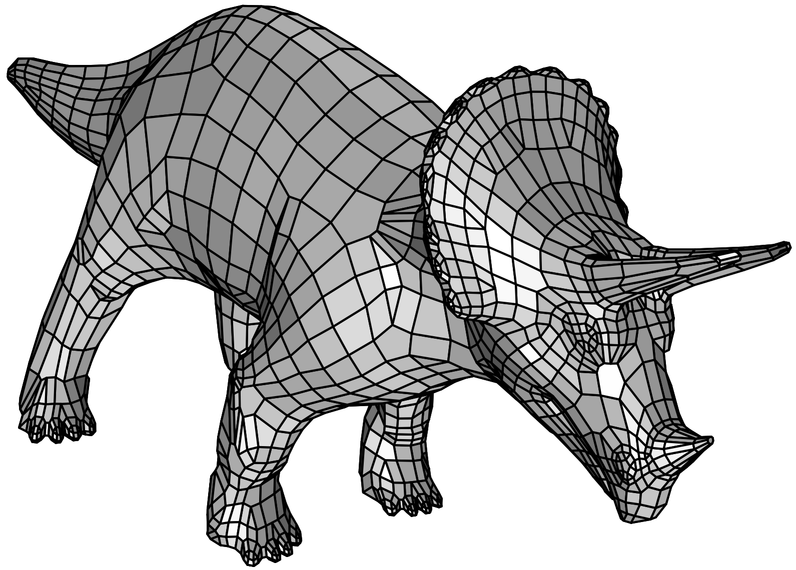
- We want to count them : Tutte counted rooted 3-c planar maps in the 60's, according to their number of edges, Mullin and Schellenberg according to the numbers of faces and vertices.

- We want to generate them uniformly at random :
  $\Rightarrow$ random triangulations and random combinatorial planar maps in general are popular models of discrete random surfaces in physics : random sampler are used to make "experiments" about "2d quantum gravity" (Ambjorn et al. 94,...).
  $\Rightarrow$ random graphs are sometimes used to test graph drawing algorithms.
  $\Rightarrow$ uniform 3-connected planar graphs are needed to sample labelled planar graphs uniformly (Bodirsky–Gröpl–Kang 03)

- We want to encode them compactly.

What do we do with 3-connected planar graphs ?



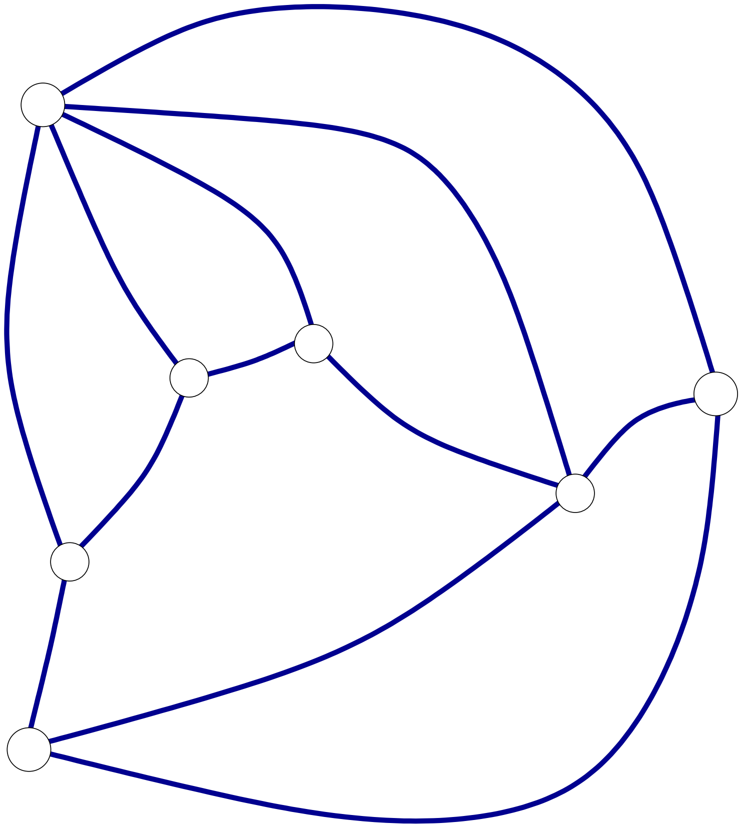


3-connected planar maps = the standard abstraction of the combinatorial part of polygonal meshes with spherical topology (half-edge representations...)

$\Rightarrow$ a number of compression algorithms improving *compression rates*

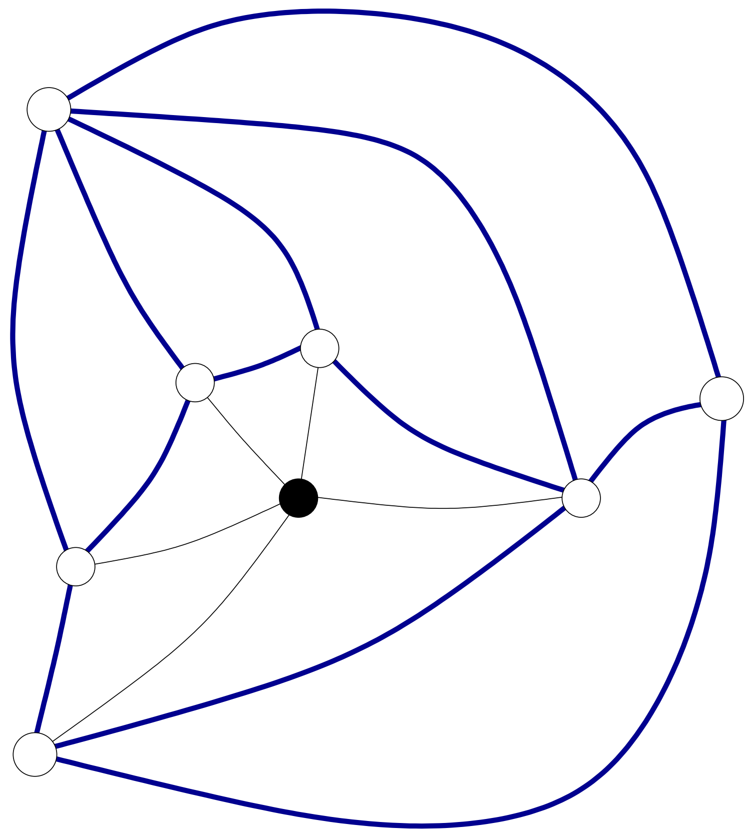
Rossignac's Edgebreaker (98), Touma-Gotsman valency coder (99)...

Planar maps and dissections



Start from a planar map

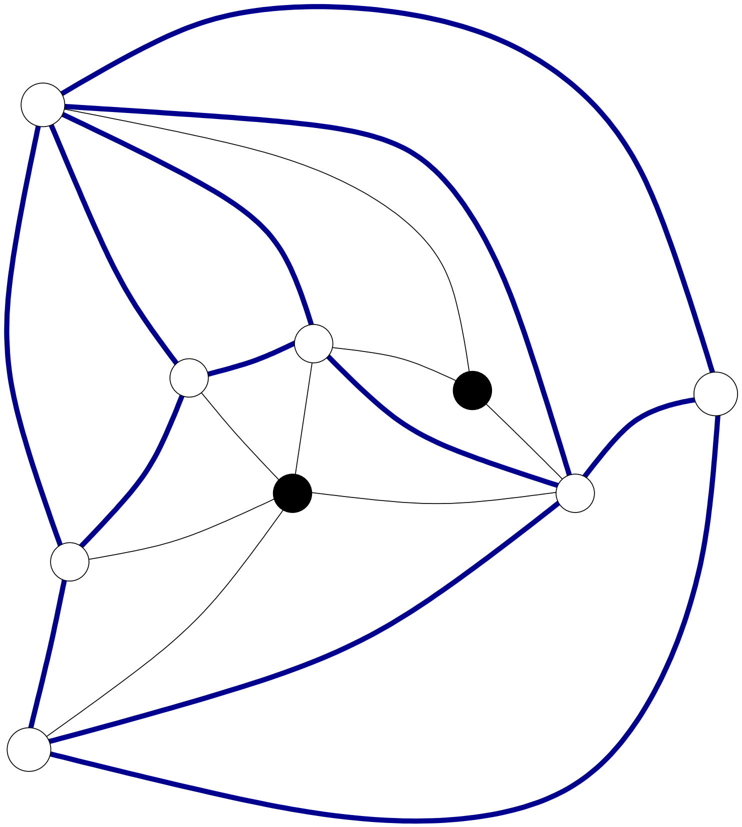
Planar maps and dissections



Start from a planar map

Triangulate faces from new black vertices

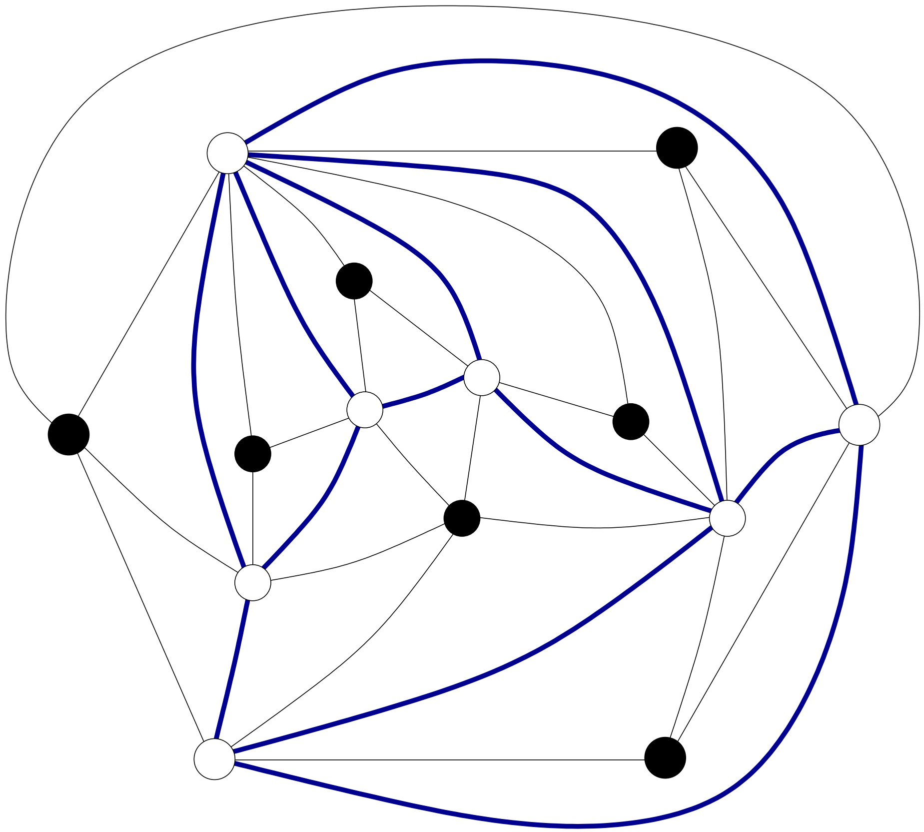
PLANAR MAPS AND DISSECTIONS



Start from a planar map

Triangulate faces from new black vertices

PLANAR MAPS AND DISSECTIONS

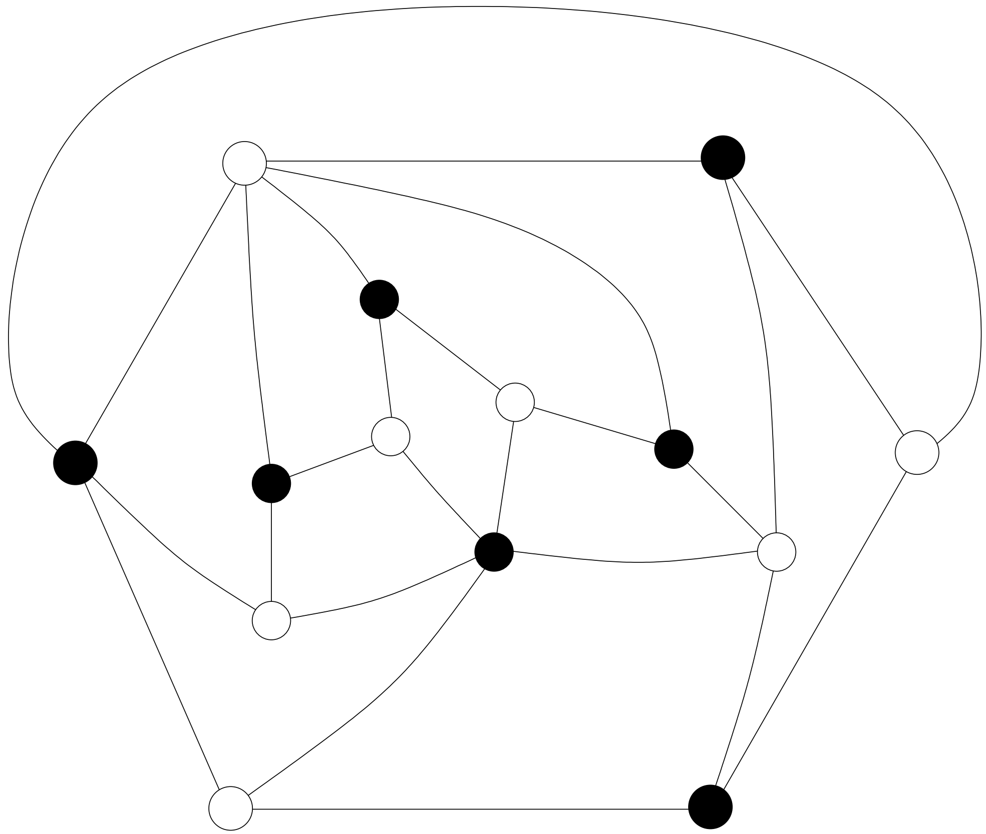


Start from a planar map

Triangulate faces from new black vertices

Forget former edges $\Rightarrow$ quadrangles

Planar maps and dissections

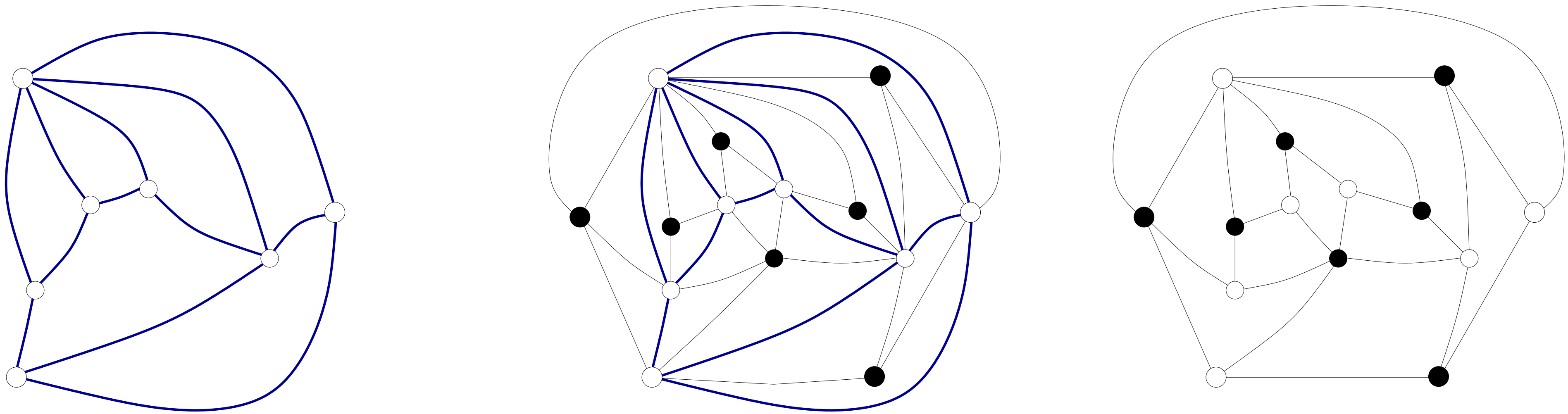


Start from a planar map

Triangulate faces from new black vertices

Forget former edges ⇒ quadrangles ⇒ a quadrangular dissection

PLANAR MAPS AND DISSECTIONS

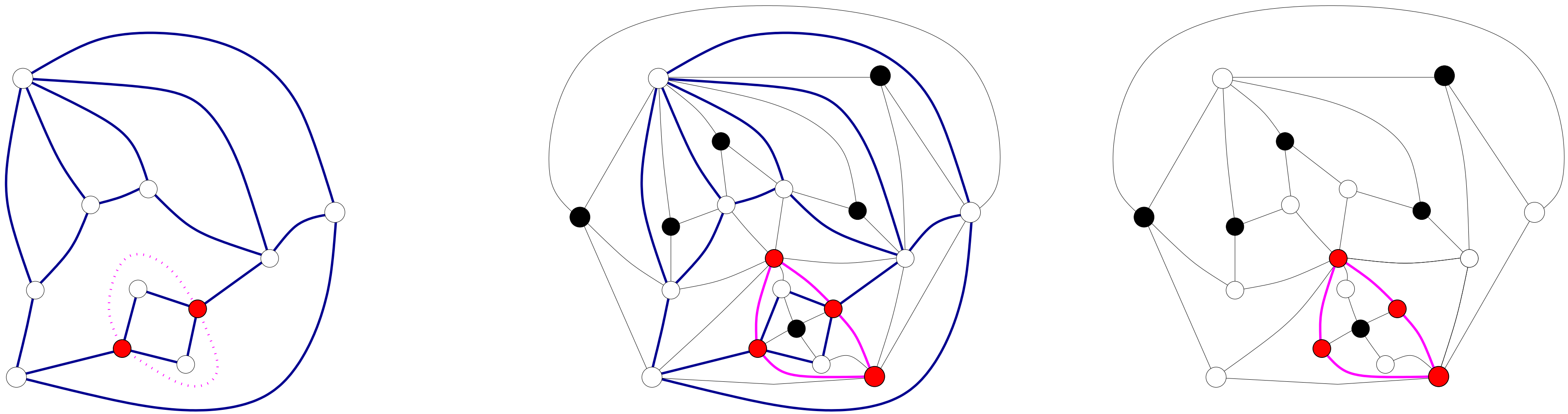


Proposition. This is one-to-one between :

- 3-connected planar maps with n edges,
- irreducible dissections with n faces.

Irreducible = all 4-cycles are faces

PLANAR MAPS AND DISSECTIONS

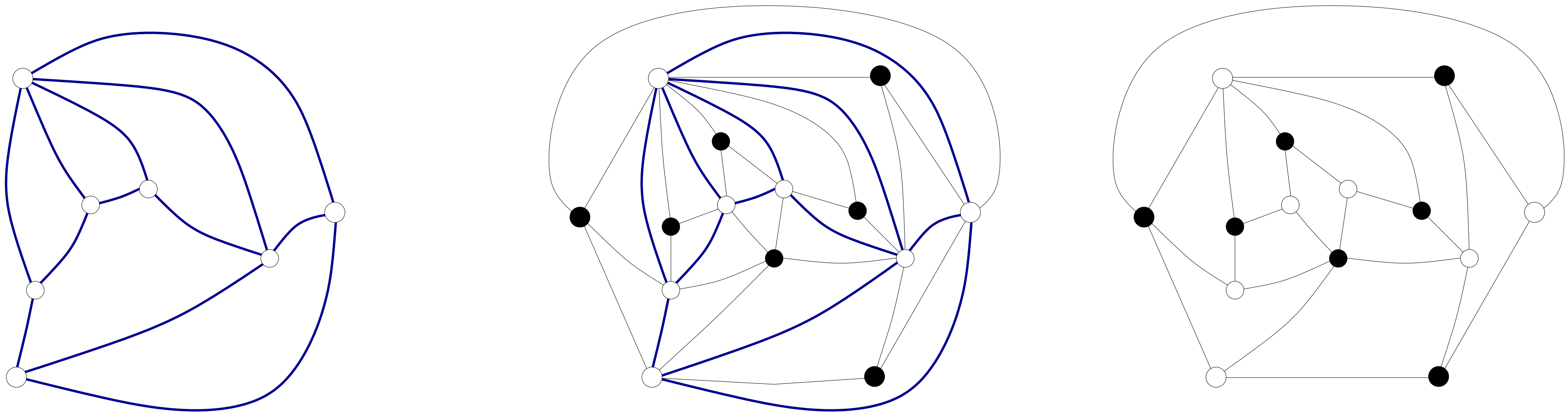


Proposition. This is one-to-one between :

- 3-connected planar maps with n edges,
- irreducible dissections with n faces.

Irreducible = all 4-cycles are faces

PLANAR MAPS AND DISSECTIONS



Proposition. This is one-to-one between :

- 3-connected planar maps with n edges,
- irreducible dissections with n faces.

Irreducible = all 4-cycles are faces

Conclusion of Part 1.

- Our aim : to code, count and sample 3-c planar graphs.
- Equivalently we can consider irreducible dissections.

Part 2. A combinatorial approach to counting, coding and sampling.

CASTING for this part :

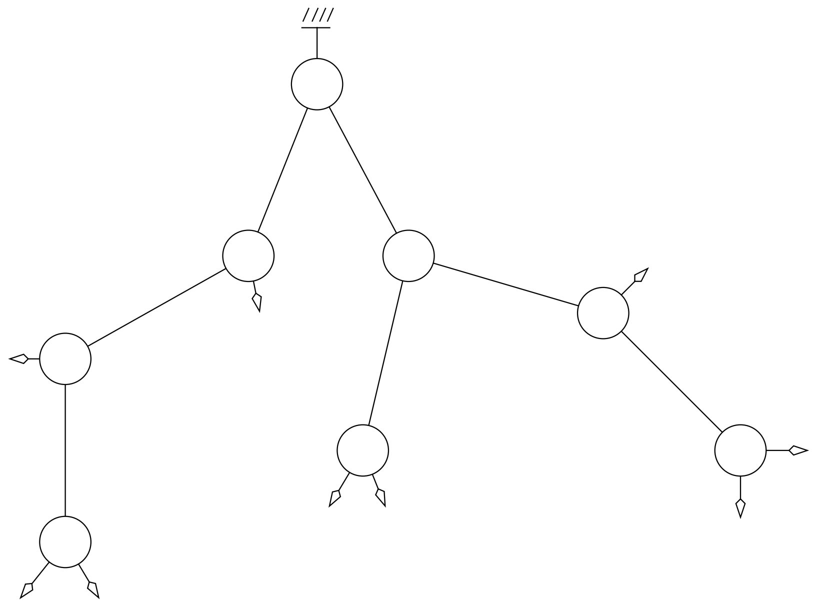
- binary trees
- Catalan numbers

INSPIRATION for this part :

Rémi, Łukasiewicz, folklore...

Binary trees...

Let B_n be the set of binary trees with n inner nodes.

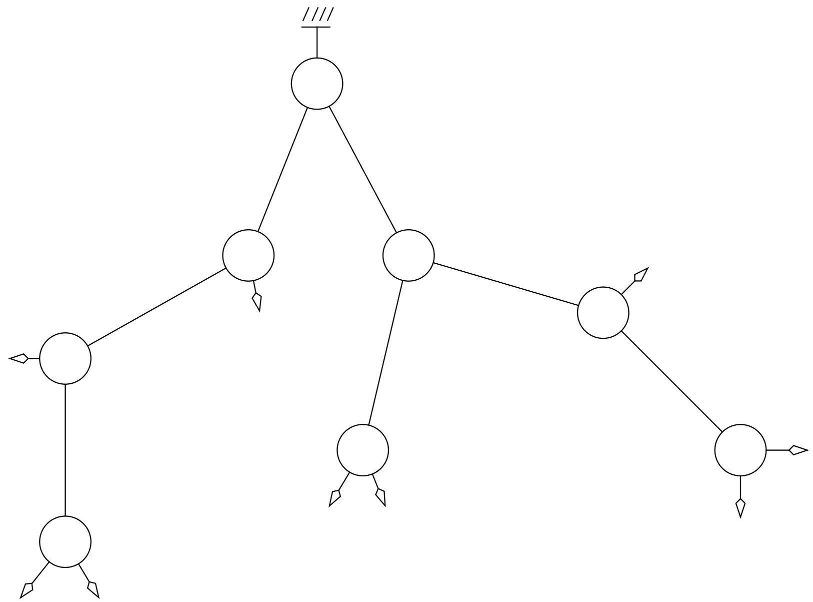


Well known : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n}$, the nth Catalan number.

Seek a constructive proof of this formula and use it for sampling.

Binary trees...

Let B_n be the set of binary trees with n inner nodes.



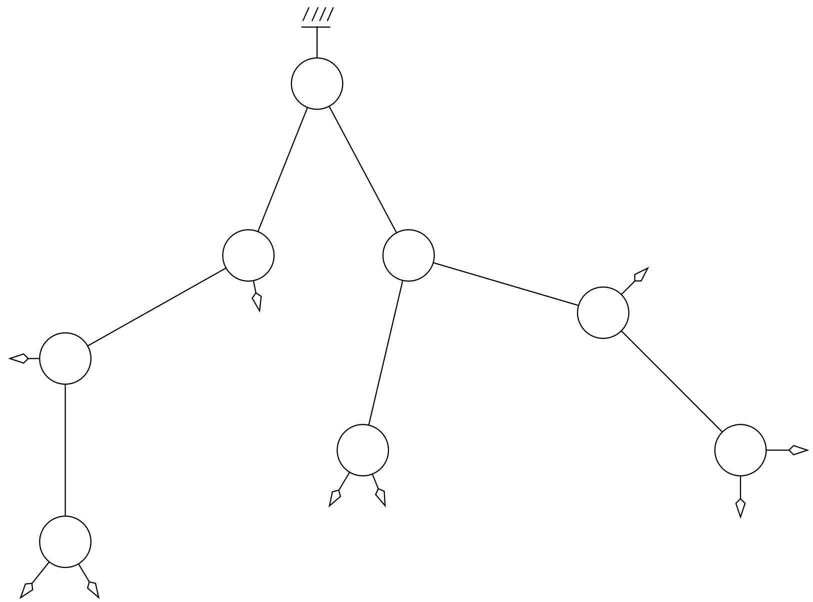
Well known : $|B_n| = \frac{1}{n+1}\binom{2n}{n}$, the nth Catalan number.

In other terms : $2(2n-1)|B_{n-1}| = (n+1)|B_n|$.

Binary trees...

Let B_n be the set of binary trees with n inner nodes.

A tree of B_n has $n+1$ leaves.



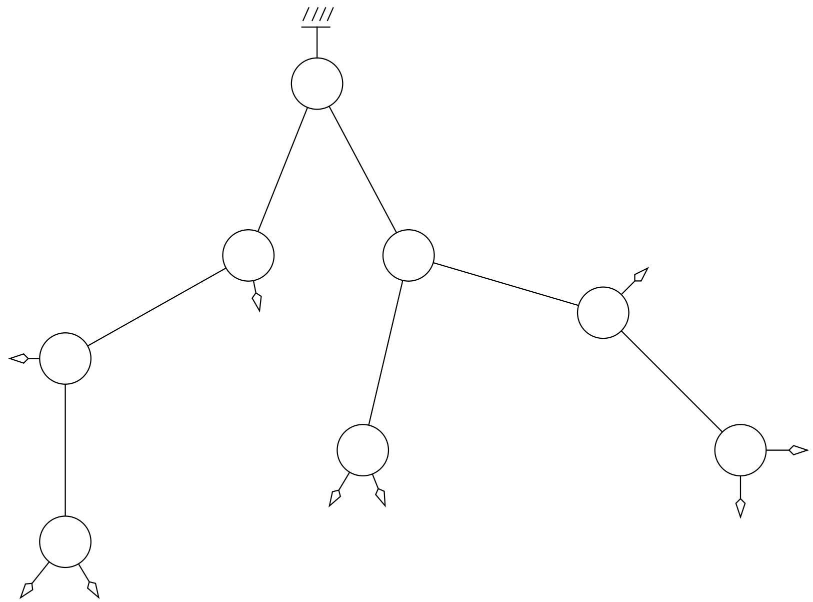
Well known : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n}$, the nth Catalan number.

In other terms : $2(2n-1)|B_{n-1}| = (n+1)|B_n|$.

Binary trees...

Let B_n be the set of binary trees with n inner nodes.

A tree of B_n has $n+1$ leaves.

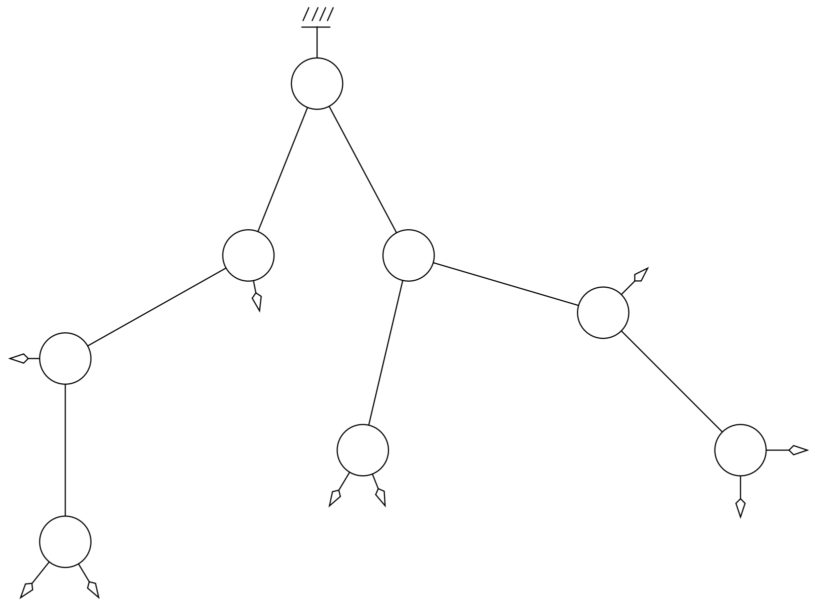


Well known : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n}$, the nth Catalan number.

In other terms : $2(2n-1)|B_{n-1}| = |\{\text{leaves}\} \times B_n|$.

Binary trees...

Let B_n be the set of binary trees with n inner nodes.



A tree of B_n has $n+1$ leaves.

A tree of B_{n-1} has $2n-1$ edges.
(including the root edge)

Well known : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n}$, the nth Catalan number.

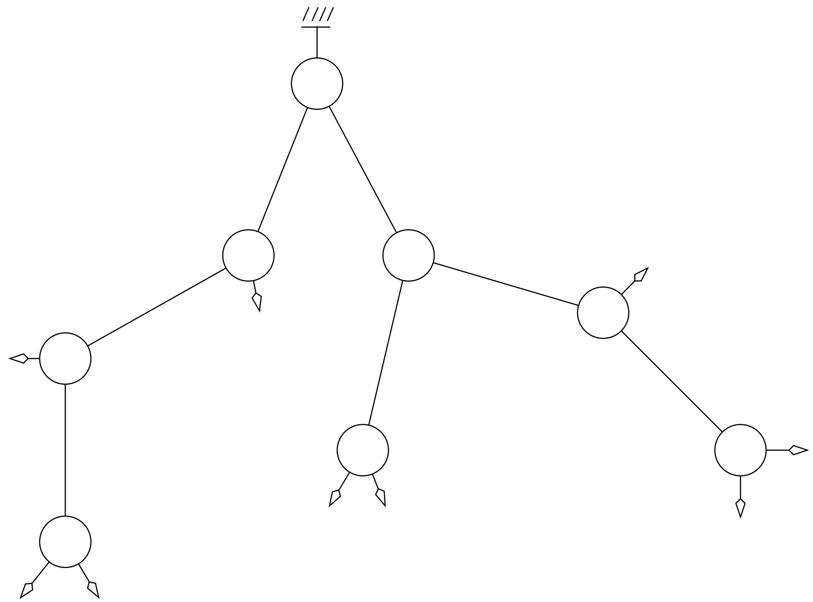
In other terms : $2(2n-1)|B_{n-1}| = |\{\text{leaves}\} \times B_n|$.

Binary trees...

Let B_n be the set of binary trees with n inner nodes.

A tree of B_n has $n+1$ leaves.

A tree of B_{n-1} has $2n-1$ edges. (including the root edge)

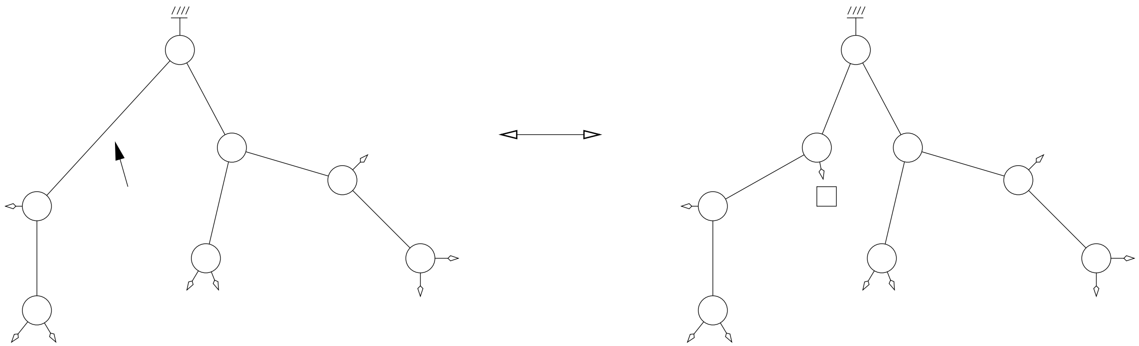


Well known : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n}$, the nth Catalan number.

In other terms : $|\{l, r\} \times \{\text{edges}\} \times B_{n-1}| = |\{\text{leaves}\} \times B_n|$.

Binary trees... Constructive counting

A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.

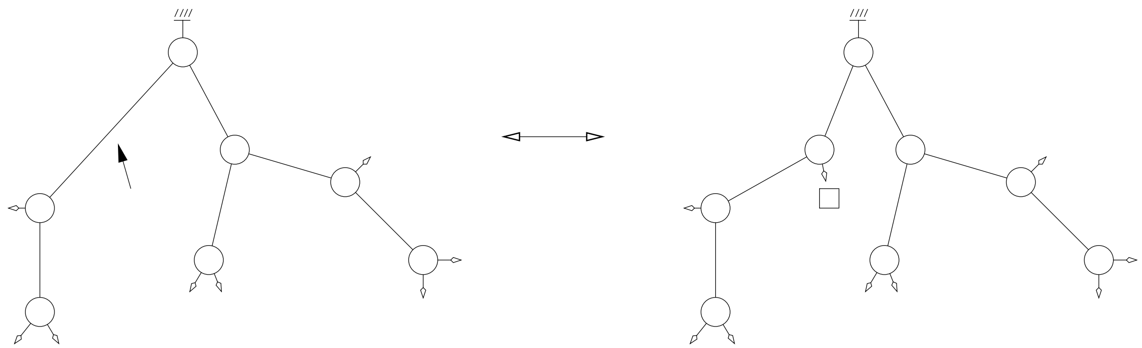


This yields :

- A proof of the recurrence
  $\Rightarrow$ constructive counting.

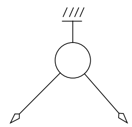
BINARY TREES... CONSTRUCTIVE COUNTING

A bijection : $\{l,r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

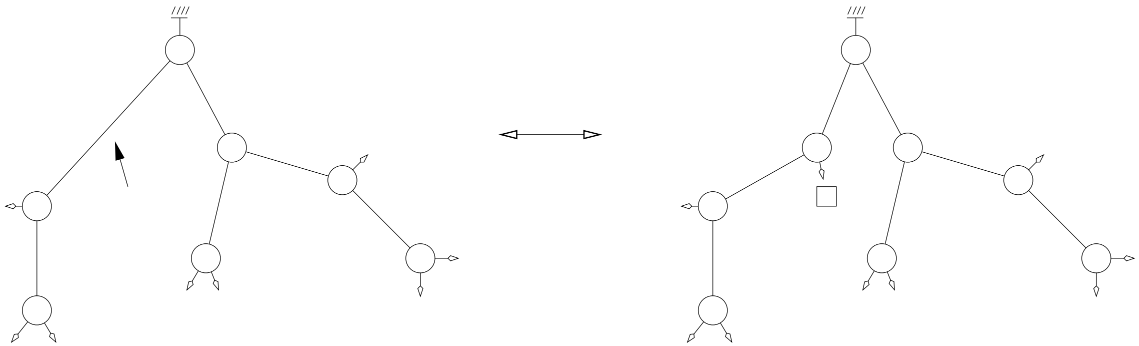
- A proof of the recurrence
  $\Rightarrow$ constructive counting.



- A random sampling algorithm :
  Pick a side in $\{l,r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.

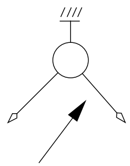
Binary trees... Constructive counting

A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

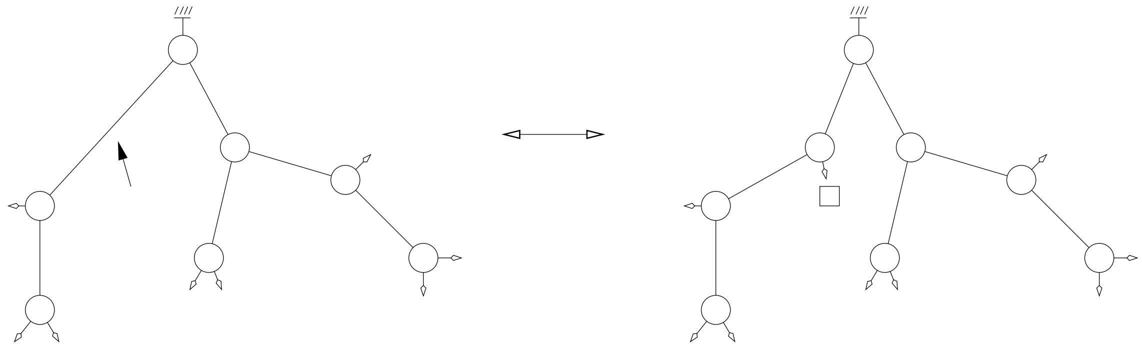
- A proof of the recurrence
  $\Rightarrow$ constructive counting.



- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.

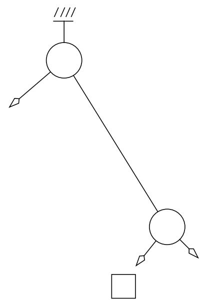
Binary trees... Constructive counting

A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

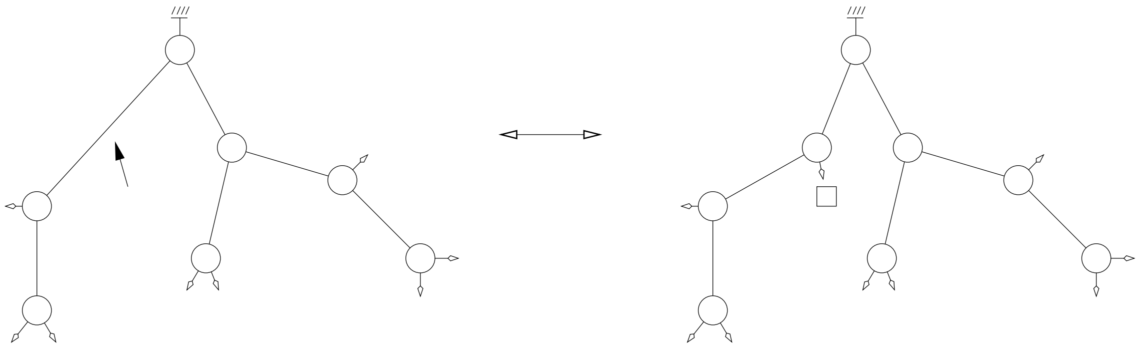
- A proof of the recurrence
  $\Rightarrow$ constructive counting.



- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.

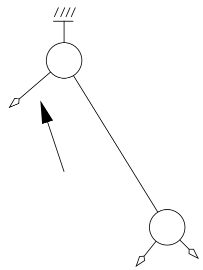
BINARY TREES... CONSTRUCTIVE COUNTING

A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

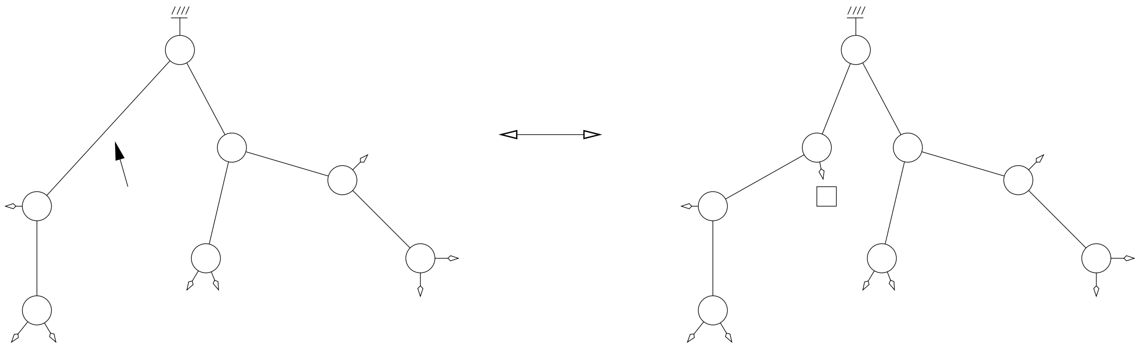
- A proof of the recurrence
  $\Rightarrow$ constructive counting.



- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.

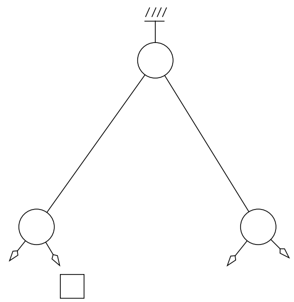
BINARY TREES... CONSTRUCTIVE COUNTING

A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

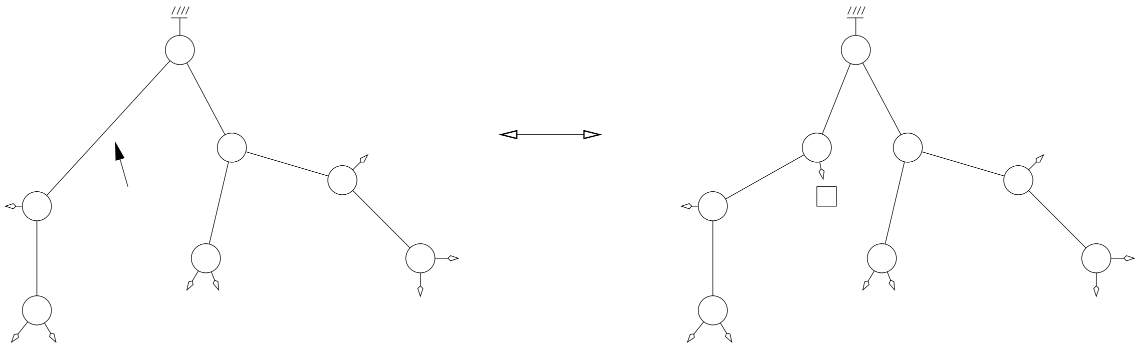
- A proof of the recurrence
  $\Rightarrow$ constructive counting.



- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.

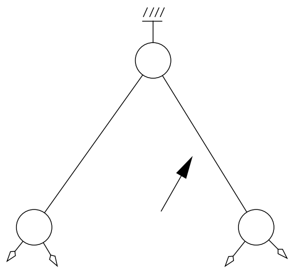
BINARY TREES... CONSTRUCTIVE COUNTING

A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

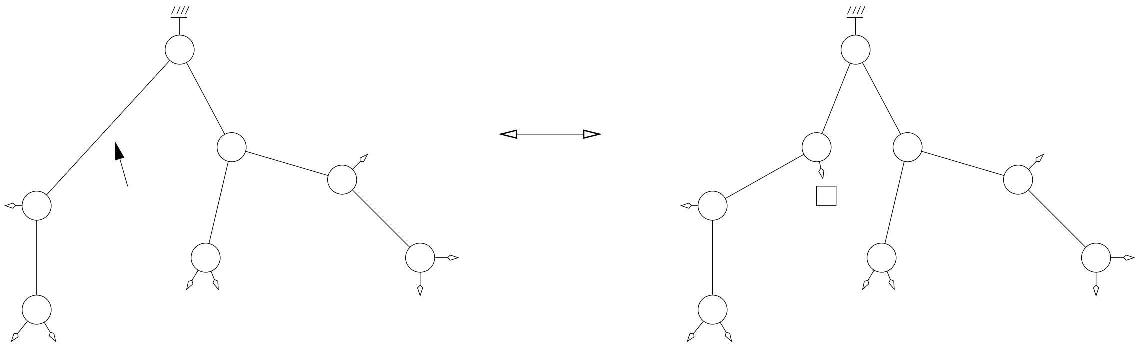
- A proof of the recurrence
  $\Rightarrow$ constructive counting.



- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.

Binary trees... Constructive counting

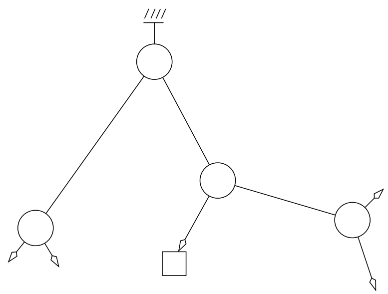
A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

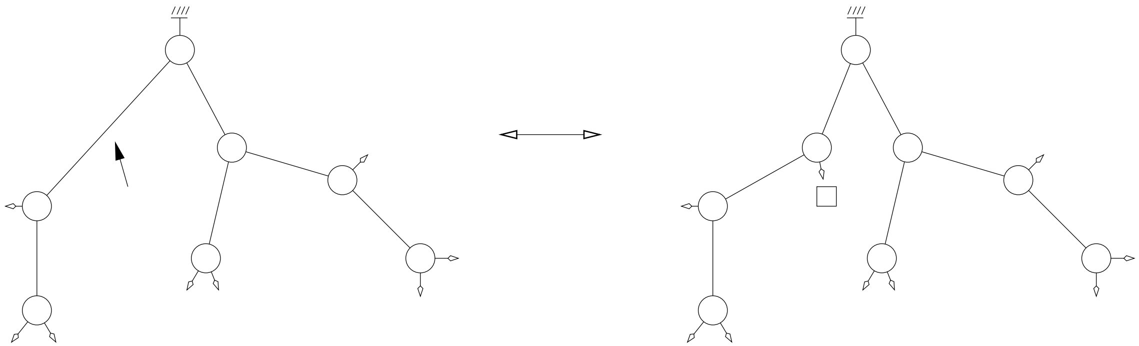
- A proof of the recurrence
  $\Rightarrow$ constructive counting.

- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.



Binary trees... Constructive counting

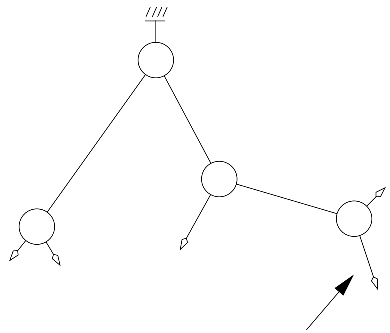
A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

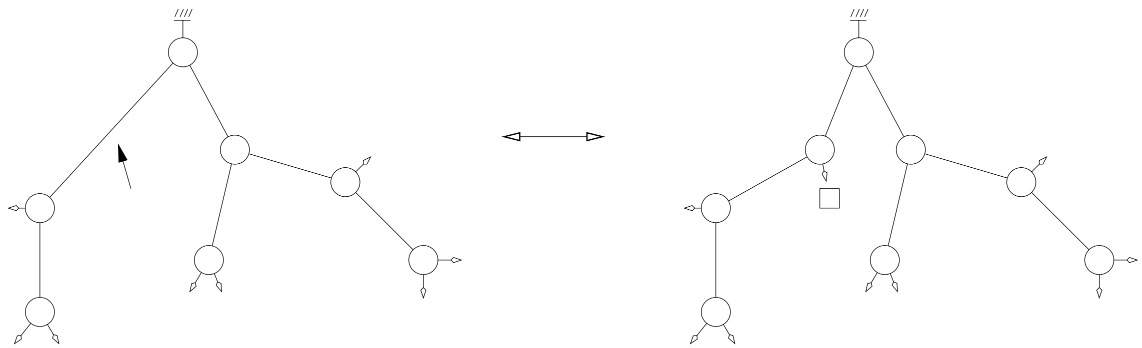
- A proof of the recurrence
  $\Rightarrow$ constructive counting.

- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.



Binary trees... Constructive counting

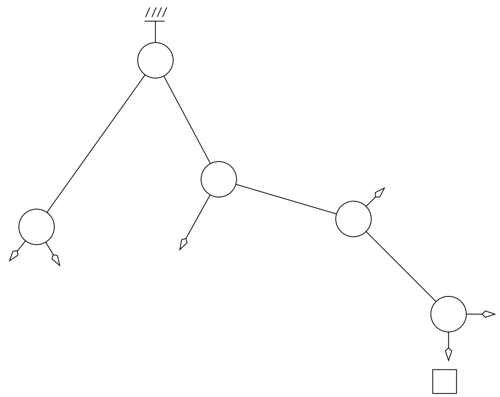
A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

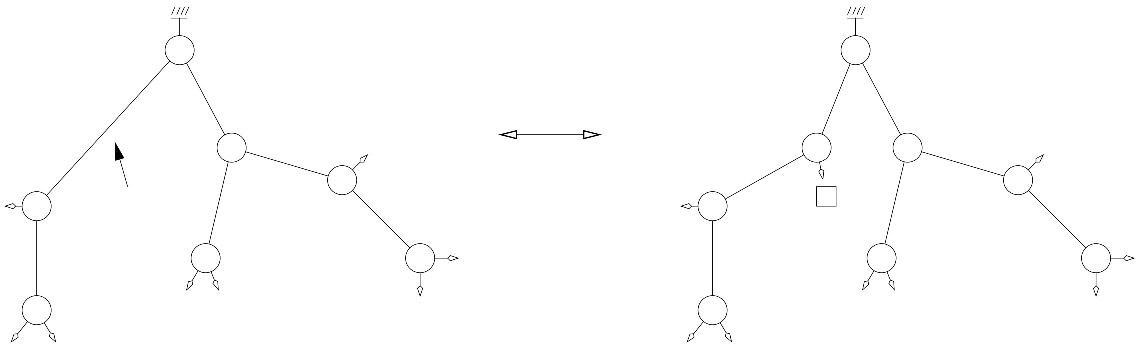
- A proof of the recurrence
  $\Rightarrow$ constructive counting.

- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.



BINARY TREES... CONSTRUCTIVE COUNTING

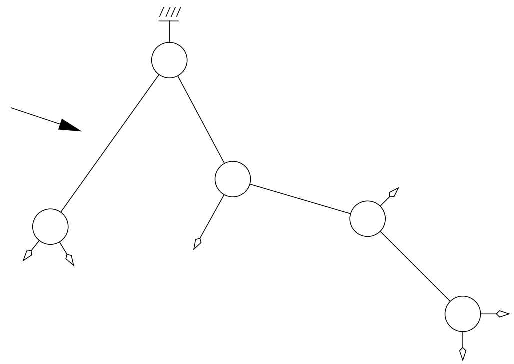
A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

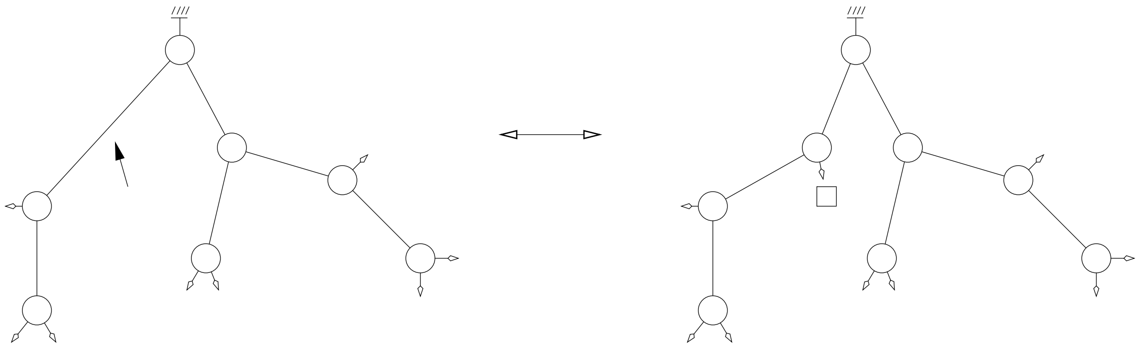
- A proof of the recurrence
  $\Rightarrow$ constructive counting.

- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.



BINARY TREES... CONSTRUCTIVE COUNTING

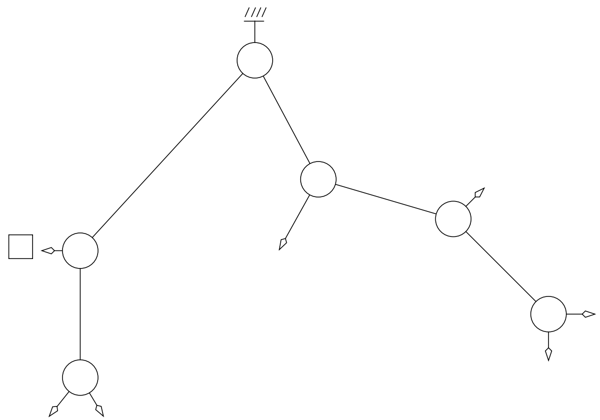
A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

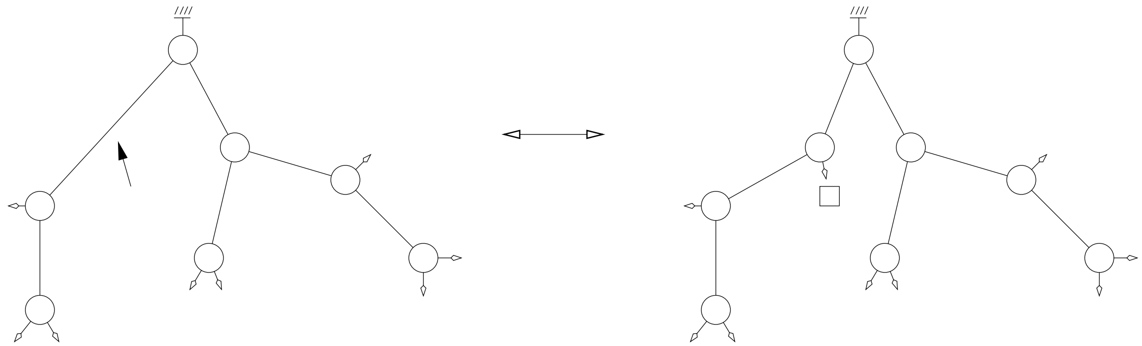
- A proof of the recurrence
  $\Rightarrow$ constructive counting.

- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.



BINARY TREES... CONSTRUCTIVE COUNTING

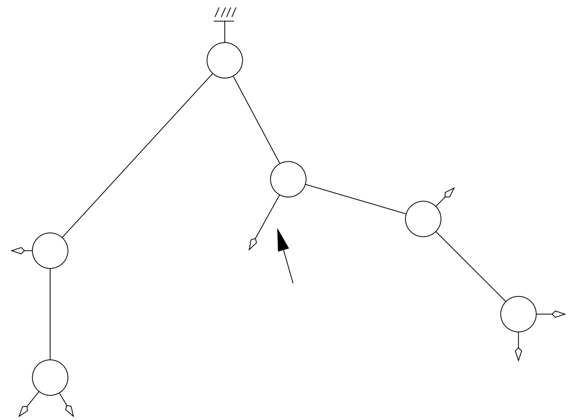
A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

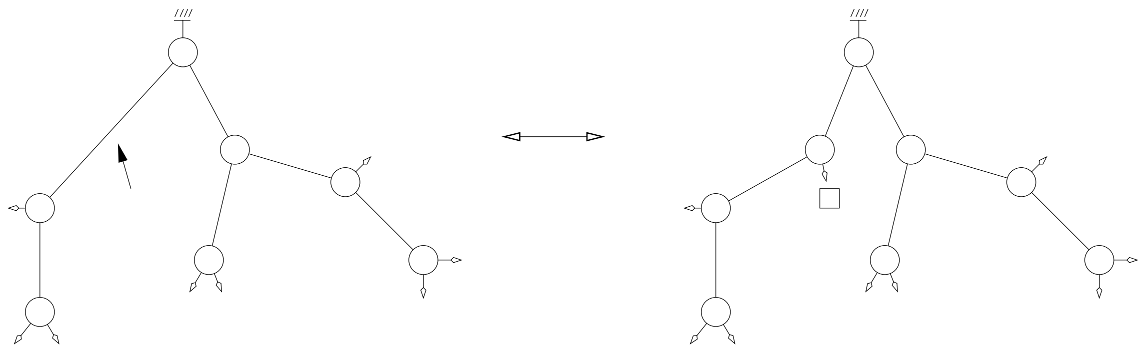
- A proof of the recurrence
  $\Rightarrow$ constructive counting.

- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.



Binary trees... Constructive counting

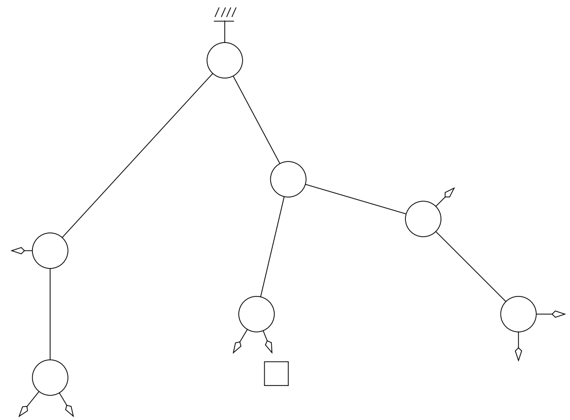
A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

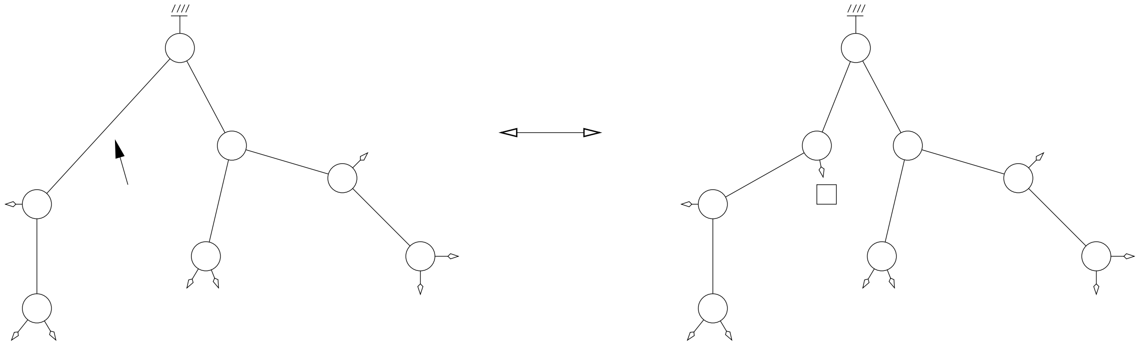
- A proof of the recurrence
  $\Rightarrow$ constructive counting.

- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.



Binary trees... Constructive counting

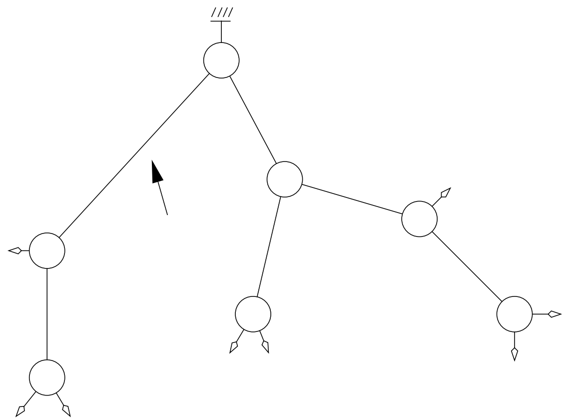
A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

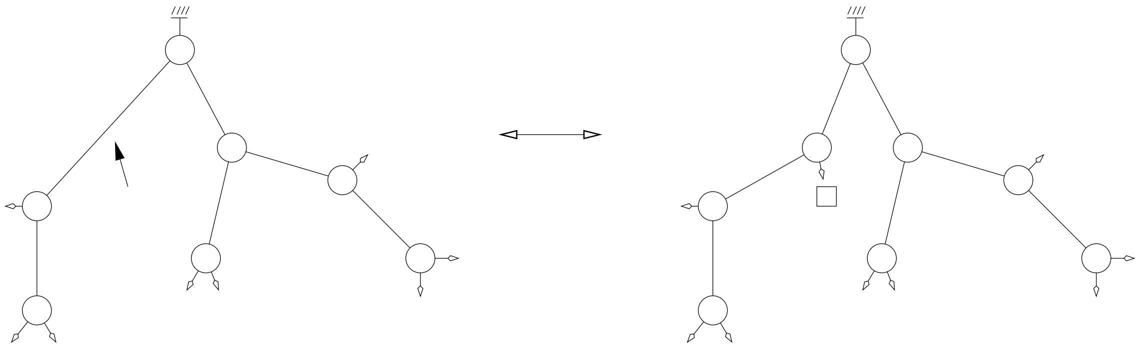
- A proof of the recurrence
  $\Rightarrow$ constructive counting.

- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.



BINARY TREES... CONSTRUCTIVE COUNTING

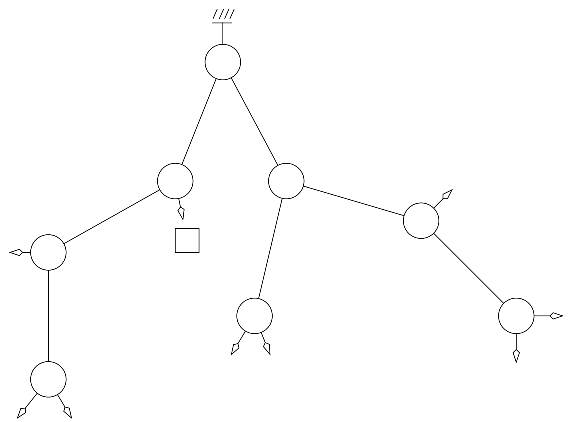
A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

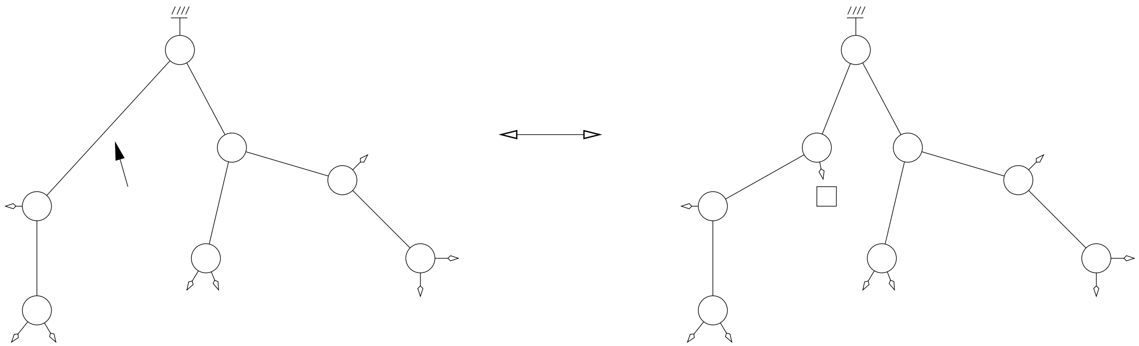
- A proof of the recurrence
  $\Rightarrow$ constructive counting.

- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.



BINARY TREES... CONSTRUCTIVE COUNTING

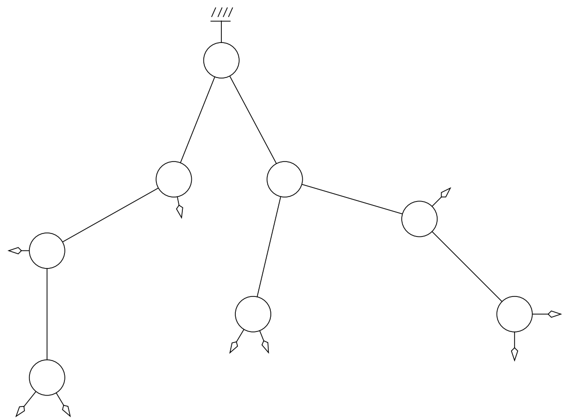
A bijection : $\{l, r\} \times \{\text{edges}\} \times B_{n-1} \leftrightarrow \{\text{leaves}\} \times B_n$.



This yields :

- A proof of the recurrence
  $\Rightarrow$ constructive counting.

- A random sampling algorithm :
  Pick a side in $\{l, r\}$ and an edge uniformly at random and grow
  $\Rightarrow$ the nth tree is uniform in B_n.

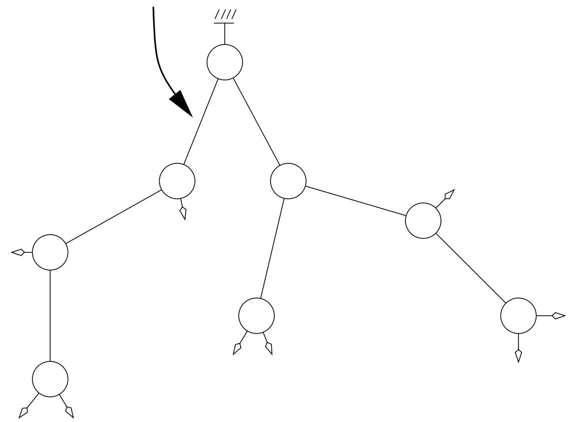


Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



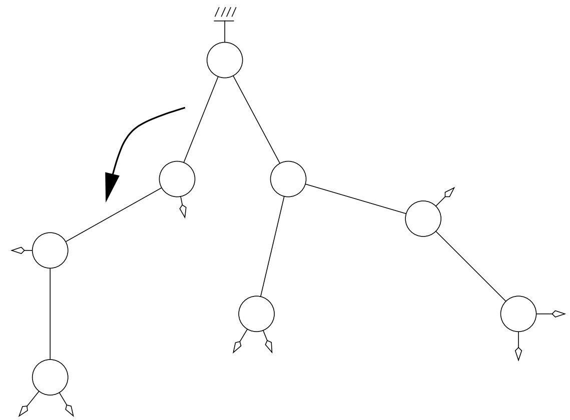
1

BINARY TREES... CONSTRUCTIVE COUNTING

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.



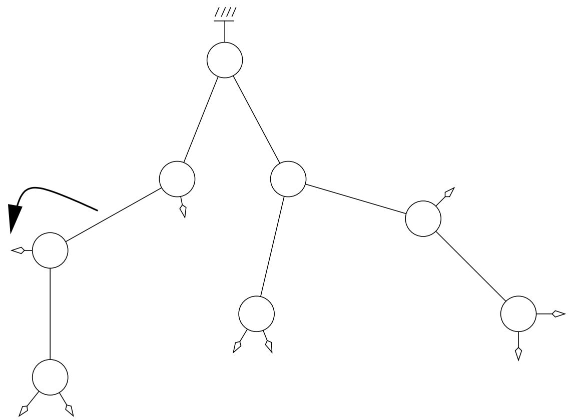
– Write 1 for left edges, 0 for right ones along a prefix traversal.

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.

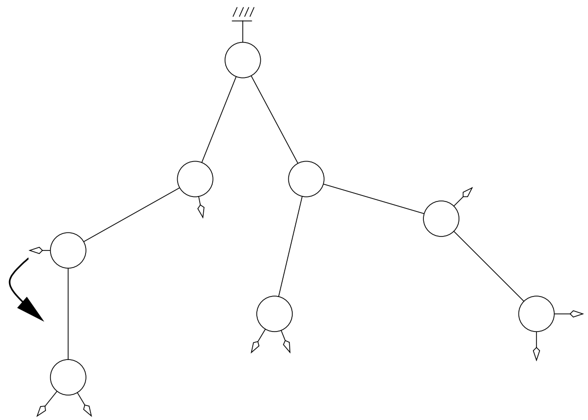


111

Another observation : $|B_n| = \frac{1}{n+1}\binom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.



– Write 1 for left edges, 0 for right ones along a prefix traversal.

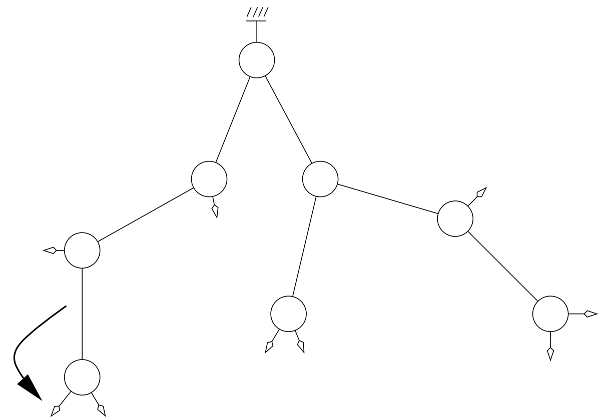
1110

BINARY TREES... CONSTRUCTIVE COUNTING

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.



– Write 1 for left edges, 0 for right ones along a prefix traversal.

11101

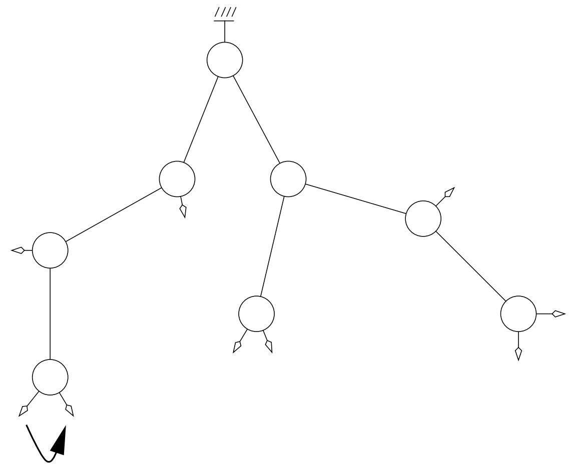
Binary trees... Constructive counting

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



111010

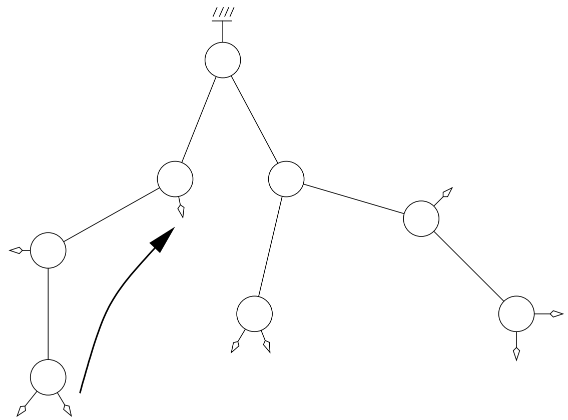
Binary trees... Constructive counting

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



1110100

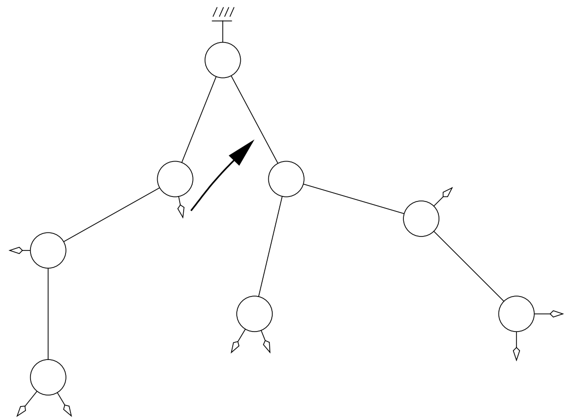
Binary trees... Constructive counting

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



11101000

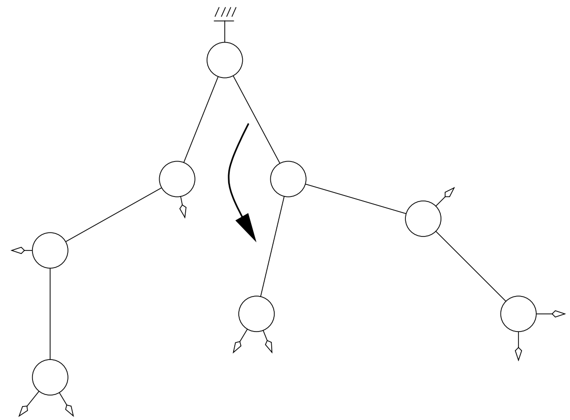
Binary trees... Constructive counting

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



111010001

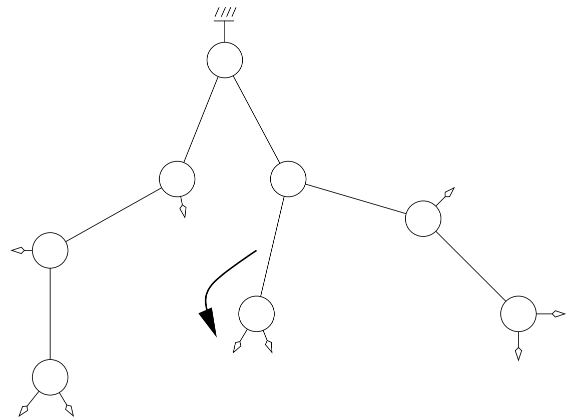
BINARY TREES... CONSTRUCTIVE COUNTING

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0, 1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



1110100011

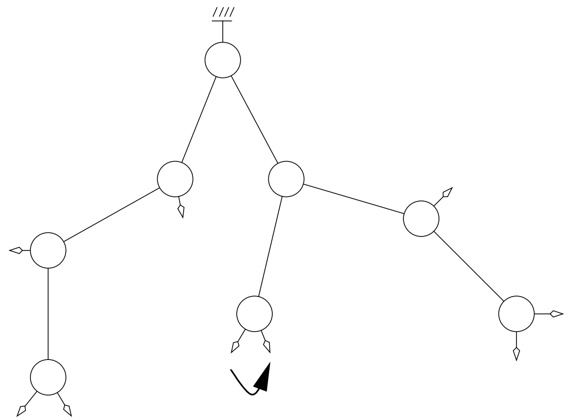
BINARY TREES... CONSTRUCTIVE COUNTING

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



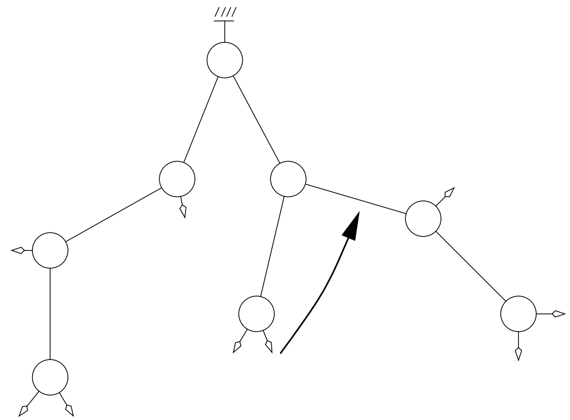
11101000110

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



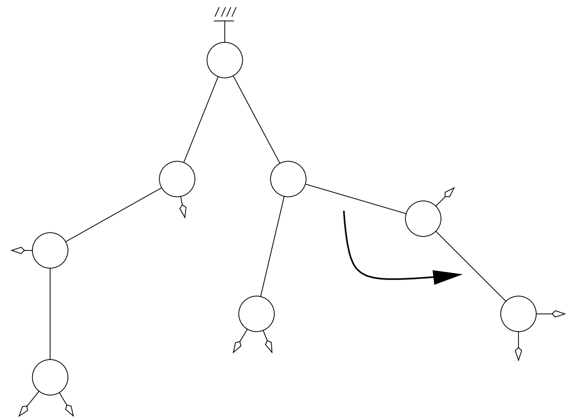
111010001100

Binary trees... Constructive counting

Another observation : $|B_n| = \frac{1}{n+1}\binom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.



– Write 1 for left edges, 0 for right ones along a prefix traversal.

1110100011001

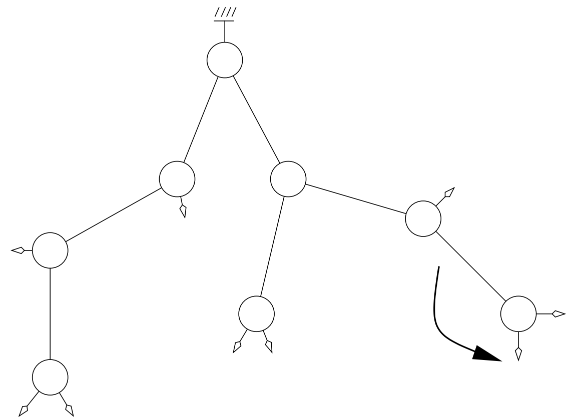
BINARY TREES... CONSTRUCTIVE COUNTING

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



11101000110011

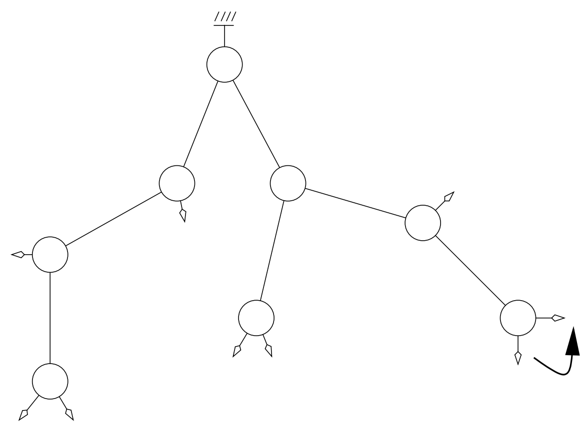
BINARY TREES... CONSTRUCTIVE COUNTING

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



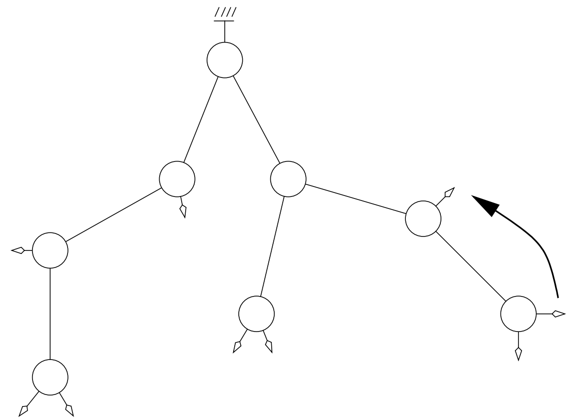
111010001100110

Binary trees... Constructive counting

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.



– Write 1 for left edges, 0 for right ones along a prefix traversal.

1110100011001100

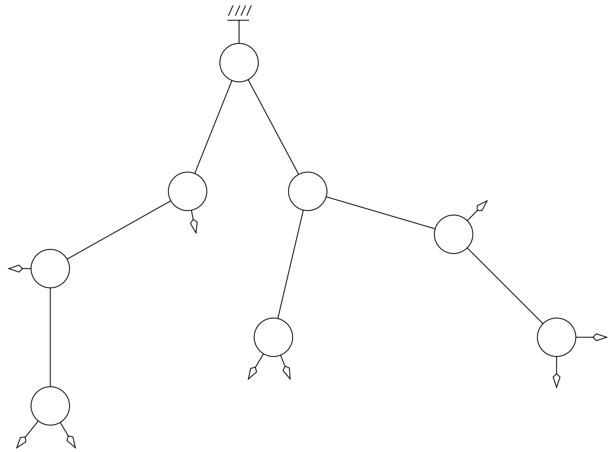
BINARY TREES... CONSTRUCTIVE COUNTING

Another observation : $|B_n| = \dfrac{1}{n+1}\dbinom{2n}{n} \sim 2^{2n} \cdot cn^{-5/2}$.

$\Rightarrow$ It should be possible to encode trees of $|B_n|$ by words of $\{0,1\}^{2n}$.

This can be done by prefix encoding.

– Write 1 for left edges, 0 for right ones along a prefix traversal.



1110100011001100

This code has length $2n$ and is optimal in the sense that a code must use at least $2n + o(n)$ bits in the worst case.

Conclusion of Part 2.

- Bijections can help for counting, coding and sampling.
- Binary trees are well known...

Part 3. The closure of a binary tree into a dissection

CASTING for this part :

- binary trees (again)
- irreducible dissections (stunt men return)
- 3-connected planar graphs (hors champs)

Tutte's results about 3-connected planar graphs

The number 3-connected planar graphs ?

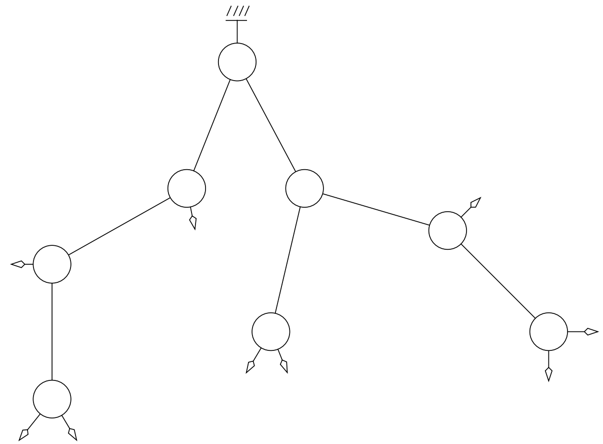
Tutte (62) : a complicated formula for rooted 3-c planar maps.

However these numbers are "Catalan related" (their generating function lies in the same algebraic extention).

$\Rightarrow$ explain this combinatorially...

We would like to find a simple one-to-one correspondence between 3c planar graphs and binary trees.

The closure of a binary tree



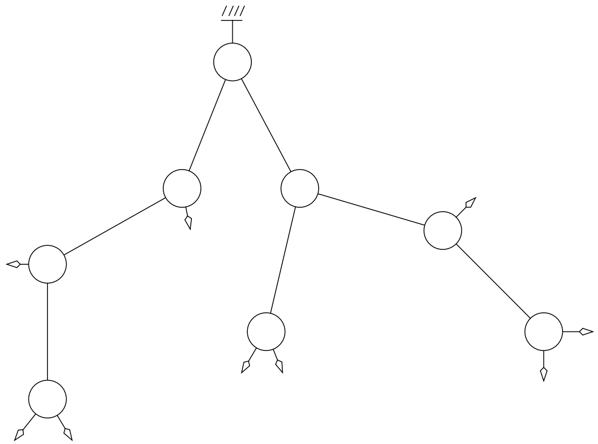
Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n+1$ edges.

Compare to :

3-c planar graphs : n edges, i vertices, j faces, with $i+j=n+2$ (Euler).

THE CLOSURE OF A BINARY TREE

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n+1$ edges.

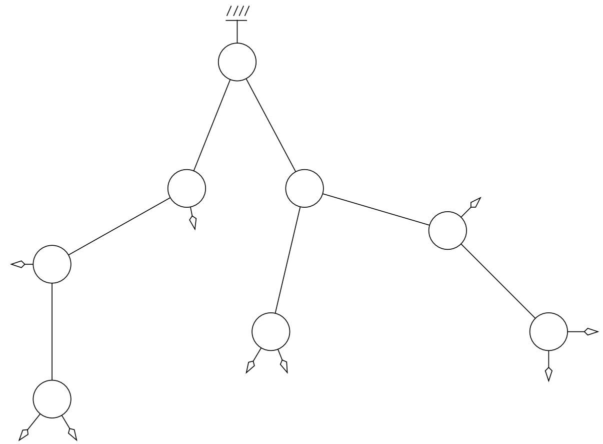


Compare to :

Dissections : n faces, $2n$ edges and $n+2$ vertices (by Euler).

The closure of a binary tree

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n+1$ edges.

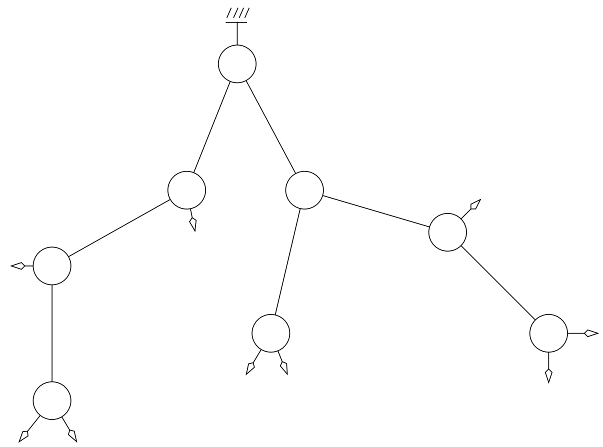


Compare to :

Dissections : n faces, $2n$ edges and $n+2$ vertices (by Euler).

Create faces of degree four, keeping the number of vertices and edges...

The closure of a binary tree



Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n+1$ edges.

Compare to :

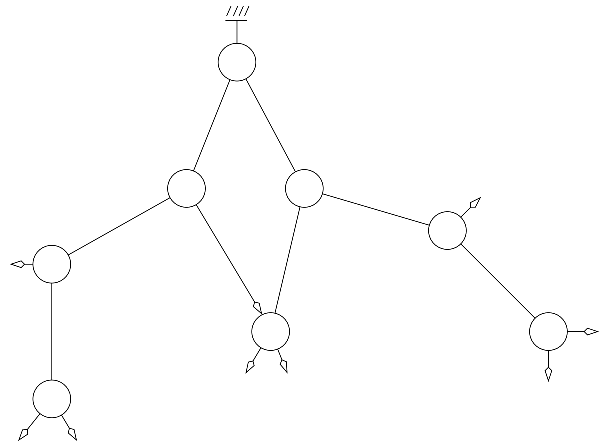
Dissections : n faces, $2n$ edges and $n+2$ vertices (by Euler).

Create faces of degree four, keeping the number of vertices and edges...

Local closure : close an leaf followed in ccw order by 3 sides of inner edges.

THE CLOSURE OF A BINARY TREE

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n+1$ edges.



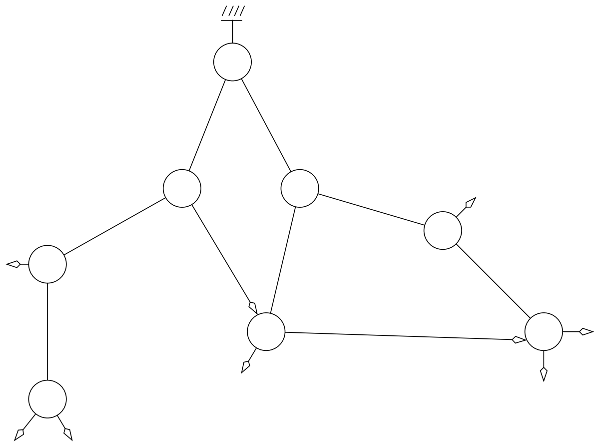
Compare to :

Dissections : n faces, $2n$ edges and $n+2$ vertices (by Euler).

Create faces of degree four, keeping the number of vertices and edges...

Local closure : close an leaf followed in ccw order by 3 sides of inner edges.

THE CLOSURE OF A BINARY TREE



Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n+1$ edges.

Compare to :

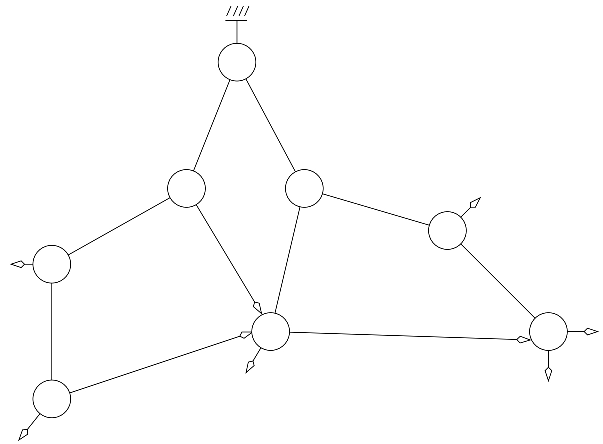
Dissections : n faces, $2n$ edges and $n+2$ vertices (by Euler).

Create faces of degree four, keeping the number of vertices and edges...

Local closure : close an leaf followed in ccw order by 3 sides of inner edges.

THE CLOSURE OF A BINARY TREE

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n+1$ edges.



Compare to :

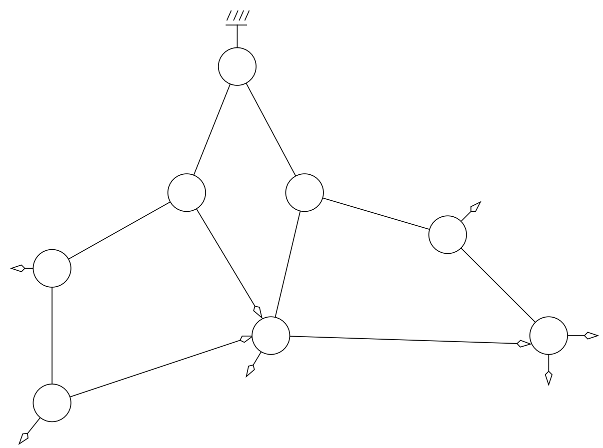
Dissections : n faces, $2n$ edges and $n+2$ vertices (by Euler).

Create faces of degree four, keeping the number of vertices and edges...

Local closure : close an leaf followed in ccw order by 3 sides of inner edges.

The closure of a binary tree

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n+1$ edges.



Compare to :

Dissections : n faces, $2n$ edges and $n+2$ vertices (by Euler).

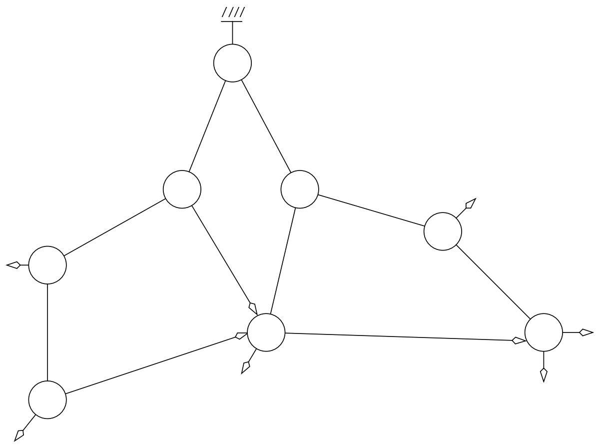
Create faces of degree four, keeping the number of vertices and edges...

Local closure : close an leaf followed in ccw order by 3 sides of inner edges.

Remark : local closures commute, the resulting partial closure is well defined.

BINARY TREES AND THE CLOSURE

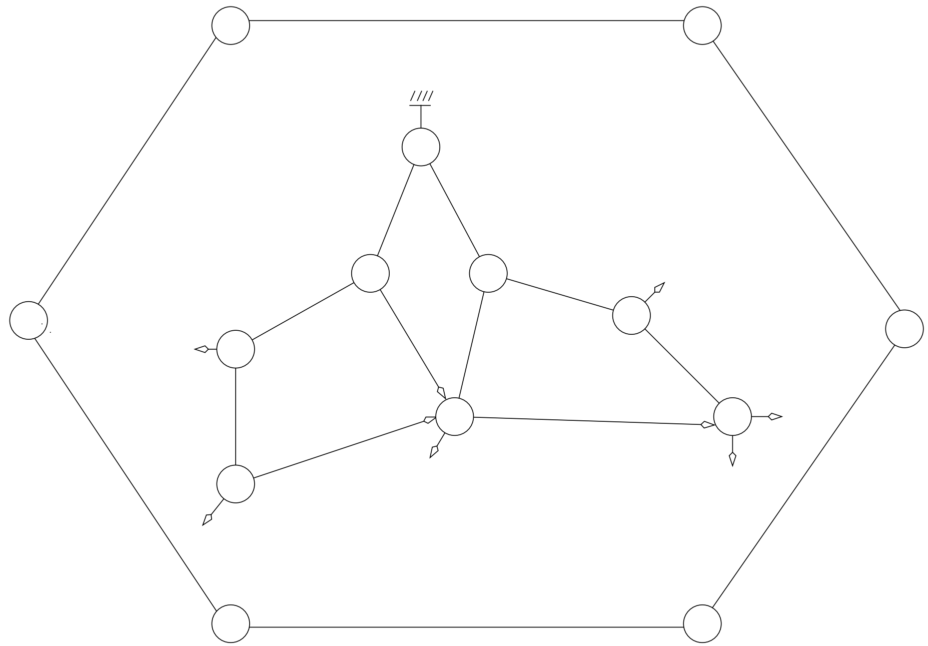
Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n$ edges.



Partial closure : when all local closures are done the numbers k of remaining leaves and ℓ of sides of inner edges in the infinite face satisfy $2k-\ell=6$.

BINARY TREES AND THE CLOSURE

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n$ edges.

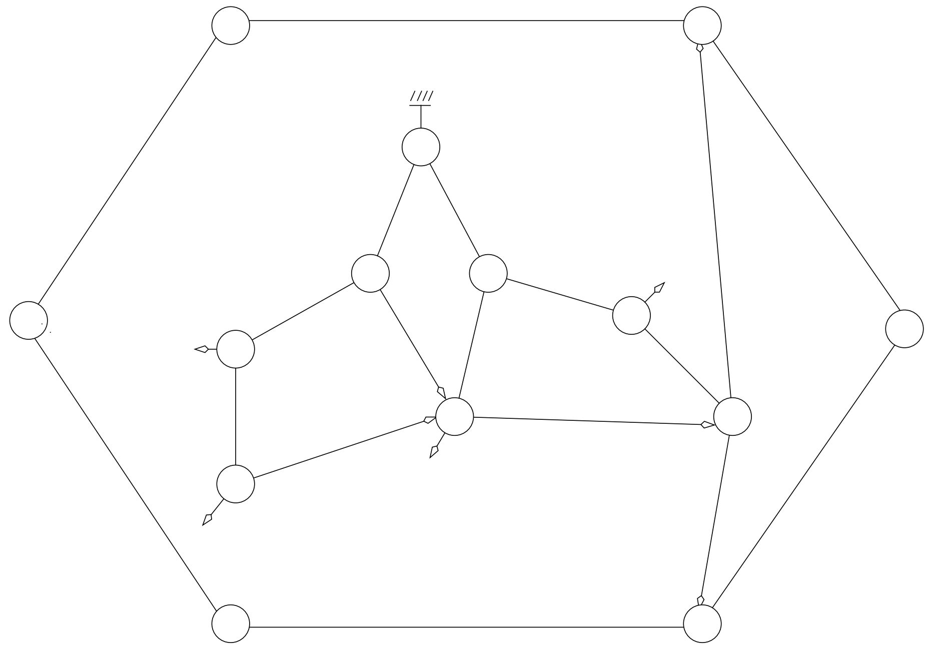


Partial closure : when all local closures are done the numbers k of remaining leaves and ℓ of sides of inner edges in the infinite face satisfy $2k - \ell = 6$.

Complete closure : The remaining leaves can be attached to the vertices of hexagon so as to form faces of degree 4.

Binary trees and the closure

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n$ edges.

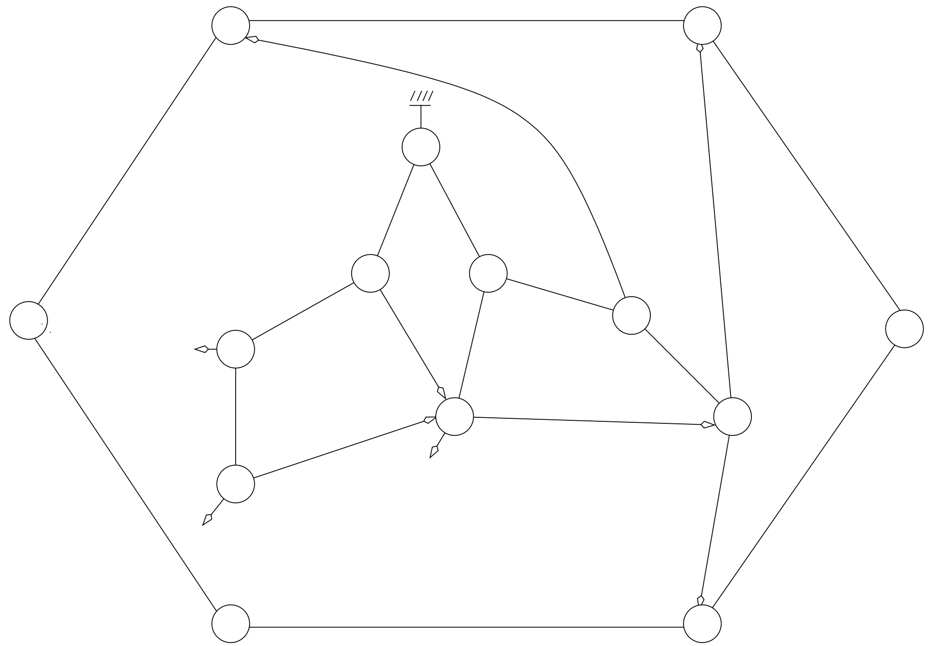


Partial closure : when all local closures are done the numbers k of remaining leaves and ℓ of sides of inner edges in the infinite face satisfy $2k-\ell=6$.

Complete closure : The remaining leaves can be attached to the vertices of hexagon so as to form faces of degree 4.

Binary trees and the closure

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n$ edges.

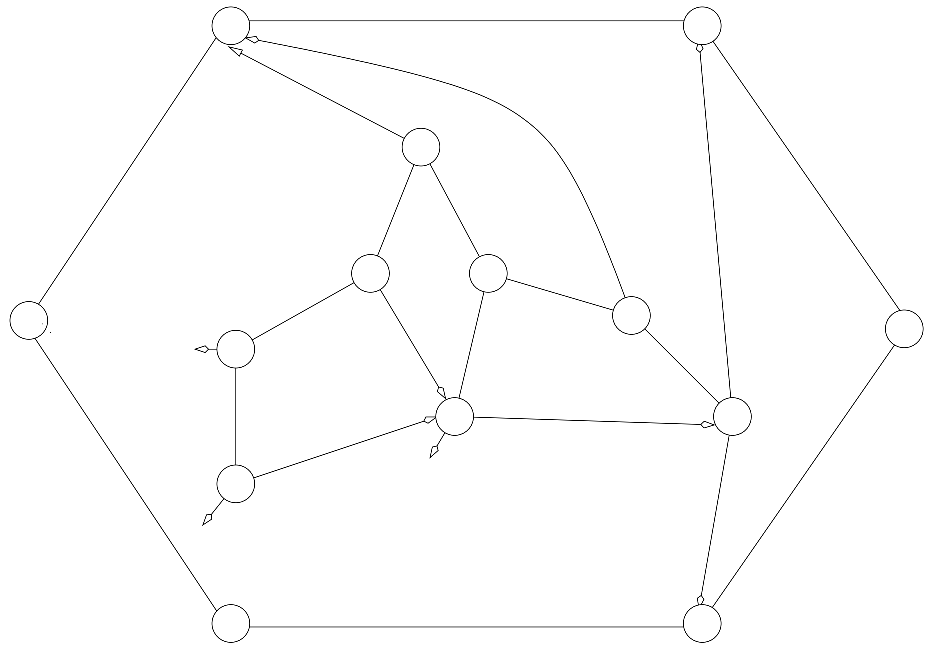


Partial closure : when all local closures are done the numbers k of remaining leaves and ℓ of sides of inner edges in the infinite face satisfy $2k-\ell=6$.

Complete closure : The remaining leaves can be attached to the vertices of hexagon so as to form faces of degree 4.

Binary trees and the closure

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n$ edges.

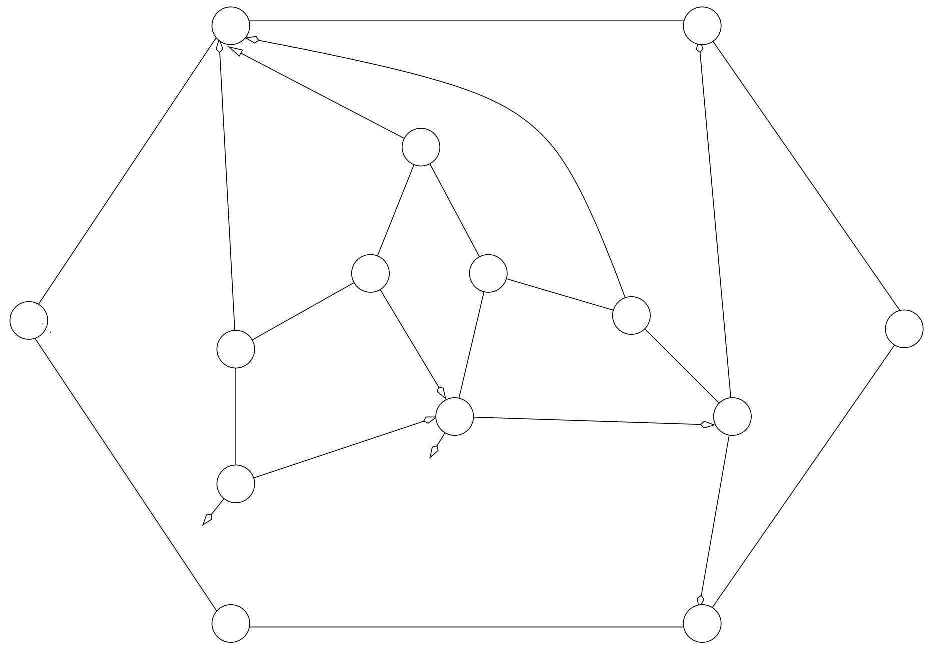


Partial closure : when all local closures are done the numbers k of remaining leaves and ℓ of sides of inner edges in the infinite face satisfy $2k-\ell=6$.

Complete closure : The remaining leaves can be attached to the vertices of hexagon so as to form faces of degree 4.

BINARY TREES AND THE CLOSURE

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n$ edges.

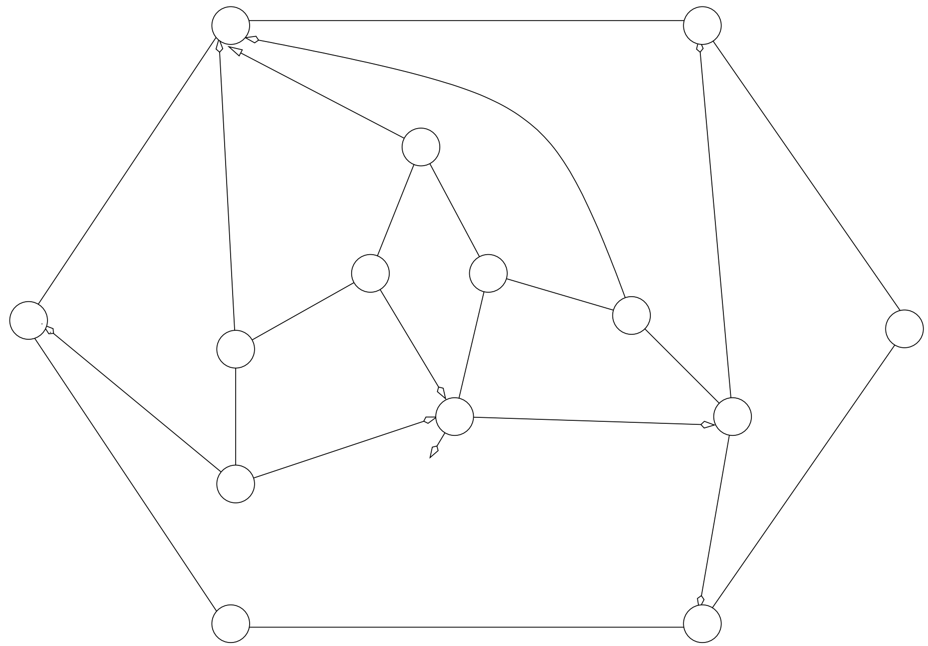


Partial closure : when all local closures are done the numbers k of remaining leaves and ℓ of sides of inner edges in the infinite face satisfy $2k-\ell=6$.

Complete closure : The remaining leaves can be attached to the vertices of hexagon so as to form faces of degree 4.

BINARY TREES AND THE CLOSURE

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n$ edges.



Partial closure : when all local closures are done the numbers k of remaining leaves and ℓ of sides of inner edges in the infinite face satisfy $2k-\ell=6$.

Complete closure : The remaining leaves can be attached to the vertices of hexagon so as to form faces of degree 4.

Binary trees and the closure

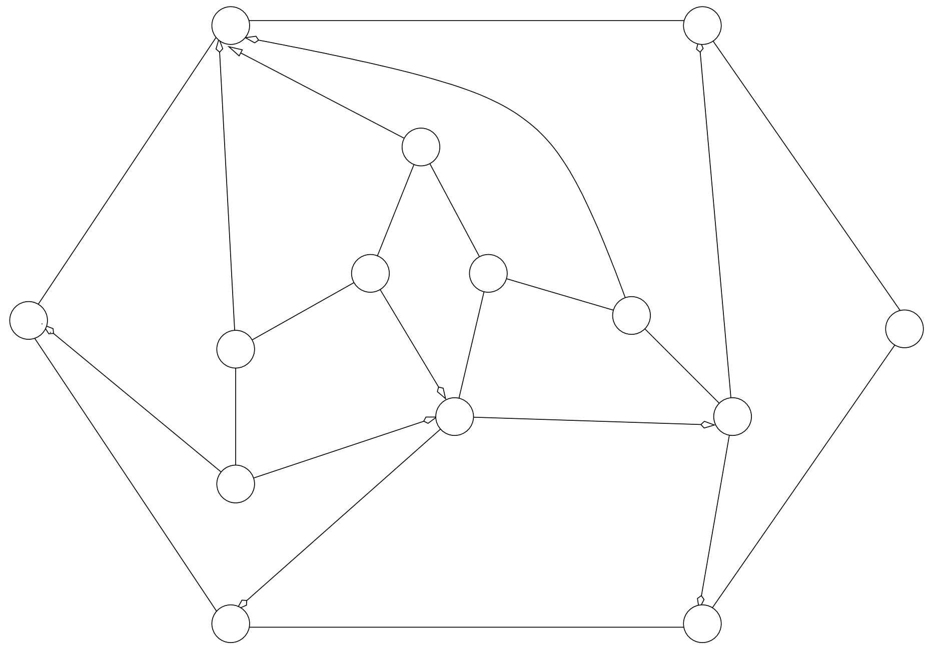
Consider a tree of $|B_n|$:

— n inner vertices,

— $n-1$ inner edges,

— $n+2$ leaves (root included),

— and $2n$ edges.

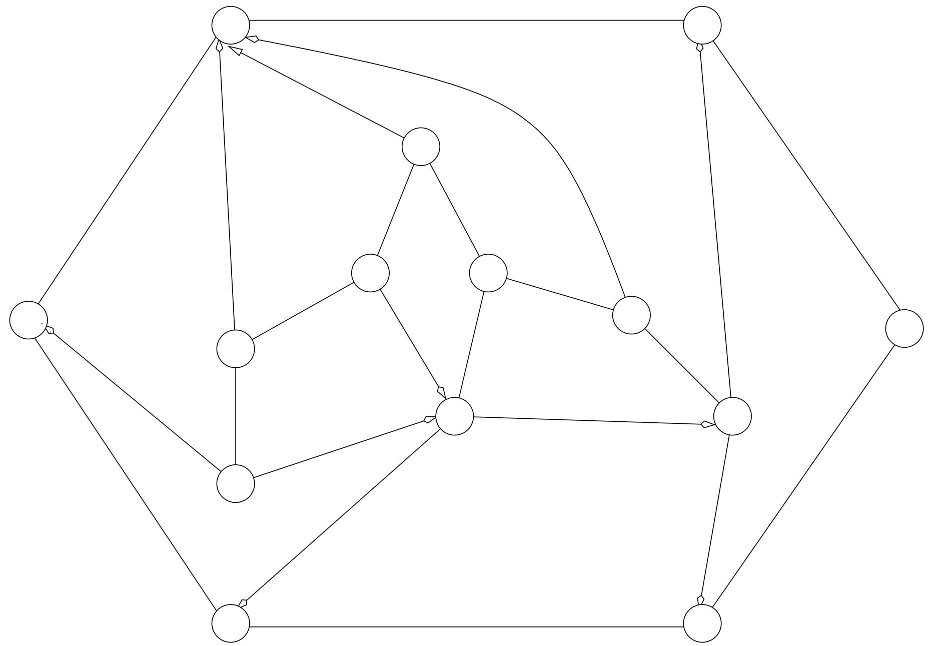


Partial closure : when all local closures are done the numbers k of remaining leaves and ℓ of sides of inner edges in the infinite face satisfy $2k-\ell=6$.

Complete closure : The remaining leaves can be attached to the vertices of hexagon so as to form faces of degree 4.

Binary trees and the closure

Consider a tree of $|B_n|$:
— n inner vertices,
— $n-1$ inner edges,
— $n+2$ leaves (root included),
— and $2n$ edges.

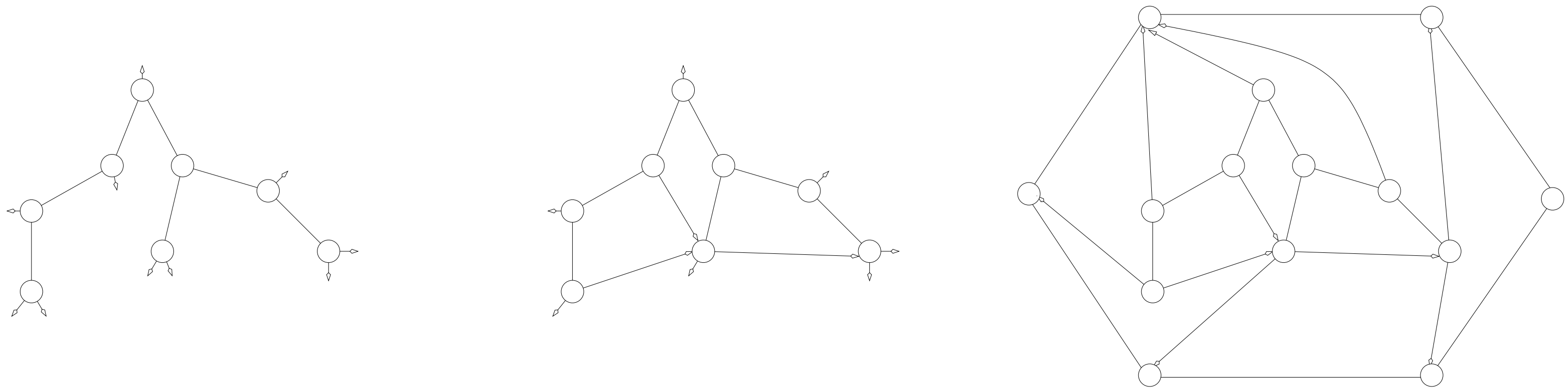


Partial closure : when all local closures are done the numbers k of remaining leaves and ℓ of sides of inner edges in the infinite face satisfy $2k-\ell=6$.

Complete closure : The remaining leaves can be attached to the vertices of hexagon so as to form faces of degree 4.

Up to rotation of the hexagon there is a unique way to do this.

Binary trees and dissection of a hexagon

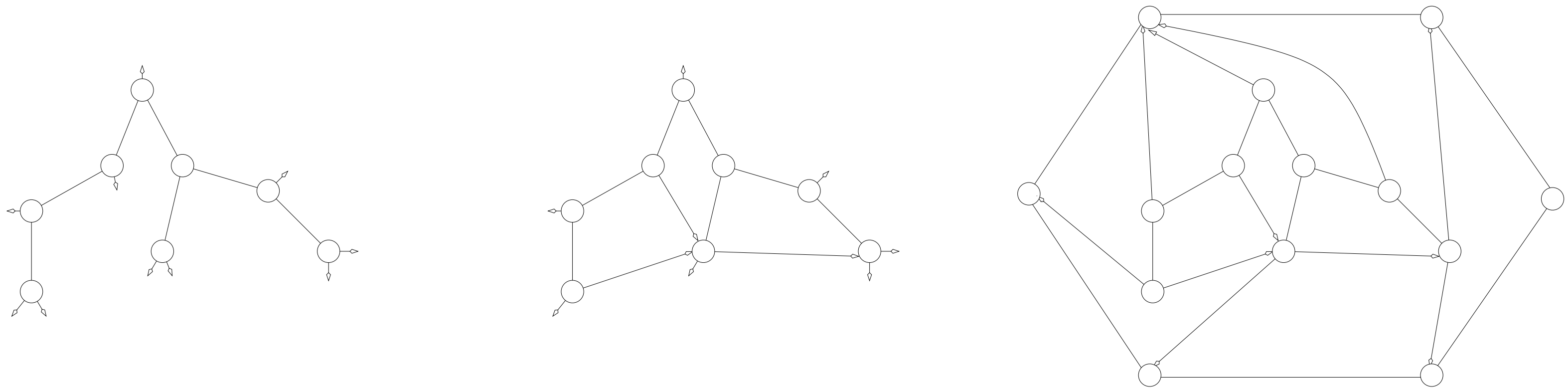


Theorem (Fusy–Poulalhon–Schaeffer 04).

Closure is a bijection between

- unrooted binary trees with n inner nodes
- irreducible dissection of an hexagon with n internal vertices.

BINARY TREES AND DISSECTION OF A HEXAGON



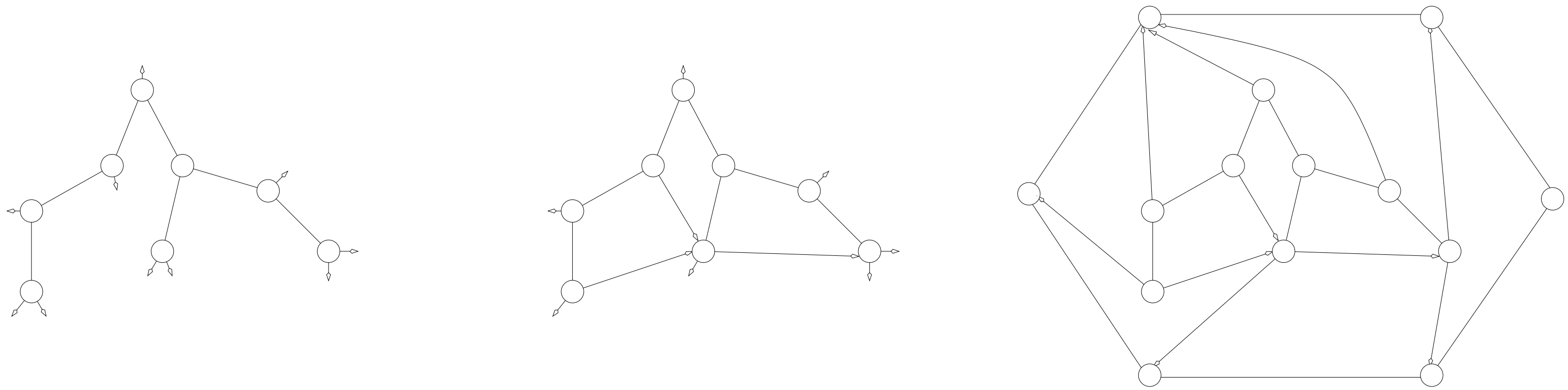
Theorem (Fusy–Poulalhon–Schaeffer 04).

Closure is a bijection between

- unrooted binary trees with n inner nodes
- irreducible dissection of an hexagon with n internal vertices.

Impossibility of 4-cycle is checked by a counting argument.

Binary trees and dissection of a hexagon



Theorem (Fusy–Poulalhon–Schaeffer 04).

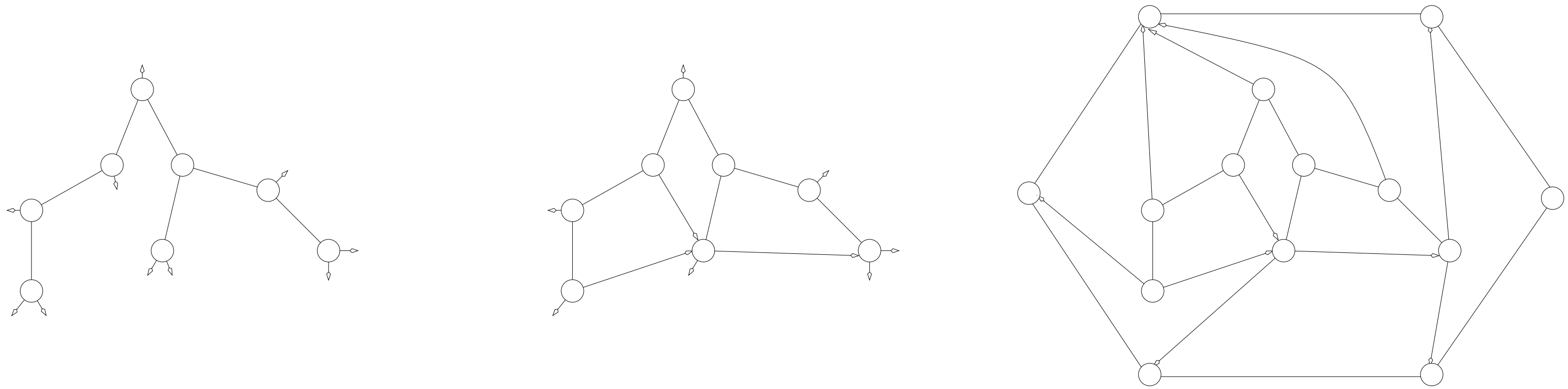
Closure is a bijection between

- unrooted binary trees with n inner nodes
- irreducible dissection of an hexagon with n internal vertices.

Impossibility of 4-cycle is checked by a counting argument.

Orient half-edges of the tree away from a vertex v inside.

Binary trees and dissection of a hexagon



Theorem (Fusy–Poulalhon–Schaeffer 04).

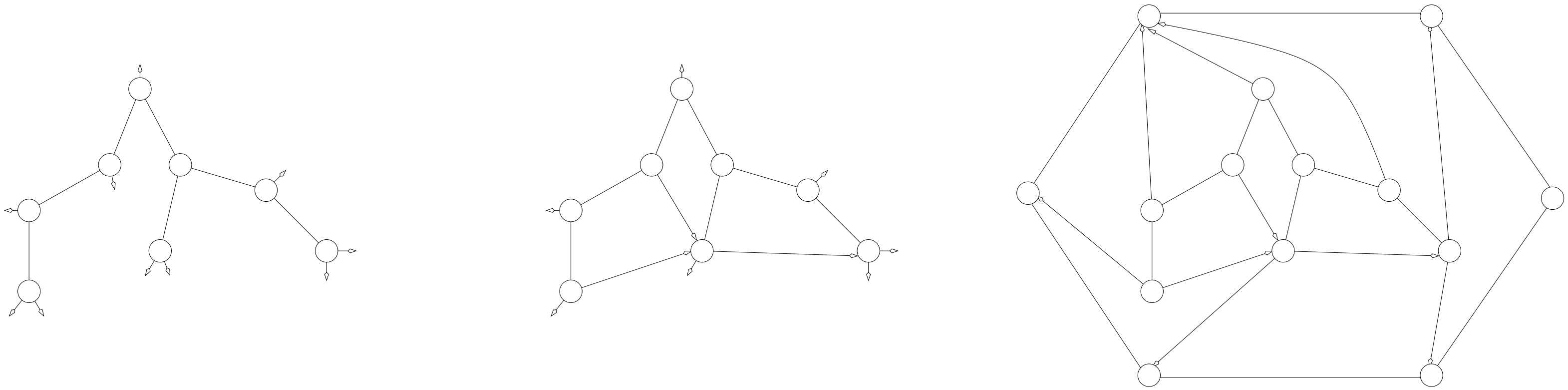
Closure is a bijection between

- unrooted binary trees with n inner nodes
- irreducible dissection of an hexagon with n internal vertices.

Impossibility of 4-cycle is checked by a counting argument.

Orient half-edges of the tree away from a vertex v inside.
Count outgoing half-edges inside the 4-cycle : $3 + 2k \neq 2(k+1)$

Binary trees and dissection of a hexagon

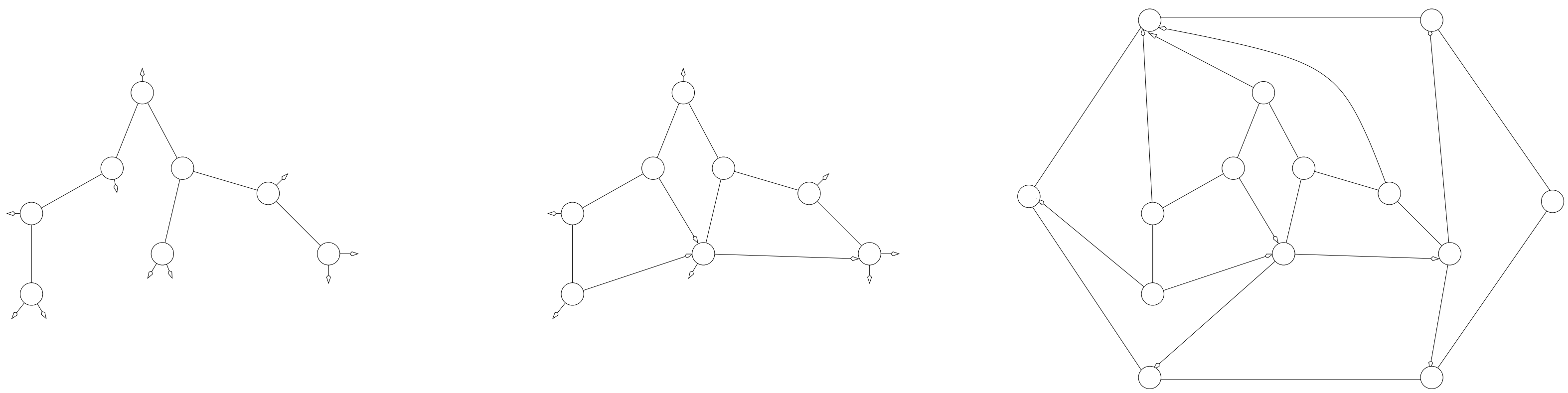


Theorem (Fusy–Poulalhon–Schaeffer 04).

Closure is a bijection between

- unrooted binary trees with n inner nodes
- irreducible dissection of an hexagon with n internal vertices.

Binary trees and dissection of a hexagon



Theorem (Fusy–Poulalhon–Schaeffer 04).

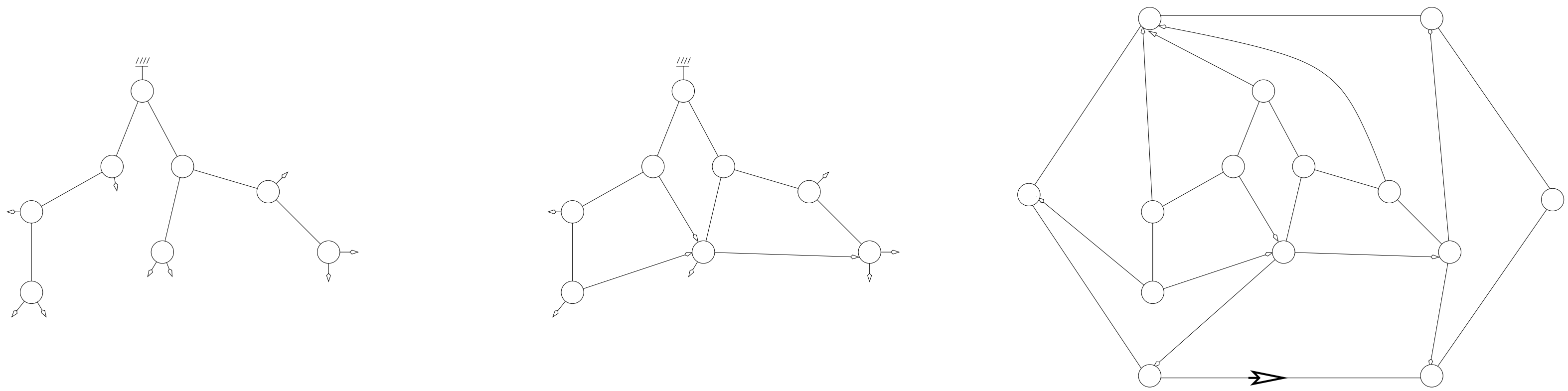
Closure is a bijection between

- unrooted binary trees with n inner nodes
- irreducible dissection of an hexagon with n internal vertices.

Corollaries : The number of rooted dissections of a hexagon with n inner vertices is

$$\frac{6}{n+2}\frac{1}{n+1}\binom{2n}{n}.$$

BINARY TREES AND DISSECTION OF A HEXAGON



Theorem (Fusy–Poulalhon–Schaeffer 04).

Closure is a bijection between

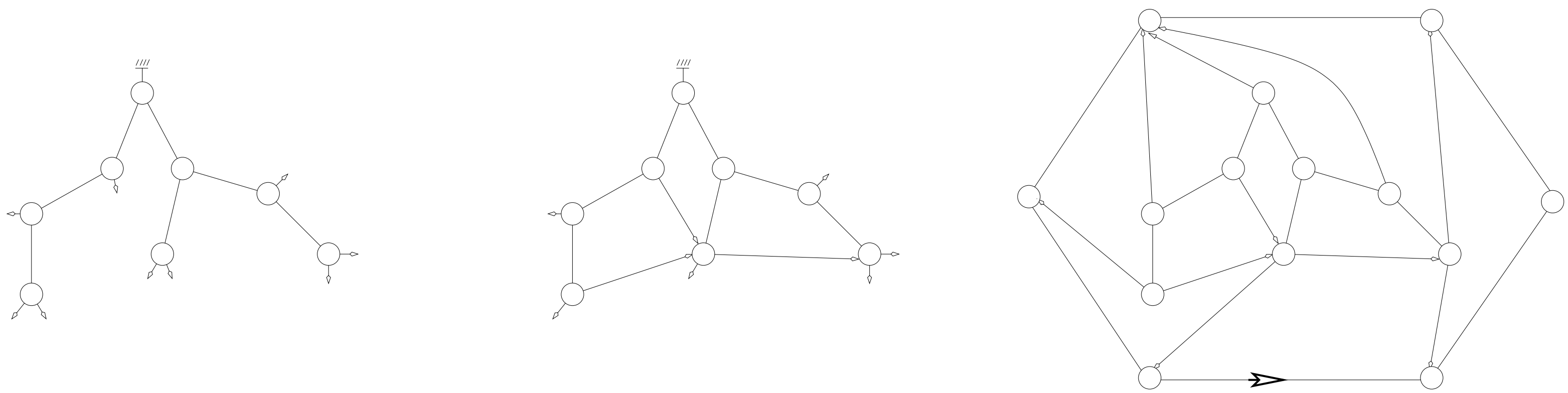
- unrooted binary trees with n inner nodes
- irreducible dissection of an hexagon with n internal vertices.

Corollaries : The number of rooted dissections of a hexagon with n inner vertices is

$$\frac{6}{n+2}\frac{1}{n+1}\binom{2n}{n}.$$

Encode a dissection of the hexagon by the $2n$ bits coding the tree.

BINARY TREES AND DISSECTION OF A HEXAGON



Theorem (Fusy–Poulalhon–Schaeffer 04).

Closure is a bijection between

- unrooted binary trees with n inner nodes
- irreducible dissection of an hexagon with n internal vertices.

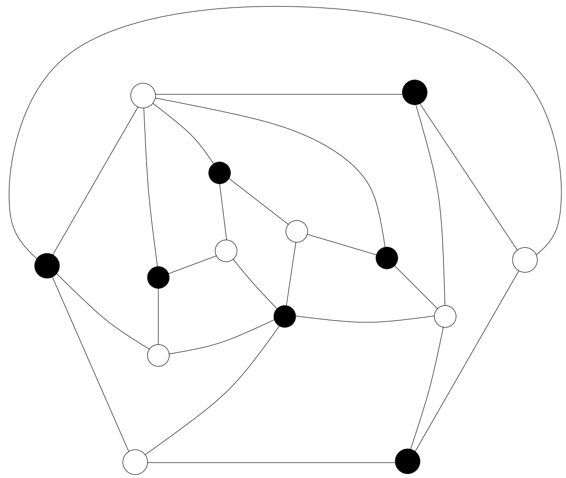
Corollaries : The number of rooted dissections of a hexagon with n inner vertices is

$$\frac{6}{n+2}\frac{1}{n+1}\binom{2n}{n}.$$

Sample uniform random rooted dissections in linear time.

DISSECTION OF A HEXAGON OR OF A SQUARE

Consider an irreducible dissection associated with a 3-c planar graph.



Removing one edge yields an irreducible dissection of a hexagon.

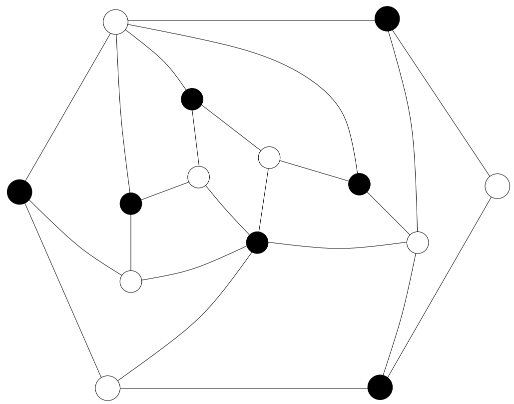
$\Rightarrow$ our approach thus immediately yields a code for 3-c planar graphs.

this code is "optimal" again..

DISSECTION OF A HEXAGON OR OF A SQUARE

Consider an irreducible dissection associated with a 3-c planar graph.

Removing one edge yields an irreducible dissection of a hexagon.

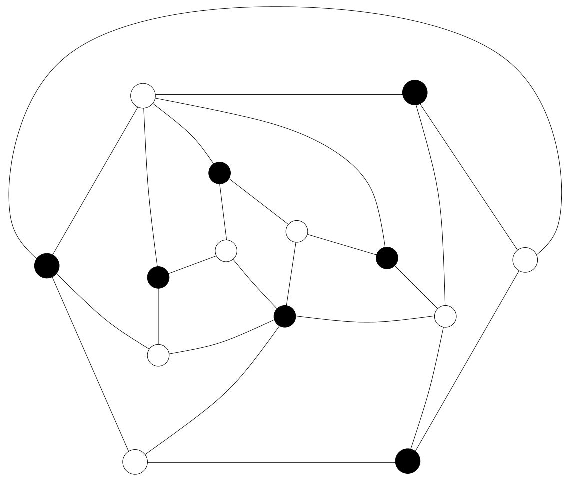


$\Rightarrow$ our approach thus immediately yields a code for 3-c planar graphs.

this code is “optimal” again..

DISSECTION OF A HEXAGON OR OF A SQUARE

Consider an irreducible dissection associated with a 3-c planar graph.



Removing one edge yields an irreducible dissection of a hexagon.

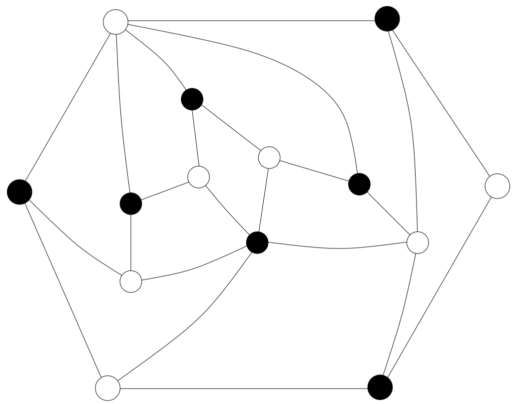
$\Rightarrow$ our approach thus immediately yields a code for 3-c planar graphs.

this code is “optimal” again..

Dissection of a hexagon or of a square

Consider an irreducible dissection associated with a 3-c planar graph.

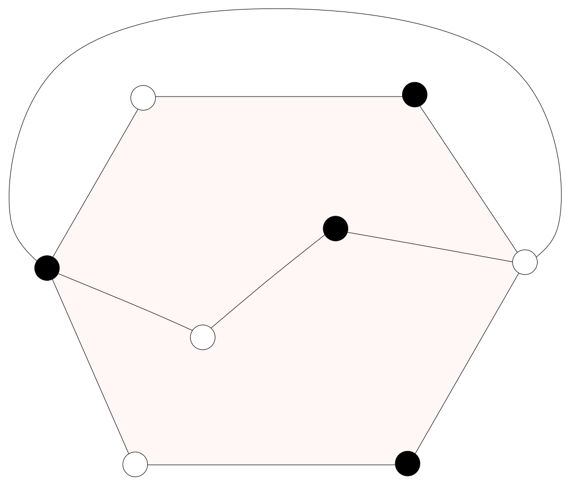
Removing one edge yields an irreducible dissection of a hexagon.



$\Rightarrow$ our approach thus immediately yields a code for 3-c planar graphs.

this code is “optimal” again..

DISSECTION OF A HEXAGON OR OF A SQUARE



Consider an irreducible dissection associated with a 3-c planar graph.

Removing one edge yields an irreducible dissection of a hexagon.

$\Rightarrow$ our approach thus immediately yields a code for 3-c planar graphs.

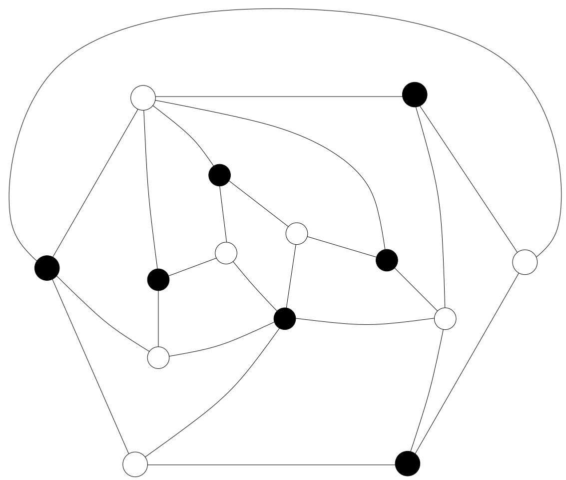
this code is “optimal” again..

Conversely : an irreducible dissection of a hexagon $\Rightarrow$ an irreducible dissection of a square iff there was not a diagonal of length 3.

Dissection of a hexagon or of a square

Consider an irreducible dissection associated with a 3-c planar graph.

Removing one edge yields an irreducible dissection of a hexagon.



$\Rightarrow$ our approach thus immediately yields a code for 3-c planar graphs.

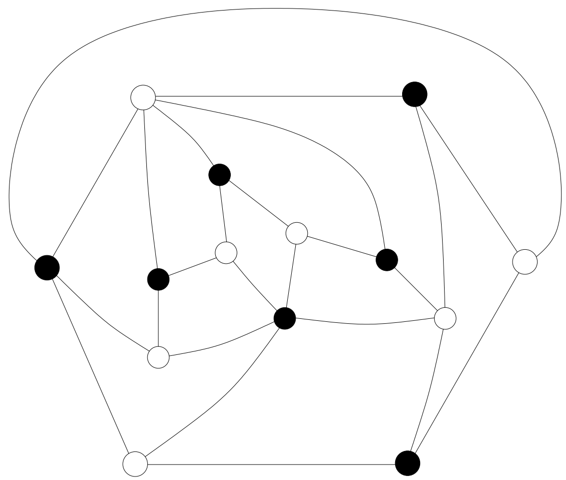
this code is "optimal" again..

Conversely : an irreducible dissection of a hexagon $\Rightarrow$ an irreducible dissection of a square iff there was not a diagonal of length 3.

DISSECTION OF A HEXAGON OR OF A SQUARE

Consider an irreducible dissection associated with a 3-c planar graph.

Removing one edge yields an irreducible dissection of a hexagon.



⇒ our approach thus immediately yields a code for 3-c planar graphs.

this code is “optimal” again..

Conversely : an irreducible dissection of a hexagon ⇒ an irreducible dissection of a square iff there was not a diagonal of length 3.

⇒ sampling by rejection : try to add an edge, restart from scratch if not ok.

Conclusion of Part 3.

- { 3-connected planar graphs } $\approx$
  { Irreducible dissections of a hexagon } $\equiv$ { binary trees }.

- Corrollaries :

  — a formula for rooted irreducible dissections,

  — a linear time random sampler for 3-c planar graphs,
  $\Rightarrow$ improvment for the generator for planar graphs of Bodirsky et al.

  — and a compact code for polyhedral meshes with spherical topology.

Conclusion of Part 3.

- { 3-connected planar graphs } $\approx$
  { Irreducible dissections of a hexagon } $\equiv$ { binary trees }.

- Corrollaries :

  — a formula for rooted irreducible dissections,

  — a linear time random sampler for 3-c planar graphs,
  $\Rightarrow$ improvment for the generator for planar graphs of Bodirsky et al.

  — and a compact code for polyhedral meshes with spherical topology.

  but until now I did not show how to compute the code.

Part 4. Minimal α-orientations and coding.

(a glimpse of the machinery behind)

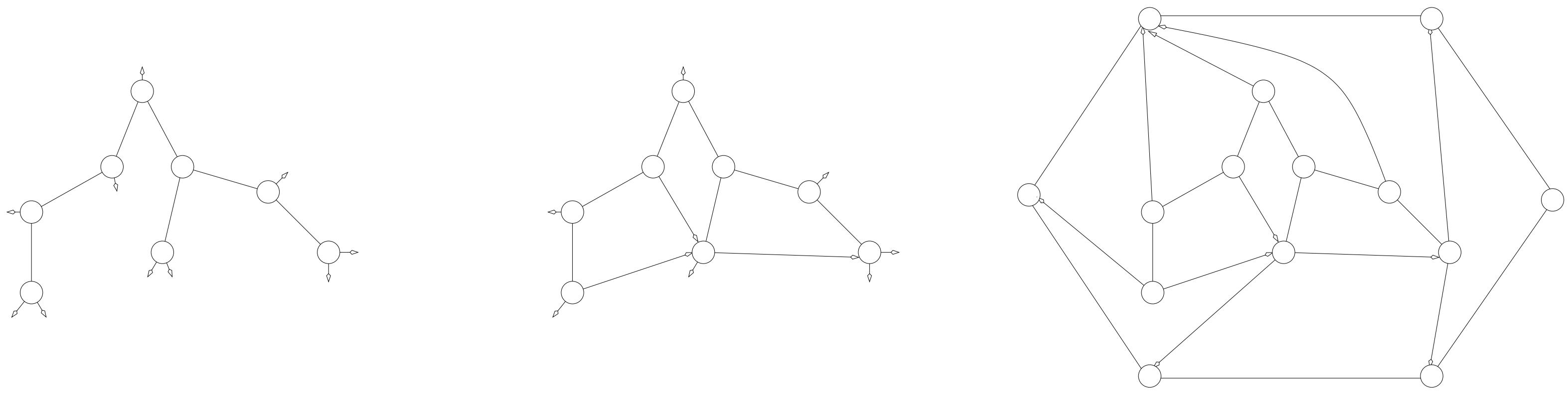
CASTING for this part :

- orientations
- derived map

INSPIRATION for this part

de Frayssex, Ossona de Mendez, Brehm, Felsner...

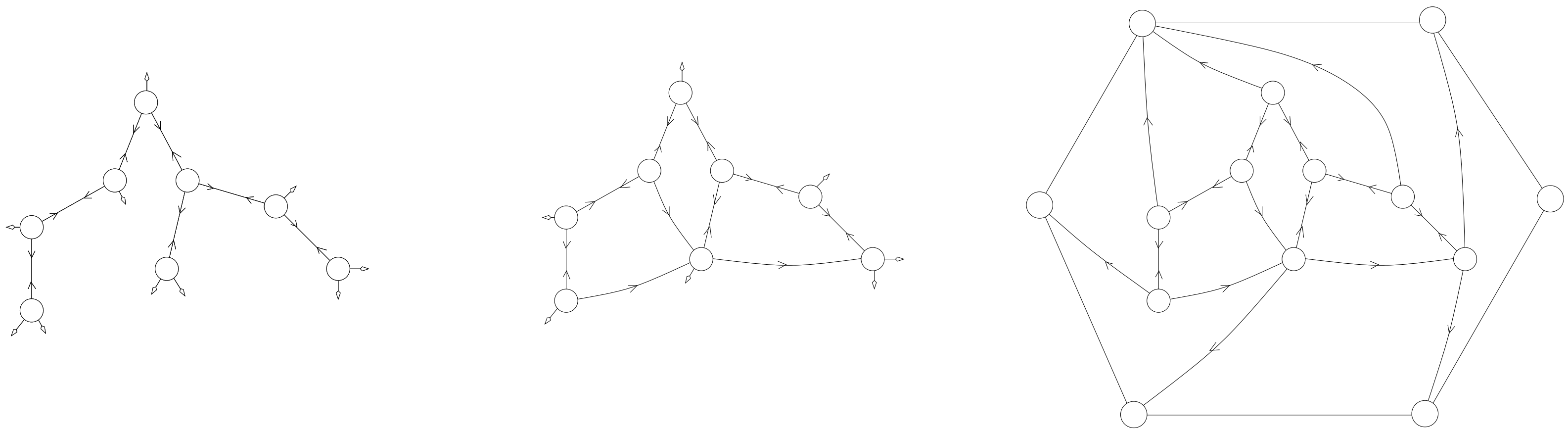
Return to the main theorem...



Closure is a bijection between unrooted binary trees with n inner nodes and irreducible dissection of an hexagon with n internal vertices.

Orient all half-edges of the binary tree $\Rightarrow$ an “orientation” of the dissection.

Return to the main theorem...

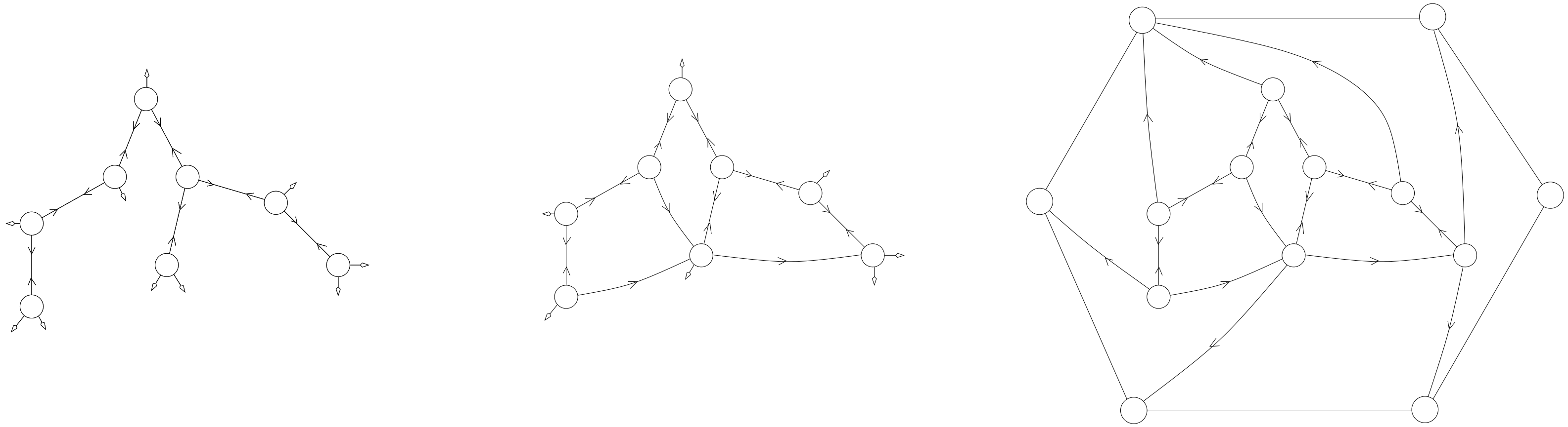


Closure is a bijection between unrooted binary trees with n inner nodes and irreducible dissection of an hexagon with n internal vertices.

Orient all half-edges of the binary tree $\Rightarrow$ an "orientation" of the dissection.

All internal vertices have out-degree 3.

RETURN TO THE MAIN THEOREM...



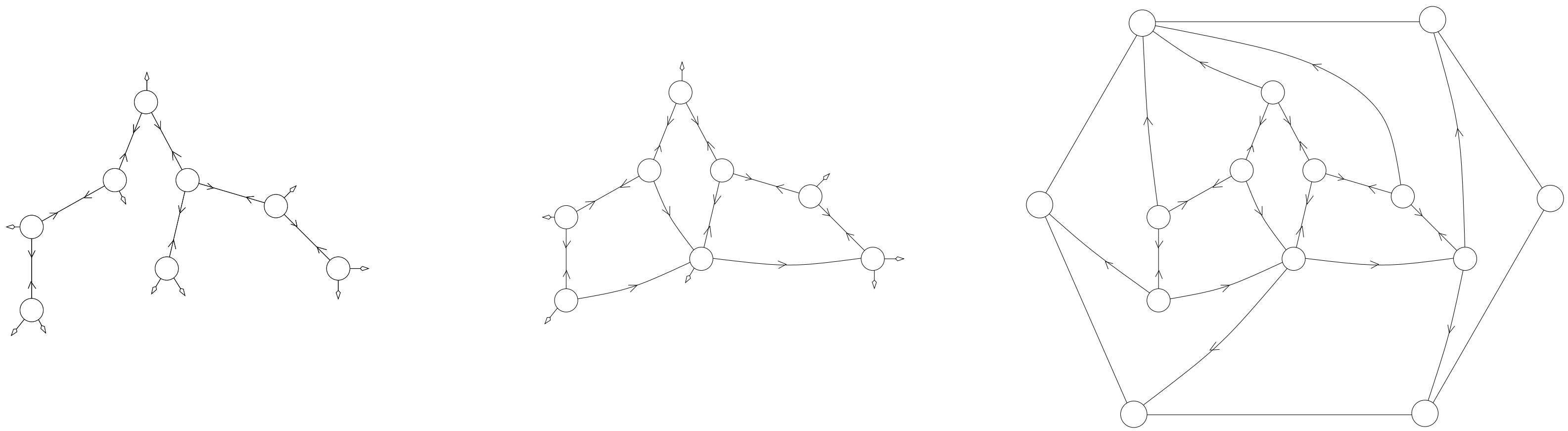
Closure is a bijection between unrooted binary trees with n inner nodes and irreducible dissection of an hexagon with n internal vertices.

Orient all half-edges of the binary tree $\Rightarrow$ an “orientation” of the dissection.

All internal vertices have out-degree 3.

Proposition. By construction, there are no cw circuits.

Return to the main theorem...



Closure is a bijection between unrooted binary trees with n inner nodes and irreducible dissection of an hexagon with n internal vertices.

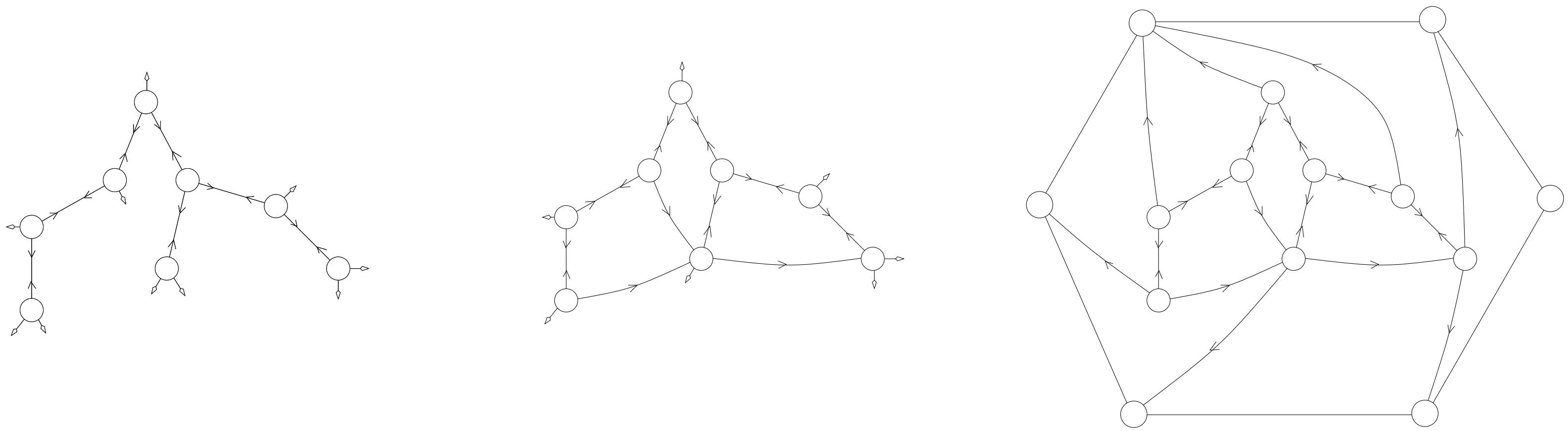
Orient all half-edges of the binary tree $\Rightarrow$ an “orientation” of the dissection.

All internal vertices have out-degree 3.

Proposition. By construction, there are no cw circuits.

Conversely, in a “3-orientation” without cw circuit, edges $\rightarrow\leftarrow$ form a tree.

RETURN TO THE MAIN THEOREM...



Refined Theorem : Closure is a bijection between unrooted binary trees and irreducible dissections of a hexagon without cw circuits.

Orient all half-edges of the binary tree $\Rightarrow$ an "orientation" of the dissection.

All internal vertices have out-degree 3.

Proposition. By construction, there are no cw circuits.

Conversely, in a "3-orientation" without cw circuit, edges $\rightarrow\leftarrow$ form a tree.

α-ORIENTATIONS

Let α be an out-degree prescription for the vertices of a planar graph.

α-orientation = orientation of edges respecting α.

Theorem (Felsner 03, Ossona de Mendez 94)
If there exists an α-orientation, then the transformation

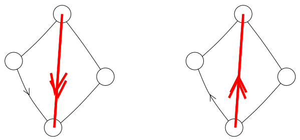
$$\text{return a cw circuit}$$

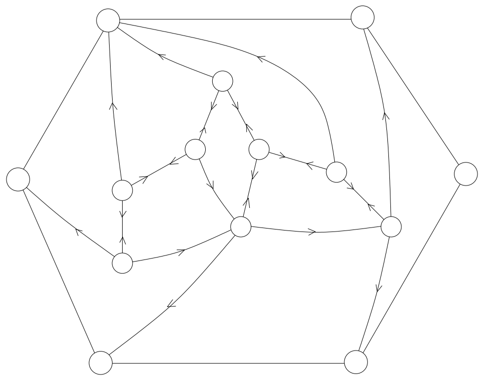
defines a distributive lattice on the set of α-orientation.

In particular :
the minimal α-orientation is the only α-orientation without cw circuits.

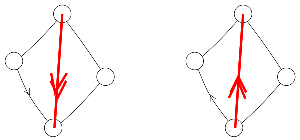
α-ORIENTATIONS AND DISSECTIONS

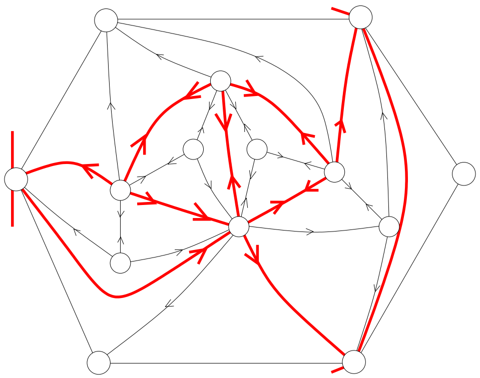
The theory does not directly apply to us : we have doubly oriented edges.





The theory does not directly apply to us : we have doubly oriented edges.

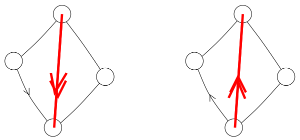


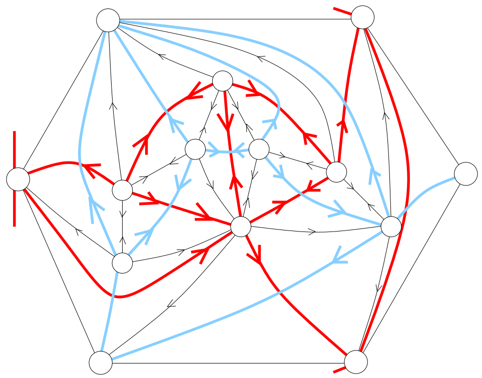


- "3-oriented" dissection $\Leftrightarrow$ α-oriented derived map : $\alpha(\circ) = 3$, $\alpha(\times) = 1$.

α-ORIENTATIONS AND DISSECTIONS

The theory does not directly apply to us : we have doubly oriented edges.





- “3-oriented” dissection $\Leftrightarrow$ α-oriented derived map : $\alpha(\circ) = 3$, $\alpha(\times) = 1$.
- prove : without cw circuits $\Leftrightarrow$ without cw circuits
- apply Felsner’s theorem to the derived map.

$\Rightarrow$ this proves that the closure send bijectively trees on dissections.

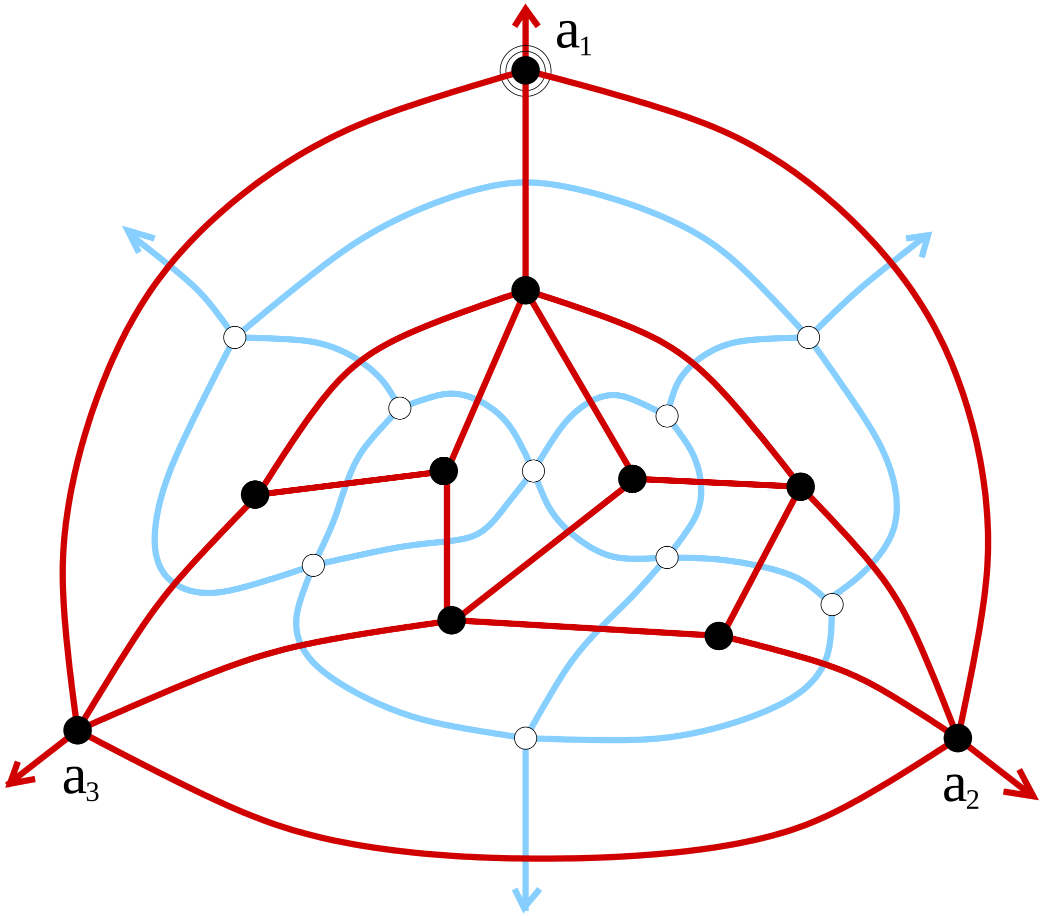
Constructing the minimal α-orientations

For coding, we still need to show that one can construct the minimal orientation in linear time.

The construction is akin to the construction for minimal 3-orientations of triangulations (Kant, Brehm).

The base line $a_2 a_3$ is fixed.

The rightmost nonseparating active vertex on the frontier is removed and incident edges are oriented.



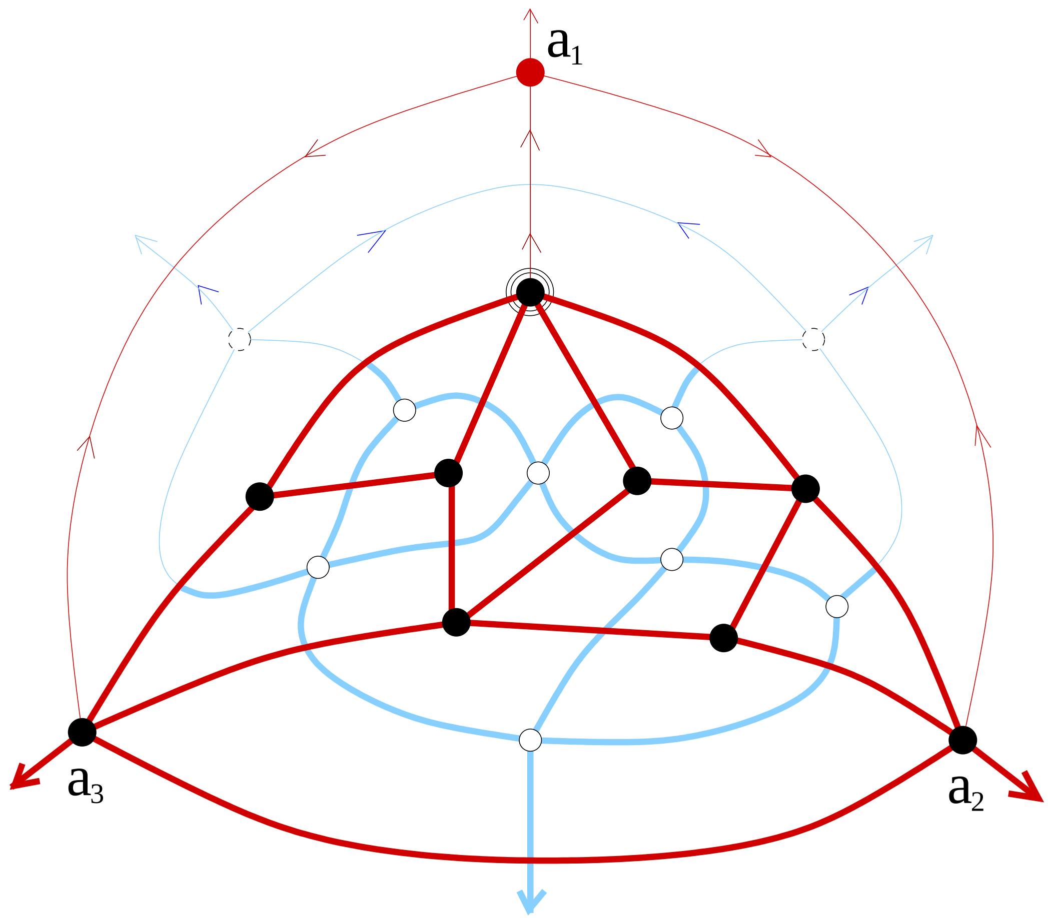
CONSTRUCTING THE MINIMAL α-ORIENTATIONS

For coding, we still need to show that one can construct the minimal orientation in linear time.

The construction is akin to the construction for minimal 3-orientations of triangulations (Kant, Brehm).

The base line $a_2 a_3$ is fixed.

The rightmost nonseparating active vertex on the frontier is removed and incident edges are oriented.



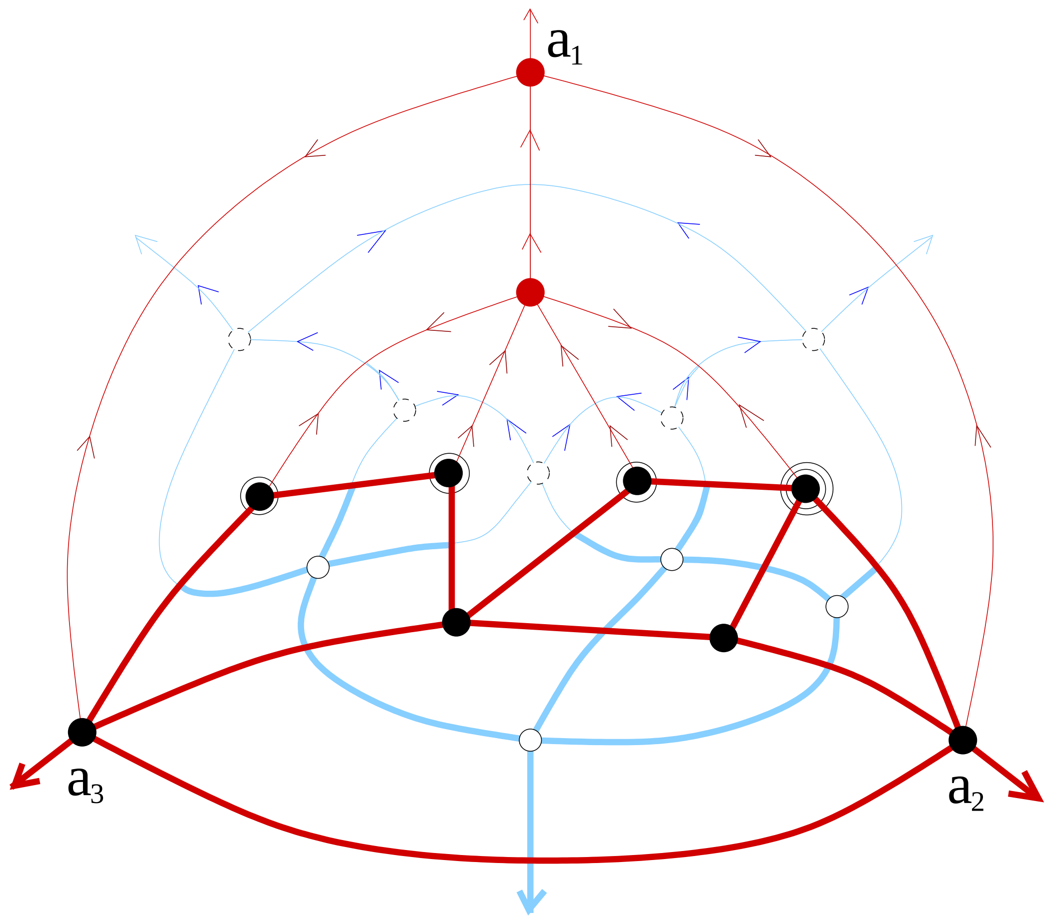
CONSTRUCTING THE MINIMAL α-ORIENTATIONS

For coding, we still need to show that one can construct the minimal orientation in linear time.

The construction is akin to the construction for minimal 3-orientations of triangulations (Kant, Brehm).

The base line $a_2 a_3$ is fixed.

The rightmost nonseparating active vertex on the frontier is removed and incident edges are oriented.



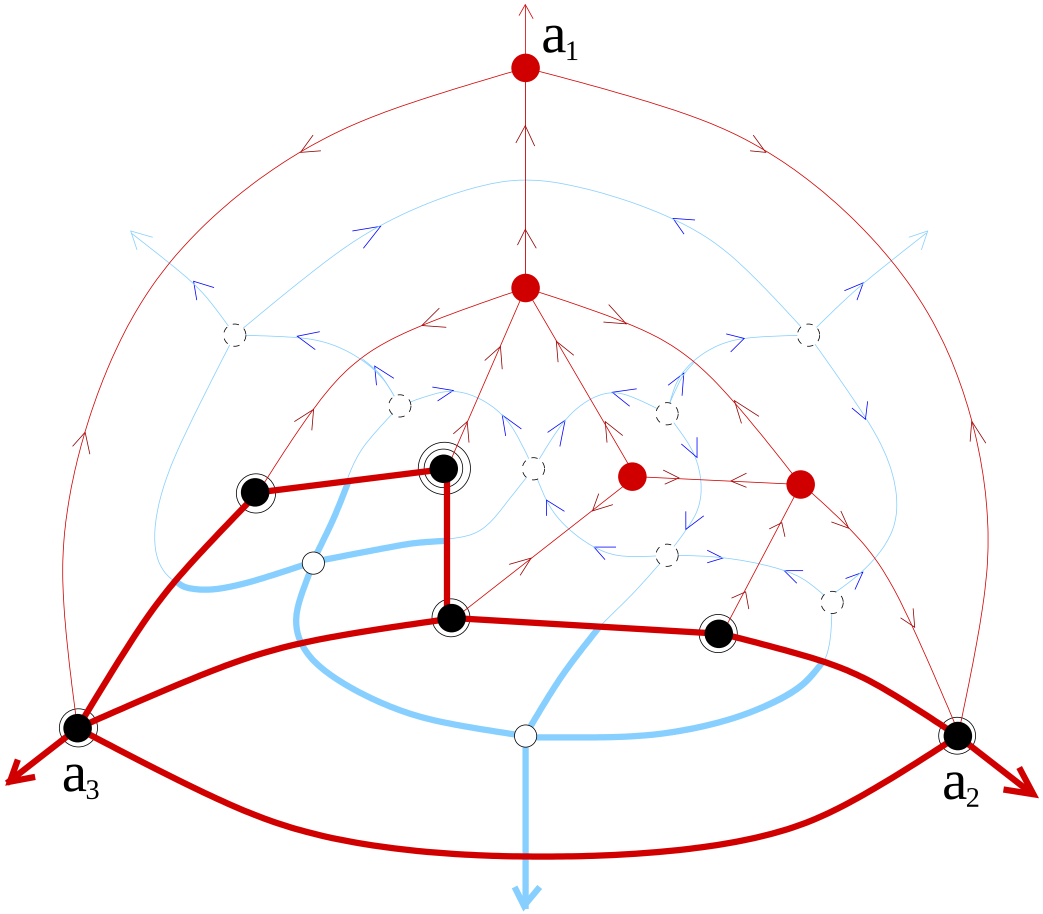
CONSTRUCTING THE MINIMAL α-ORIENTATIONS

For coding, we still need to show that one can construct the minimal orientation in linear time.

The construction is akin to the construction for minimal 3-orientations of triangulations (Kant, Brehm).

The base line a_2a_3 is fixed.

The rightmost nonseparating active vertex on the frontier is removed and incident edges are oriented.



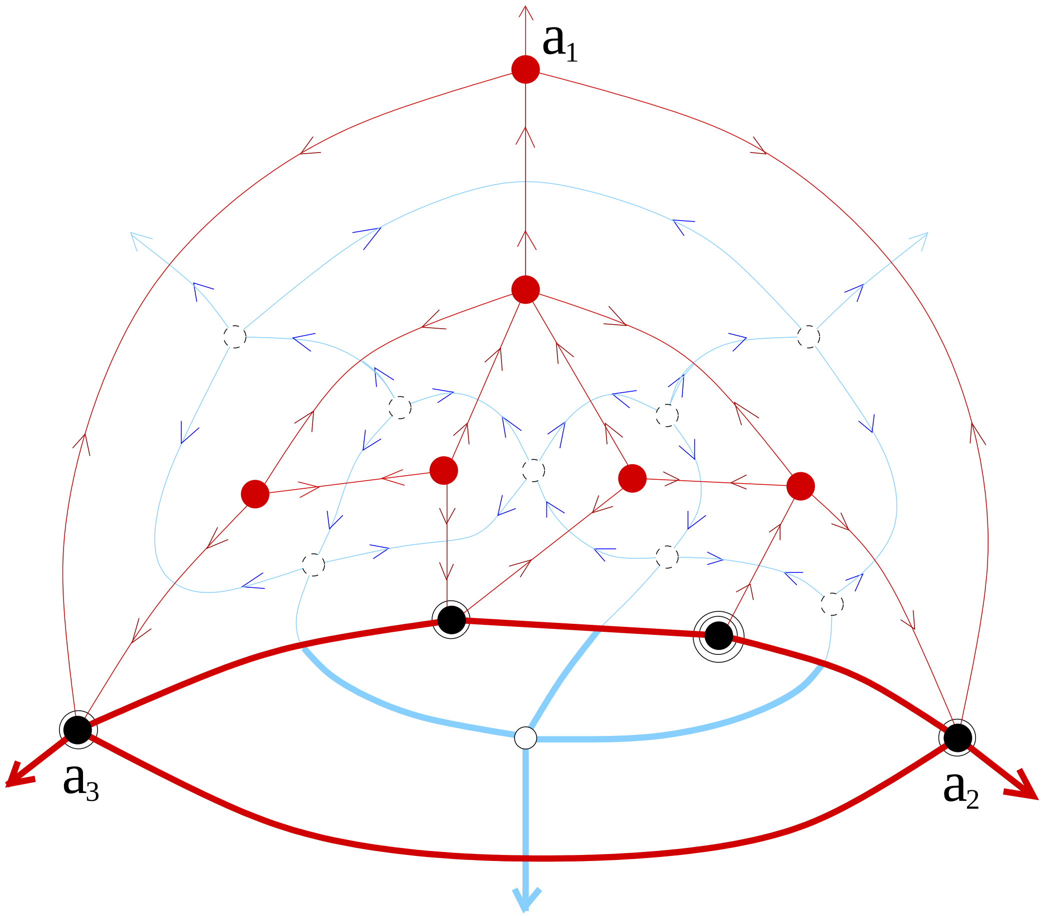
CONSTRUCTING THE MINIMAL α-ORIENTATIONS

For coding, we still need to show that one can construct the minimal orientation in linear time.

The construction is akin to the construction for minimal 3-orientations of triangulations (Kant, Brehm).

The base line a_2a_3 is fixed.

The rightmost nonseparating active vertex on the frontier is removed and incident edges are oriented.



CONSTRUCTING THE MINIMAL α-ORIENTATIONS

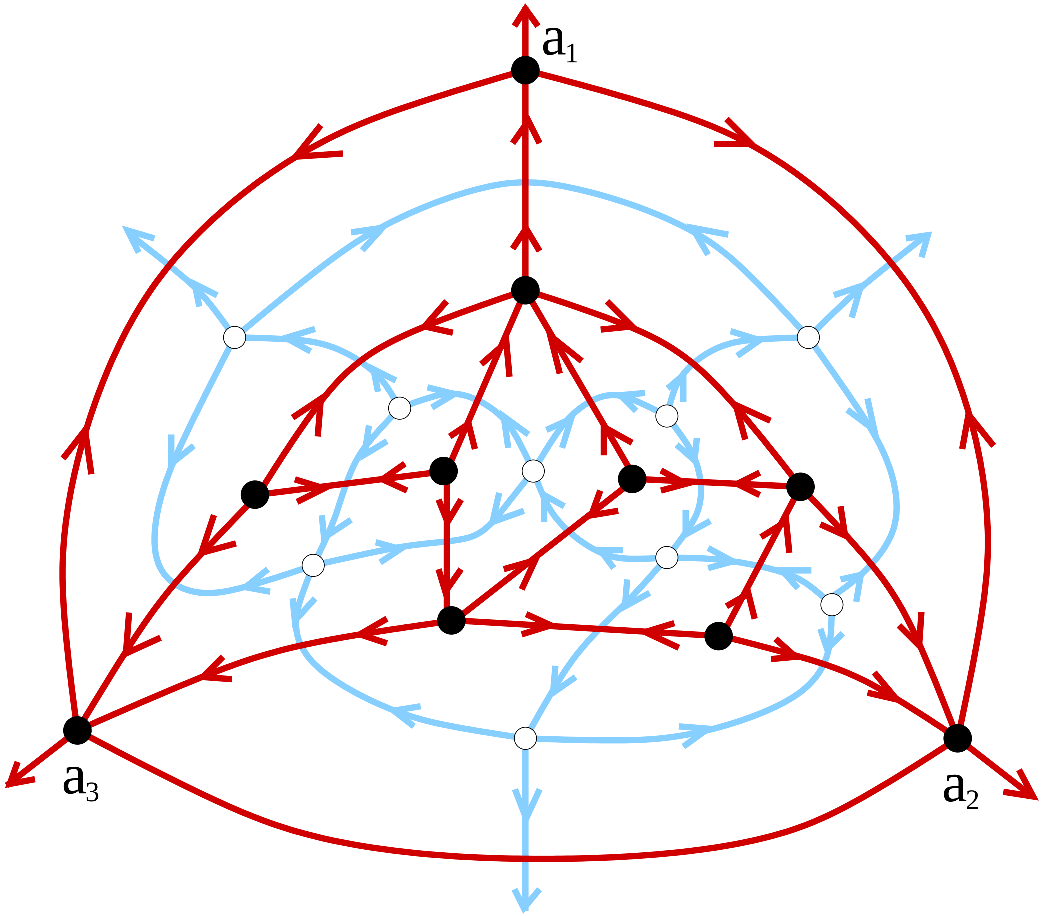
For coding, we still need to show that one can construct the minimal orientation in linear time.

The construction is akin to the construction for minimal 3-orientations of triangulations (Kant, Brehm).

The base line a_2a_3 is fixed.

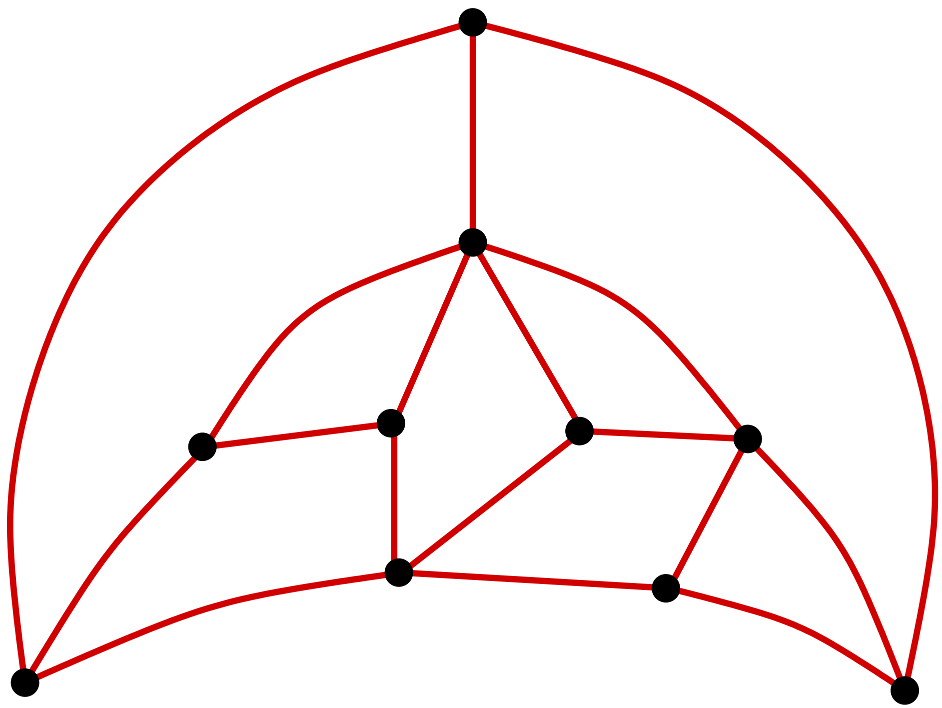
The rightmost nonseparating active vertex on the frontier is removed and incident edges are oriented.

Theorem (FPS04). This process constructs the minimal α-orientation.



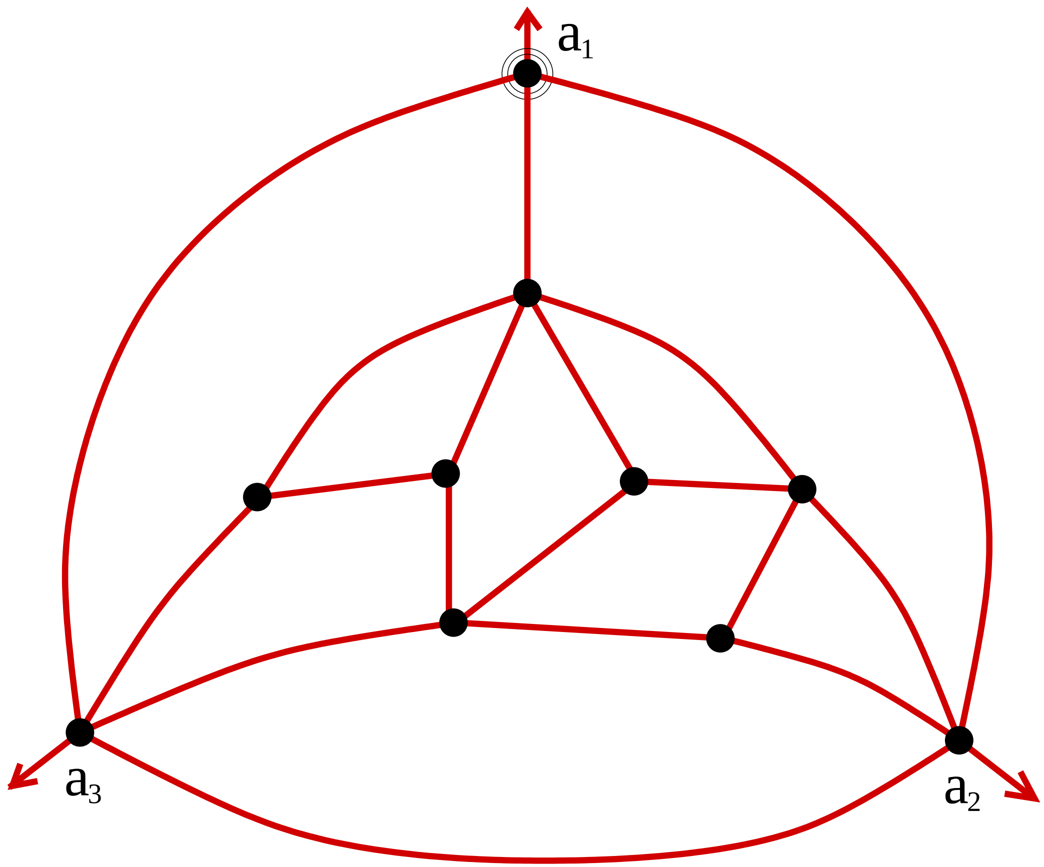
The complete encoding procedure

- Complete the 3-c graph to make it canonical.



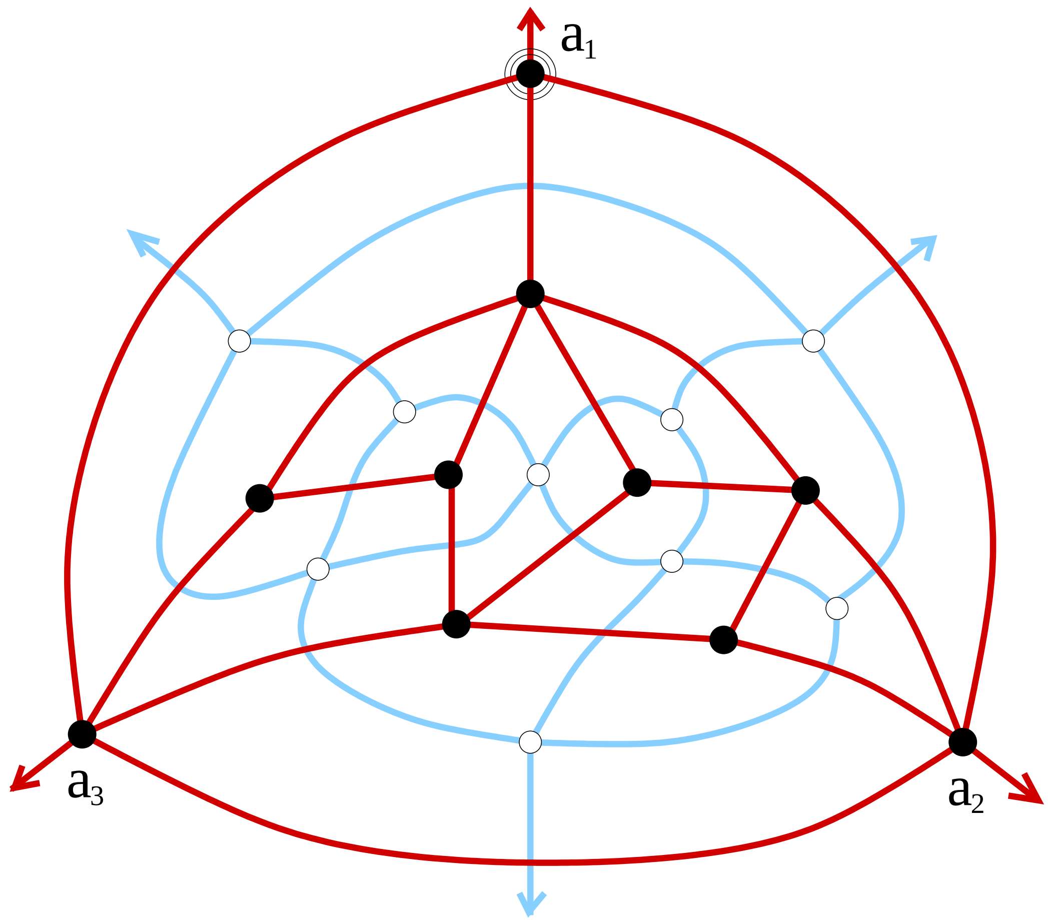
The complete encoding procedure

- Complete the 3-c graph to make it canonical.



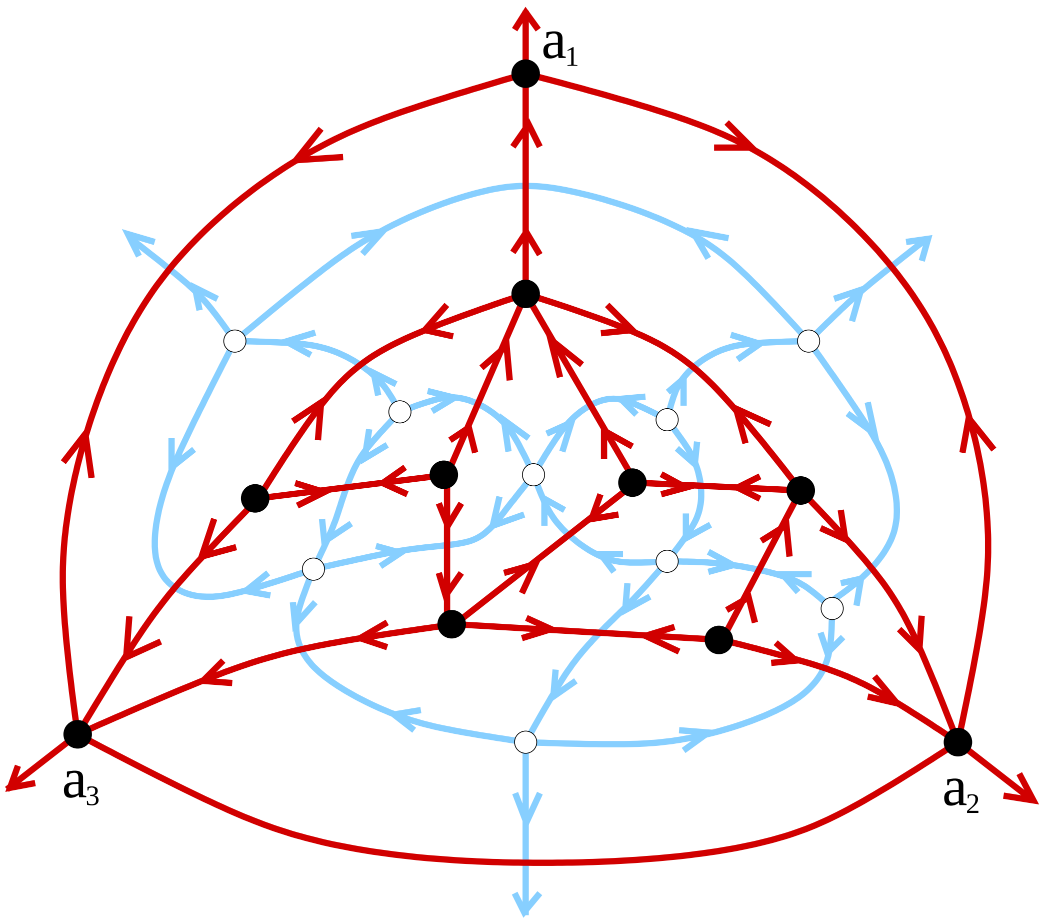
The complete encoding procedure

- Complete the 3-c graph to make it canonical.
- Superimpose the dual.



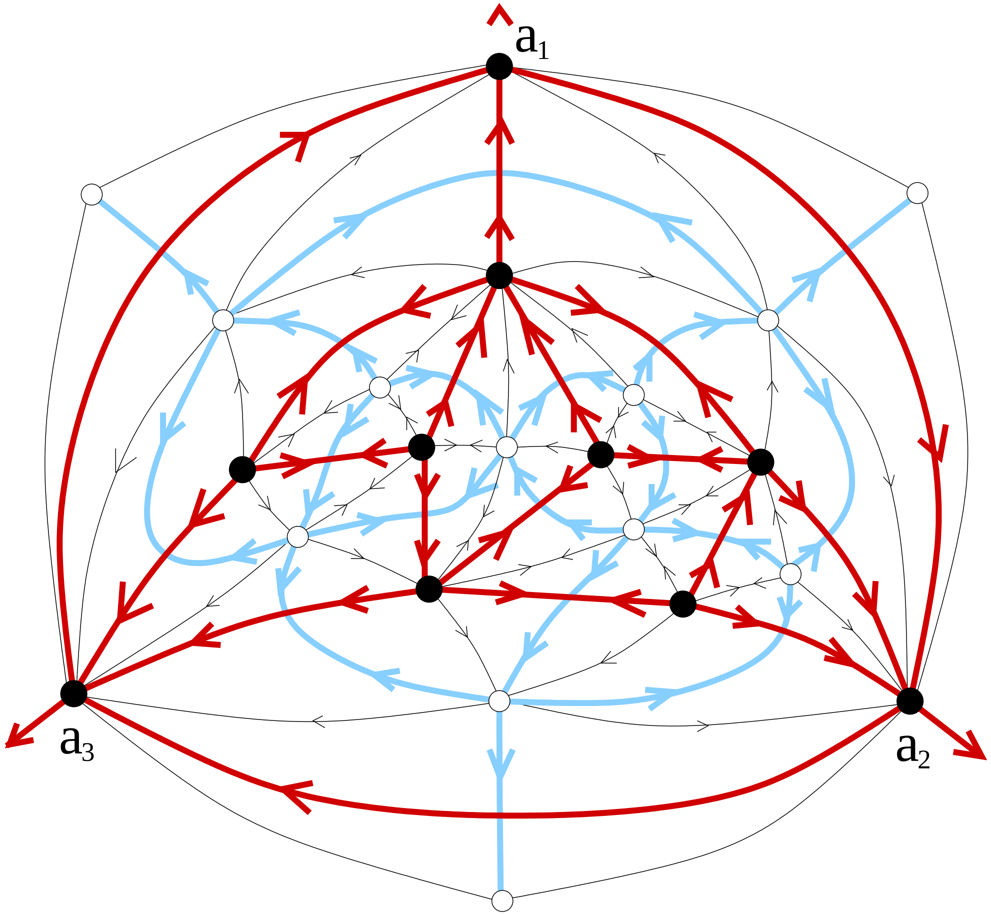
The complete encoding procedure

- Complete the 3-c graph to make it canonical.
- Superimpose the dual.
- Orient the derived map.



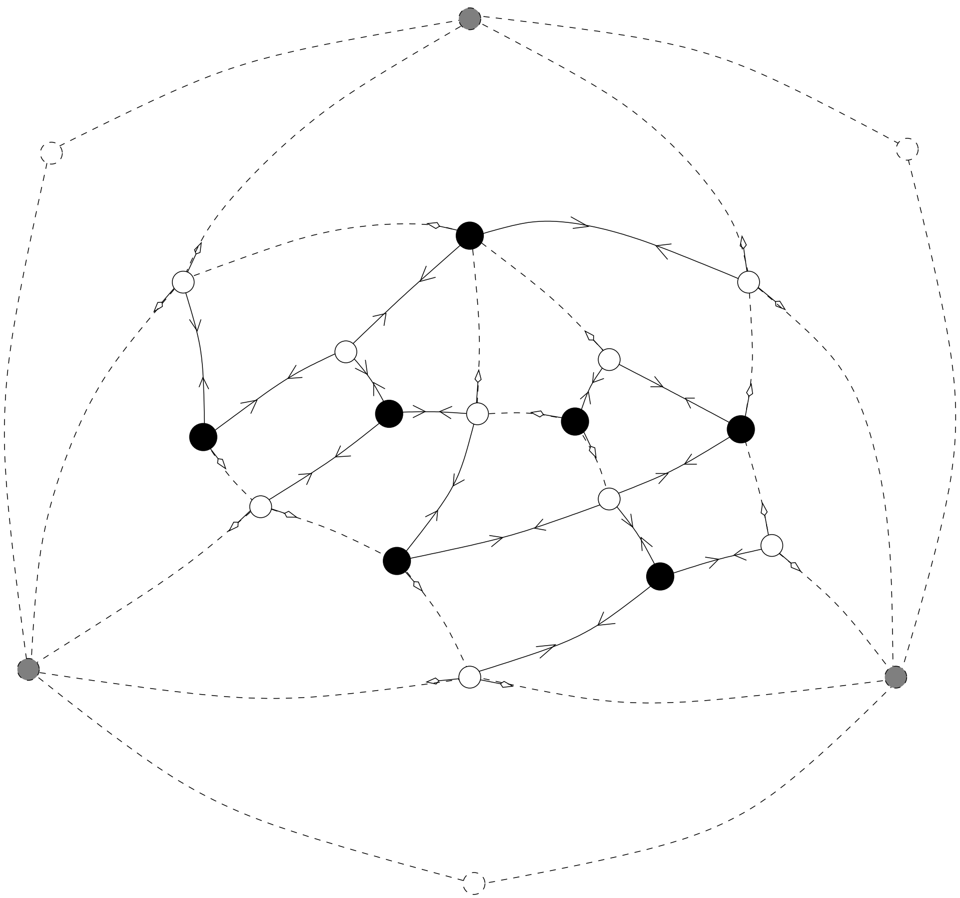
The complete encoding procedure

- Complete the 3-c graph to make it canonical.
- Superimpose the dual.
- Orient the derived map.
- Transport orientation to the dissection.



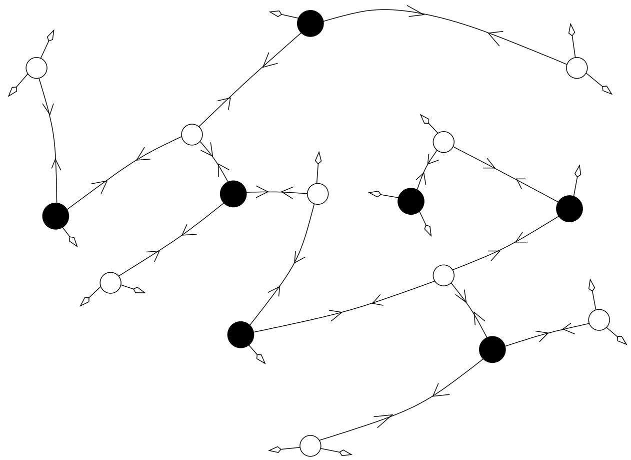
The complete encoding procedure

- Complete the 3-c graph to make it canonical.
- Superimpose the dual.
- Orient the derived map.
- Transport orientation to the dissection.
- Detach simply oriented edges.



THE COMPLETE ENCODING PROCEDURE

- Complete the 3-c graph to make it canonical.
- Superimpose the dual.
- Orient the derived map.
- Transport orientation to the dissection.
- Detach simply oriented edges.



Conclusion of Part 4.

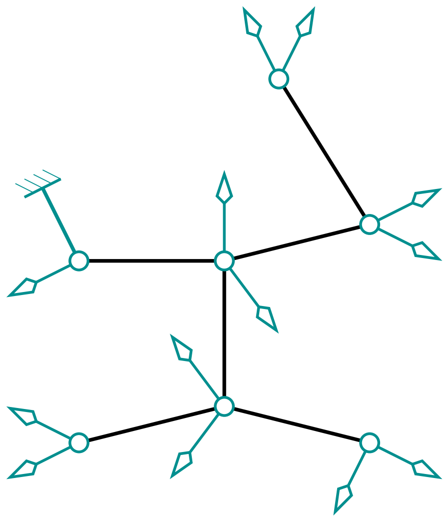
- α-orientations play a key role in proofs.
- “optimal” encoding can be performed in linear time.

Part 5. Other instances.

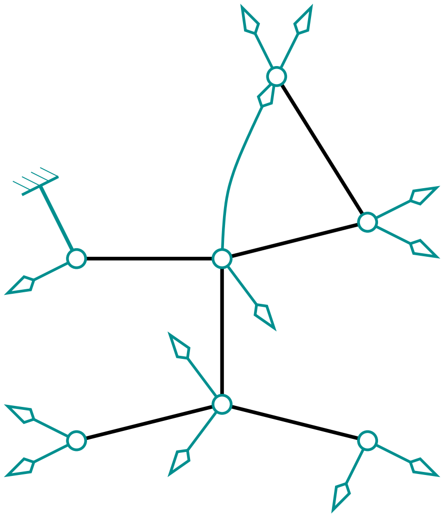
CASTING for this part :

- Triangulations and Schnyder trees [Poulalhon-Schaeffer]
- Eulerian maps and their balanced orientations [Fusy]
- Simple quadrangular dissections and 1-2-orientations [Fusy-Poulalhon]

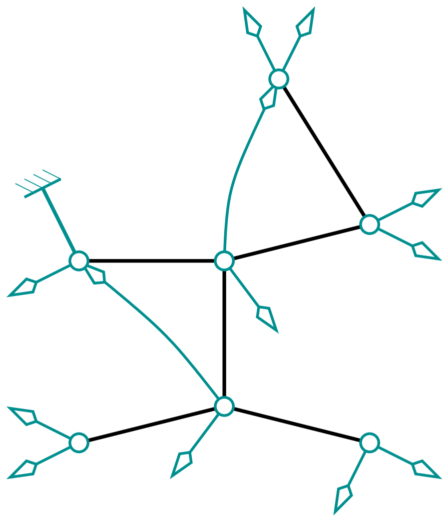
TRIANGULATIONS



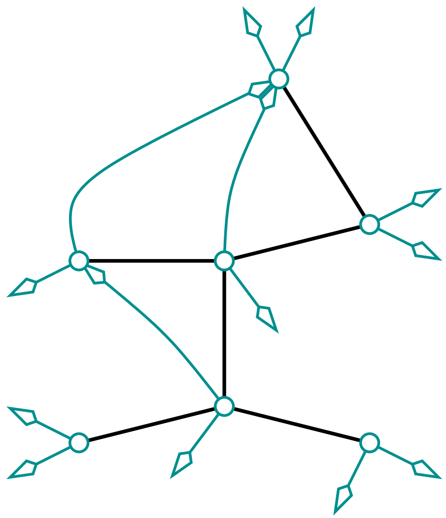
TRIANGULATIONS



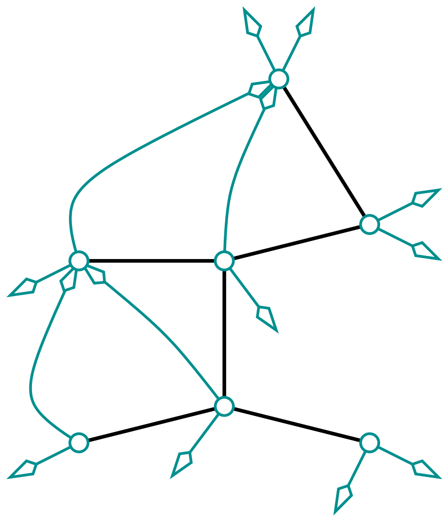
Triangulations



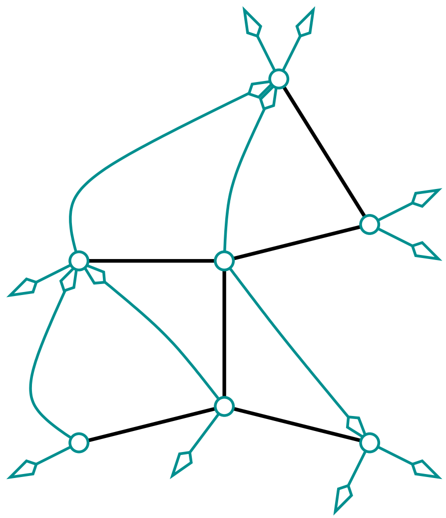
TRIANGULATIONS



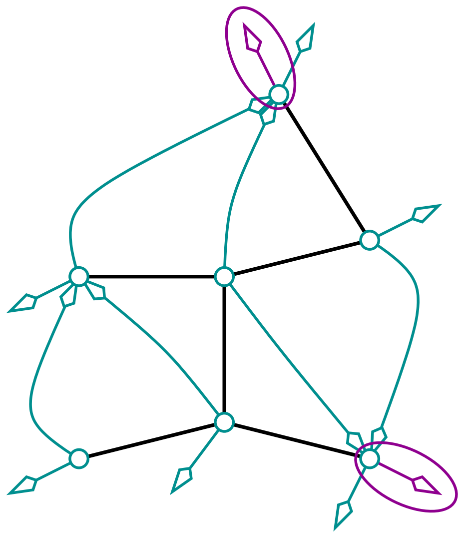
TRIANGULATIONS



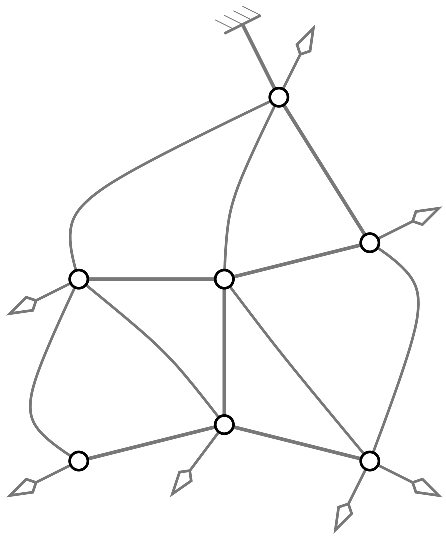
Triangulations



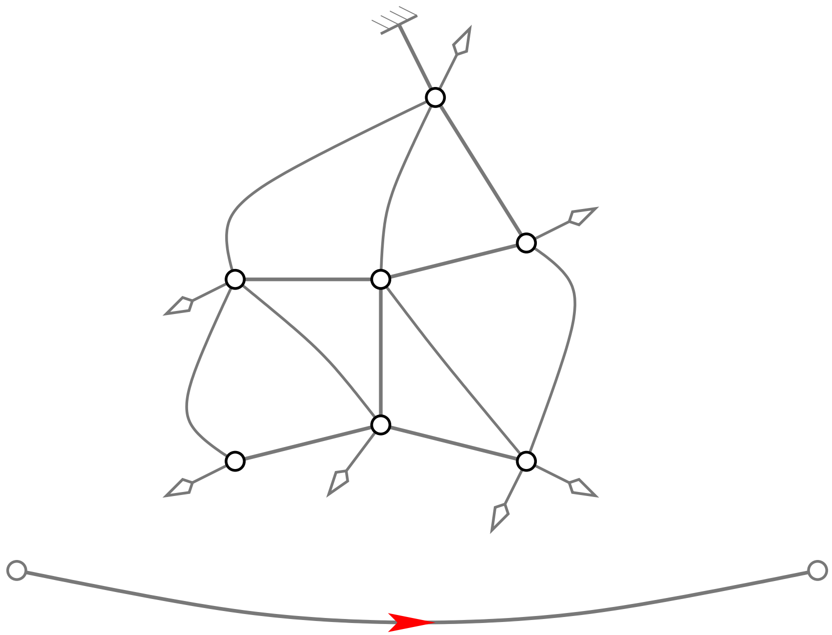
TRIANGULATIONS



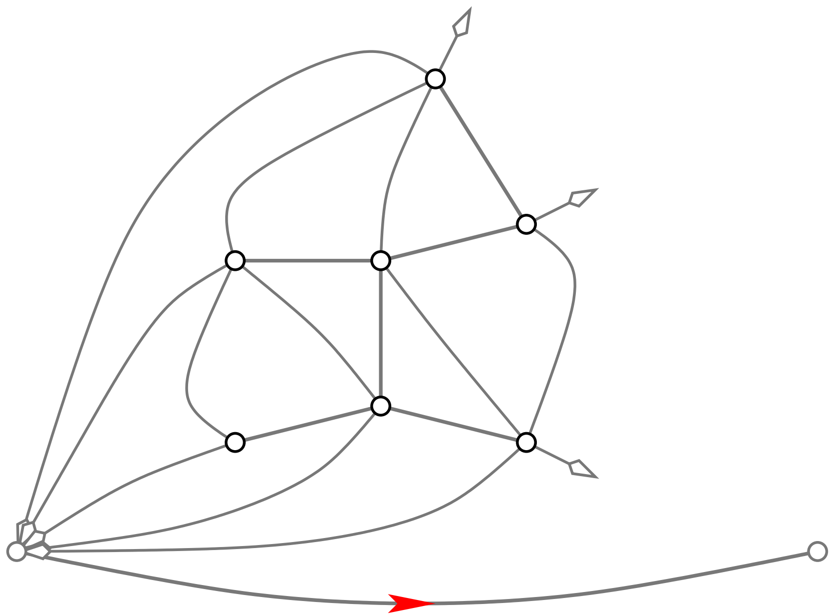
TRIANGULATIONS



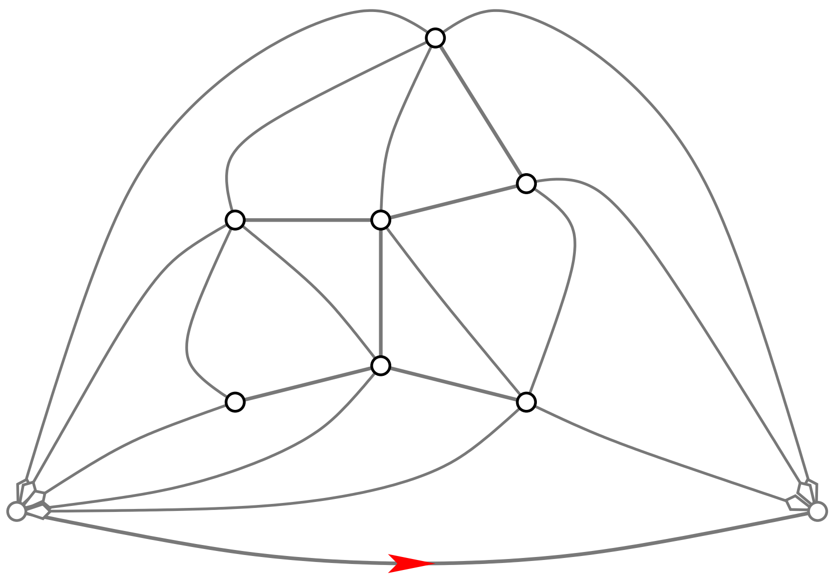
TRIANGULATIONS



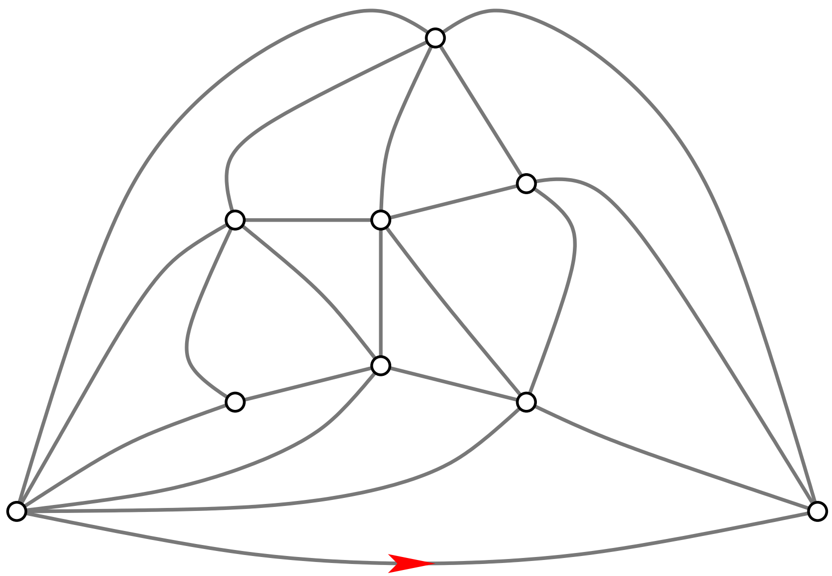
Triangulations



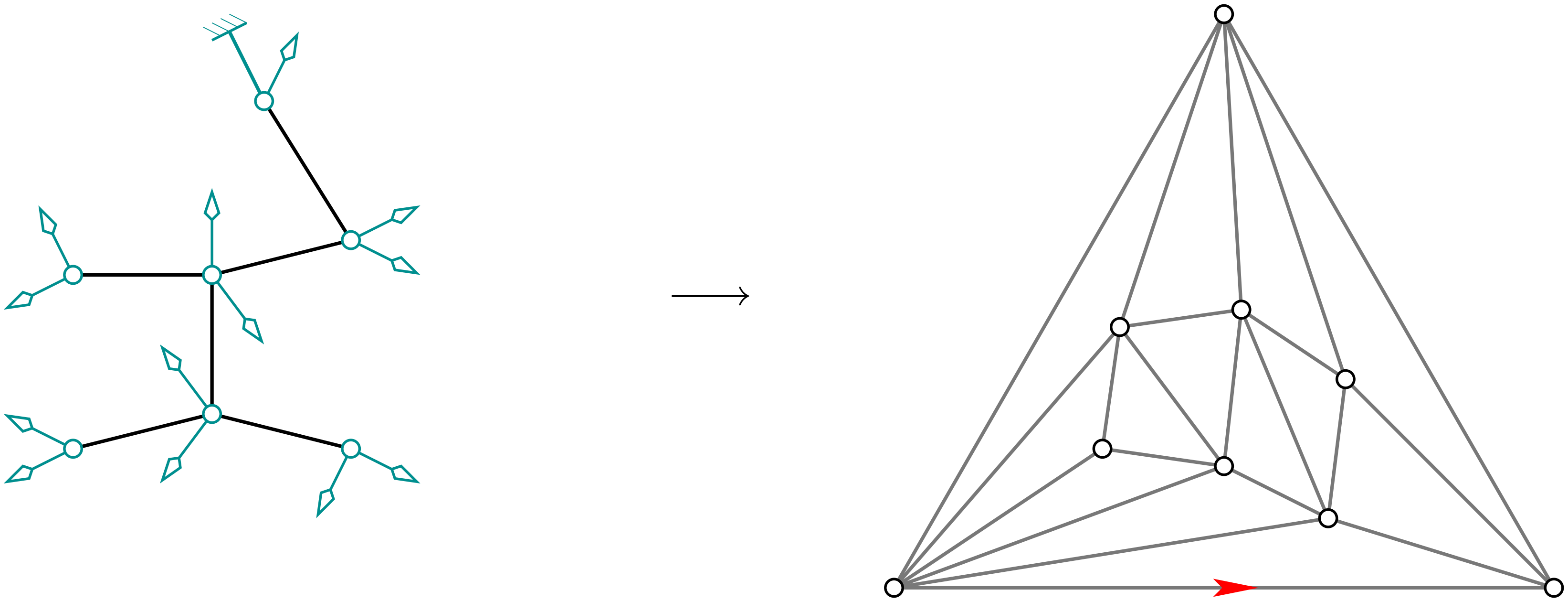
Triangulations



TRIANGULATIONS



Triangulations



Theorem (Poulalhon-Schaeffer 03). This is a bijection and its inverse is based on the minimal 3-orientations of a triangulation.

Conclusion of Part 5.

- Minimal α-orientations hide trees...
- It remains to give a common explanation to these various results : a theory of trees and minimal α-orientations.

A conclusion to bring home.

Nice counting formulas must have simple interpretations

Looking for these reveals hidden combinatorial structure