Counting planar Eulerian orientations

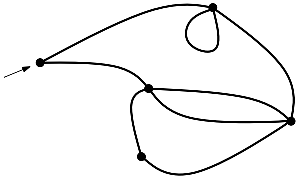
Nicolas Bonichon
Joint work with Claire Pennarun,
Mireille Bousquet-Mélou and Paul Dorbec

Egos meeting, Grenoble

Febuary 1^{st}, 2016

Definitions

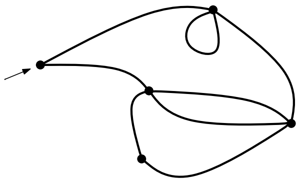
- planar rooted maps (in a corner)
- with loops and multiple edges



Definitions

- planar rooted maps (in a corner)
- with loops and multiple edges

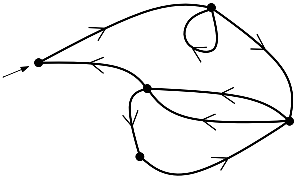
A map G is called *Eulerian* if every vertex of G has even degree.



Definitions

- planar rooted maps (in a corner)
- with loops and multiple edges

A map G is called *Eulerian* if every vertex of G has even degree.

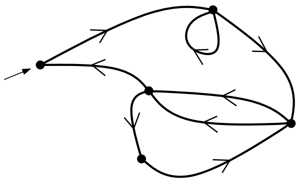


A directed planar map G is called a *planar Eulerian orientation (PEO)* if every vertex of G has in-degree and out-degree equal.

Definitions

- planar rooted maps (in a corner)
- with loops and multiple edges

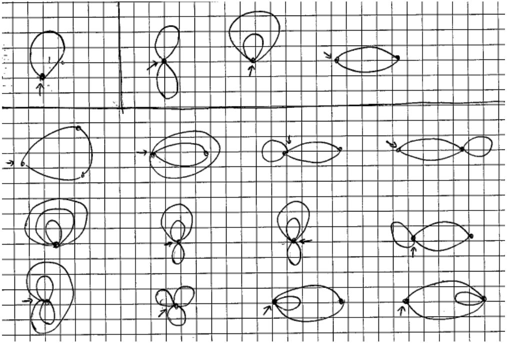
A map G is called *Eulerian* if every vertex of G has even degree.



A directed planar map G is called a *planar Eulerian orientation (PEO)* if every vertex of G has in-degree and out-degree equal.

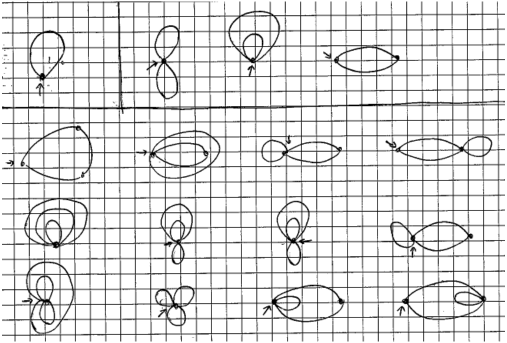
Counting Eulerian maps

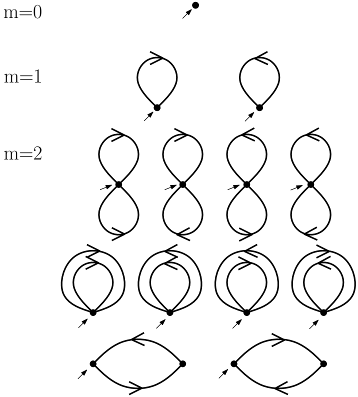
First numbers (with m edges): 1, 1, 3, 12, 56, 288, 1584, 9152... [OEIS A000257]



Formula [Tutte]: $E(m) = 3\frac{2^{m-1}.C_m}{m+2}$, where C_m is the m-th Catalan $= \frac{(2m)!}{m!(m+1)!}$.

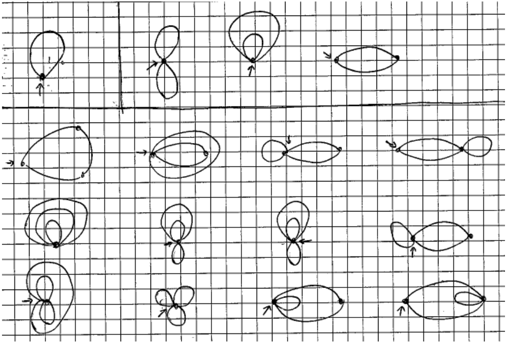
Planar Eulerian orientations

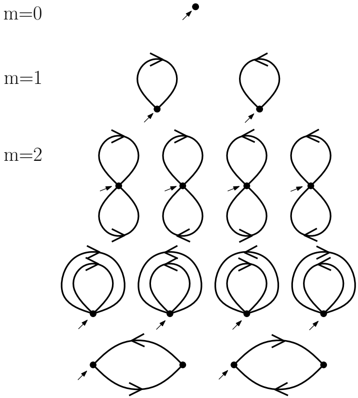




How many are there with m edges?

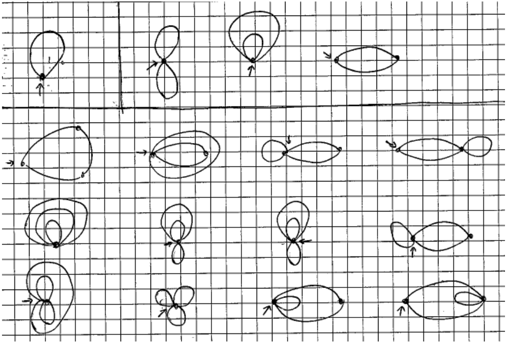
Planar Eulerian orientations

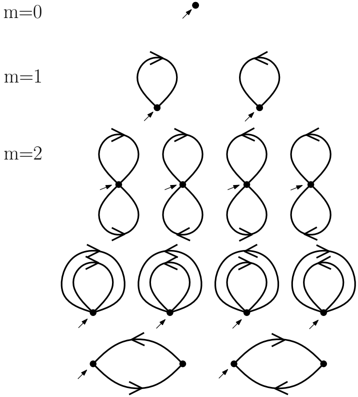




How many are there with m edges?
First idea: generate all orientations of each Eulerian map!

Planar Eulerian orientations





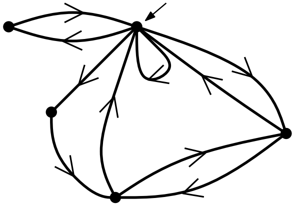
How many are there with *m* edges?
First idea: generate all orientations of each Eulerian map!

But counting the number of planar Eulerian orientations of a given map is #*P*-complete for undirected graphs [Mihail and Winckler 1996].

Generation of planar Eulerian orientations

Two possible actions to generate a bigger map:

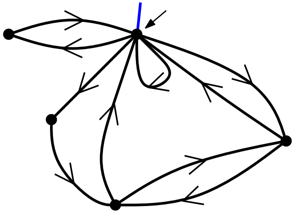
- add a loop at the root-vertex
- "split" the root-vertex in two + add a new edge
  (i-split: split giving i edges to the new vertex)



Generation of planar Eulerian orientations

Two possible actions to generate a bigger map:

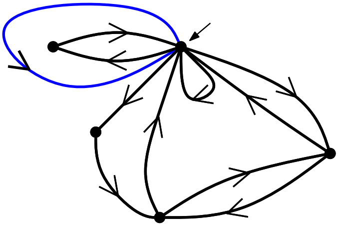
- add a loop at the root-vertex
- "split" the root-vertex in two + add a new edge
  (i-split: split giving i edges to the new vertex)



Generation of planar Eulerian orientations

Two possible actions to generate a bigger map:

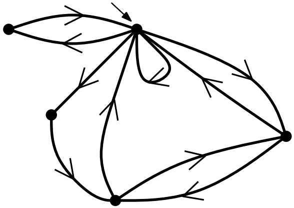
- add a loop at the root-vertex
- "split" the root-vertex in two + add a new edge
  (i-split: split giving i edges to the new vertex)



Generation of planar Eulerian orientations

Two possible actions to generate a bigger map:

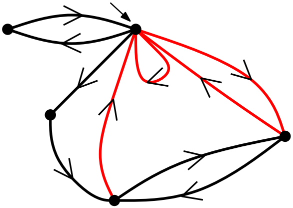
- add a loop at the root-vertex
- "split" the root-vertex in two + add a new edge
  (i-split: split giving i edges to the new vertex)



Generation of planar Eulerian orientations

Two possible actions to generate a bigger map:

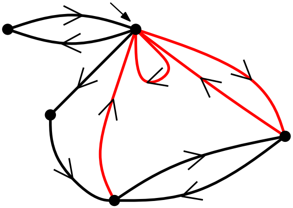
- add a loop at the root-vertex
- "split" the root-vertex in two + add a new edge
  (i-split: split giving i edges to the new vertex)



Generation of planar Eulerian orientations

Two possible actions to generate a bigger map:

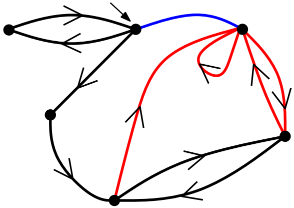
- add a loop at the root-vertex
- "split" the root-vertex in two + add a new edge
  (i-split: split giving i edges to the new vertex)
  ⚠ Legal split ⇔ correct orientation of the new edge



Generation of planar Eulerian orientations

Two possible actions to generate a bigger map:

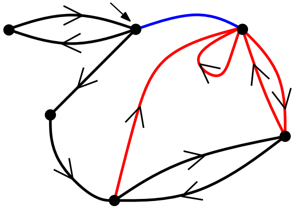
- add a loop at the root-vertex
- "split" the root-vertex in two + add a new edge
  (i-split: split giving i edges to the new vertex)
  ⚠ Legal split ⇔ correct orientation of the new edge



Generation of planar Eulerian orientations

Two possible actions to generate a bigger map:

- add a loop at the root-vertex
- "split" the root-vertex in two + add a new edge
  (i-split: split giving i edges to the new vertex)
  ⚠ Legal split ⇔ correct orientation of the new edge

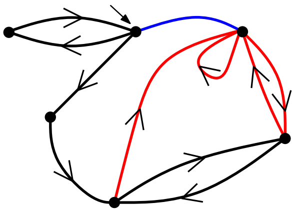


→ enough to get all planar Eulerian orientations.

Generation of planar Eulerian orientations

Two possible actions to generate a bigger map:

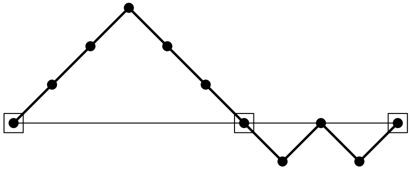
- add a loop at the root-vertex
- "split" the root-vertex in two + add a new edge
  (i-split: split giving i edges to the new vertex)
  ⚠ Legal split ⇔ correct orientation of the new edge



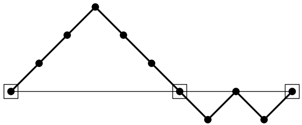
→ enough to get all planar Eulerian orientations.
Needed: appearance on the outer face + local orientation around the root-vertex

Grand-Dyck encoding

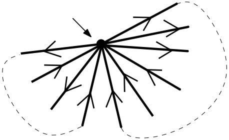
Decorated Grand-Dyck words: encode the orientation of the root-vertex and its appearance on the outer face.

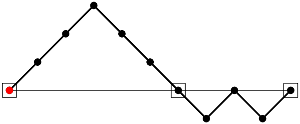


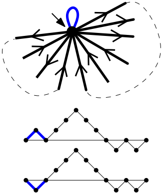
Grand-Dyck encoding



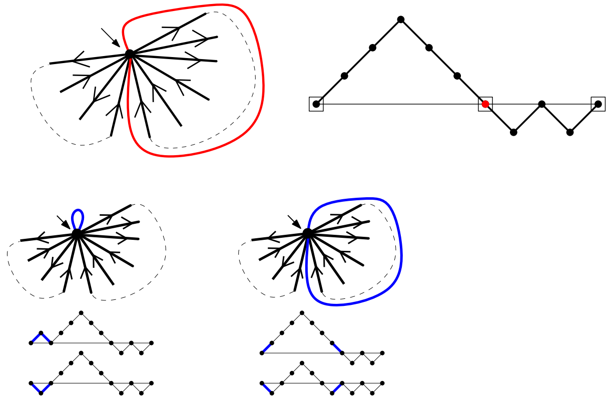
Grand-Dyck encoding



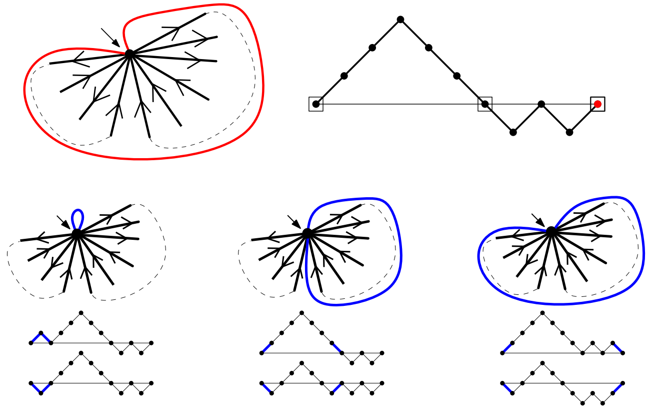




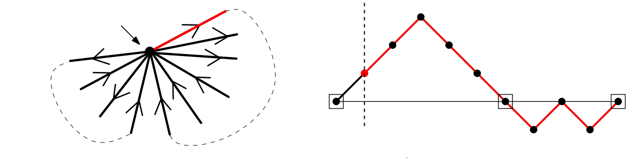
Grand-Dyck encoding

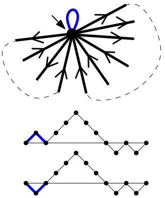


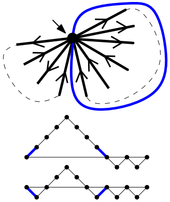
Grand-Dyck encoding

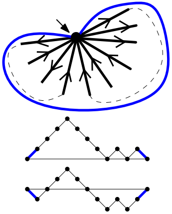


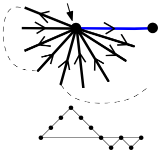
Grand-Dyck encoding



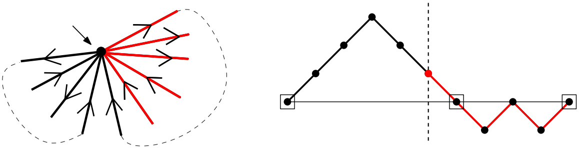


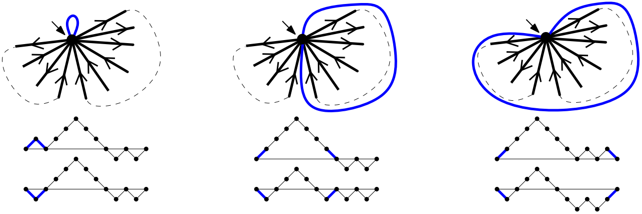


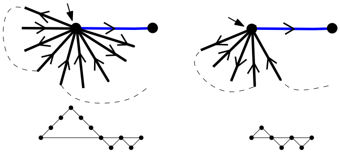




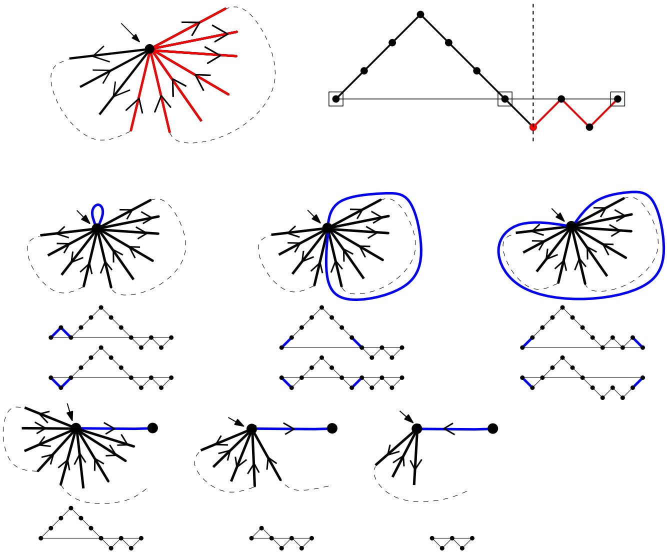
Grand-Dyck encoding



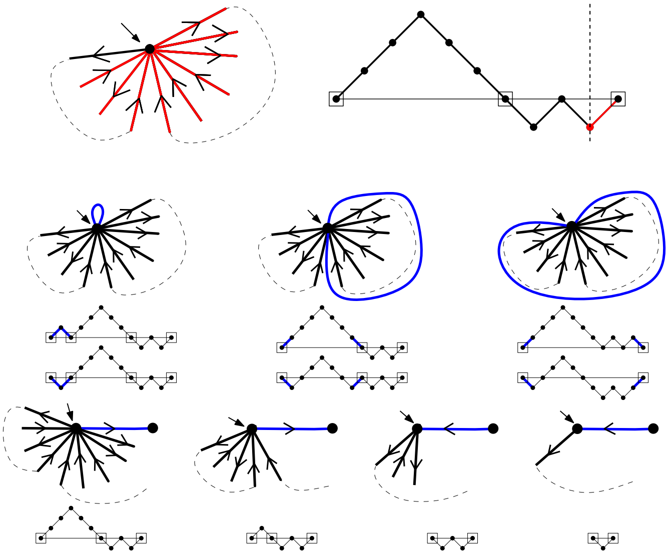




Grand-Dyck encoding



Grand-Dyck encoding



Counting with Grand-Dyck paths

m	Eul. maps	**PEO**	meanders	orient. Eul. maps
0	1	**1**	1	1
1	1	**2**	2	2
2	3	**10**	10	12
3	12	**66**	66	96
4	56	**504**	504	896

Counting with Grand-Dyck paths

m	Eul. maps	**PEO**	meanders	orient. Eul. maps
0	1	**1**	1	1
1	1	**2**	2	2
2	3	**10**	10	12
3	12	**66**	66	96
4	56	**504**	504	896
5	288	**4 216**	4 210	9 216
6	1 584	**37 548**	37 378	101 376
7	9 152	**350 090**	346 846	1 171 456
8	54 912	**3 380 520**	3 328 188	14 057 472
9	339 456	**33 558 024**	32 786 630	173 801 472
10	2 149 888	**340 670 720**	329 903 058	2 201 485 312
11	13 891 584	**3 522 993 656**	?	28 449 964 032
growth	8^m	?	?	16^m

→ No general formula... Let's try to formalize a decomposition!

Tutte-like decomposition

$$P_{\mathbf{w}} = \sum_{m>0} a_{\mathbf{w},m} t^m$$

with $a_{\mathbf{w},m}$ the number of PEO of size m with root of type $\mathbf{w}$.

$$P = 1 + \sum_{\mathbf{w} \in \{0,1\}^+} P_{\mathbf{w}}$$

with

$$P_{\mathbf{w}} = \begin{cases} \left(\sum_{a\mathbf{u}\bar{a}\mathbf{v}=\mathbf{w}} P_{\mathbf{u}} P_{\mathbf{v}} + \sum_{a\mathbf{v}=\mathbf{w}} P_{\mathbf{uv}} \right), & \text{if } \mathbf{w} \text{ is balanced,} \\ 0 & \text{otherwise.} \end{cases}$$

Tutte-like decomposition

$$P_{\mathbf{w}} = \sum_{m>0} a_{\mathbf{w},m} t^m$$

with $a_{\mathbf{w},m}$ the number of PEO of size m with root of type $\mathbf{w}$.

$$P = 1 + \sum_{\mathbf{w}\in\{0,1\}^+} P_{\mathbf{w}}$$

with

$$P_{\mathbf{w}} = \begin{cases} \left(\sum_{a\mathbf{u}\bar{a}\mathbf{v}=\mathbf{w}} P_{\mathbf{u}} P_{\mathbf{v}} + \sum_{a\mathbf{v}=\mathbf{w}} P_{\mathbf{uv}}\right), & \text{if } \mathbf{w} \text{ is balanced,} \\ 0 & \text{otherwise.} \end{cases}$$

Tutte-like decomposition

$$P_{\mathbf{w}} = \sum_{m>0} a_{\mathbf{w},m} t^m$$

with $a_{\mathbf{w},m}$ the number of PEO of size m with root of type $\mathbf{w}$.

$$P = 1 + \sum_{\mathbf{w}\in\{0,1\}^+} P_{\mathbf{w}}$$

with

$$P_{\mathbf{w}} = \begin{cases} \left(\sum_{a\mathbf{u}\bar{a}\mathbf{v}=\mathbf{w}} P_{\mathbf{u}} P_{\mathbf{v}} + \sum_{a\mathbf{v}=\mathbf{w}} P_{\mathbf{uv}}\right), & \text{if } \mathbf{w} \text{ is balanced,} \\ 0 & \text{otherwise.} \end{cases}$$

Tutte-like decomposition

$$P_{\mathbf{w}} = \sum_{m>0} a_{\mathbf{w},m} t^m$$

with $a_{\mathbf{w},m}$ the number of PEO of size m with root of type $\mathbf{w}$.

$$P = 1 + \sum_{\mathbf{w}\in\{0,1\}^+} P_{\mathbf{w}}$$

with

$$P_{\mathbf{w}} = \begin{cases} \left(\sum_{a\mathbf{u}\bar{a}\mathbf{v}=\mathbf{w}} P_{\mathbf{u}}P_{\mathbf{v}} + \sum_{a\mathbf{v}=\mathbf{w}} P_{\mathbf{uv}}\right), & \text{if } \mathbf{w} \text{ is balanced,} \\ 0 & \text{otherwise.} \end{cases}$$

Lower bounds and upper bounds

→ verify for $3 \leq i \leq \Delta - 3$ if i-split is legal
(1-split and $(\Delta - 1)$-split are always legal)

Lower bounds and upper bounds

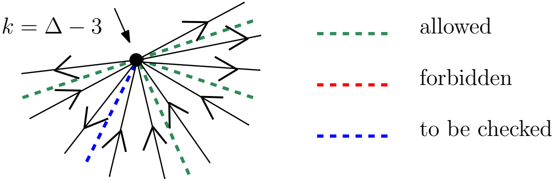
→ verify for $3 \leq i \leq \Delta - 3$ if i-split is legal
(1-split and $(\Delta - 1)$-split are always legal)

Simpler: consider it "done" for the first splits! → generate lower/upper bounds

Lower bounds and upper bounds

→ verify for $3 \leq i \leq \Delta - 3$ if i-split is legal
(1-split and $(\Delta - 1)$-split are always legal)

Simpler: consider it "done" for the first splits! → generate lower/upper bounds

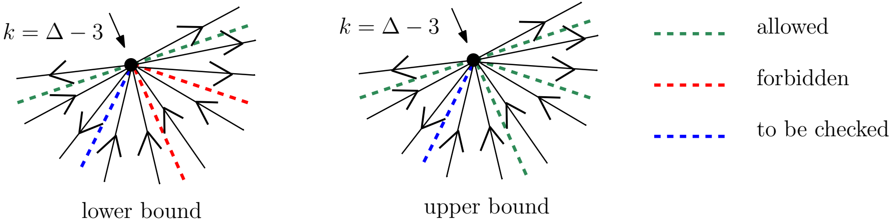


upper bound

Lower bounds and upper bounds

→ verify for $3 \le i \le \Delta - 3$ if i-split is legal
(1-split and $(\Delta - 1)$-split are always legal)

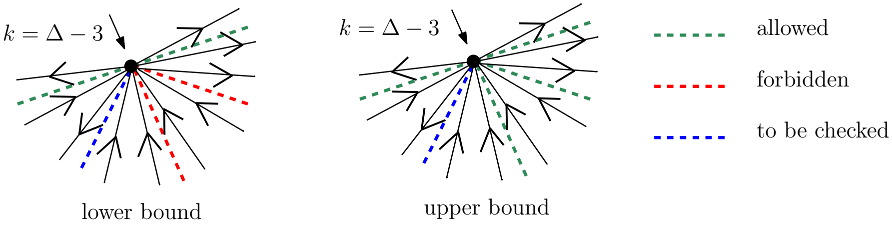
Simpler: consider it "done" for the first splits! → generate lower/upper bounds



Lower bounds and upper bounds

→ verify for $3 \leq i \leq \Delta - 3$ if i-split is legal
(1-split and $(\Delta - 1)$-split are always legal)

Simpler: consider it "done" for the first splits! → generate lower/upper bounds

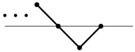


Last orientations around the root-vertex → legality of a split.

A lower bound: generating $PEO_{\Delta-3}$

F(ull): we know the whole word
L(ast): we know the last orientations

F_{1010}

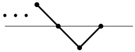


L_{001}

A lower bound: generating $PEO_{\Delta-3}$

F(ull): we know the whole word
L(ast): we know the last orientations

F_{1010}

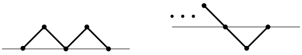


L_{001}

For $k = \Delta - 3$, four classes of orientations:

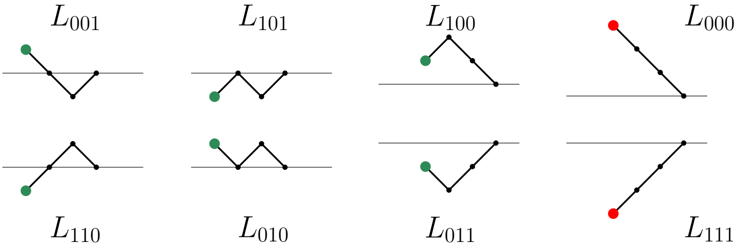
A lower bound: generating $PEO_{\Delta-3}$

F(ull): we know the whole word
L(ast): we know the last orientations



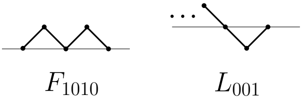
F_{1010} L_{001}

For $k = \Delta - 3$, four classes of orientations:

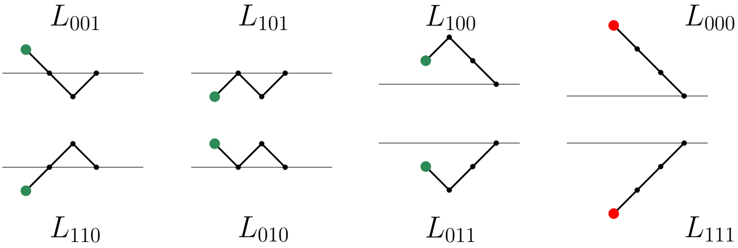


A lower bound: generating $PEO_{\Delta-3}$

F(ull): we know the whole word
L(ast): we know the last orientations

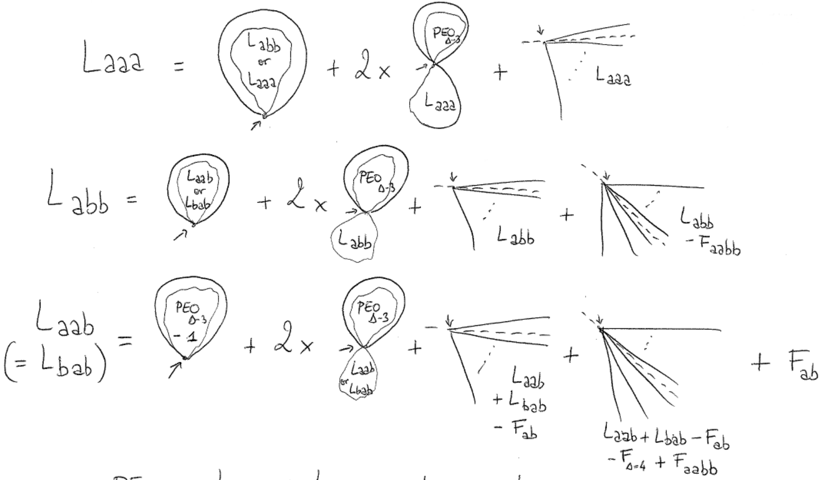


For $k = \Delta - 3$, four classes of orientations:



How are these classes generated?

A LOWER BOUND: GENERATING $PEO_{\Delta-3}$



$$PEO_{\Delta-3} = L_{aaa} + L_{abb} + L_{aab} + L_{bab} + 1$$

A lower bound: generating $PEO_{\Delta-3}$

$$
\begin{aligned}
L_{aab} = L_{bab} &= F_{ab} + 2tPEO_{\Delta-3}(L_{aab} + L_{bab}) + t((L_{aab} + L_{bab}) - F_{ab}) \\
&\quad + t((L_{aab} + L_{bab}) - F_{ab} - F_{\Delta=4} + F_{aabb}) + t(PEO_{\Delta-3} - 1) \\
L_{aaa} &= t(L_{abb} + L_{aaa}) + 2tPEO_{\Delta-3}L_{aaa} + tL_{aaa} \\
L_{abb} &= t(L_{aab} + L_{bab}) + 2tPEO_{\Delta-3}L_{abb} + tL_{abb} + t(L_{abb} - F_{aabb}) \\
PEO_{\Delta-3} &= L_{aaa} + L_{abb} + L_{aab} + L_{bab} + 1
\end{aligned}
$$

A lower bound: generating $PEO_{\Delta-3}$

$$
\begin{aligned}
L_{aab} = L_{bab} &= F_{ab} + 2tPEO_{\Delta-3}(L_{aab} + L_{bab}) + t((L_{aab} + L_{bab}) - F_{ab}) \\
&\quad + t((L_{aab} + L_{bab}) - F_{ab} - F_{\Delta=4} + F_{aabb}) + t(PEO_{\Delta-3} - 1) \\
L_{aaa} &= t(L_{abb} + L_{aaa}) + 2tPEO_{\Delta-3}L_{aaa} + tL_{aaa} \\
L_{abb} &= t(L_{aab} + L_{bab}) + 2tPEO_{\Delta-3}L_{abb} + tL_{abb} + t(L_{abb} - F_{aabb}) \\
PEO_{\Delta-3} &= L_{aaa} + L_{abb} + L_{aab} + L_{bab} + 1
\end{aligned}
$$

→ Automating the process to produce a system of equations for all classes.

Results

Computation with Maple packages `combstruct` and `NewtonGF`.

	growth	1	2	3	4	5	6
Eulerian maps	8^m	1	3	12	56	288	1 584
inf : $k = \Delta - 1$	9.68^m	2	10	66	466	3 458	26 650
inf : $k = \Delta - 3$	10.16^m	2	10	66	504	4 008	32 834
inf : $k = \Delta - 5$	10.51^m	2	10	66	504	4 216	36 316
inf : $k = \Delta - 7$	$\geq 10.69^m$	2	10	66	504	4 216	37 548
Eulerian orientations	**?**	**2**	**10**	**66**	**504**	**4 216**	**37 548**
sup : $k = \Delta - 3$	12.95^m	2	10	66	504	4 234	37 998
sup : $k = \Delta - 1$	13.06^m	2	10	66	506	4 266	38 418
Oriented Eulerian maps	16^m	2	12	96	896	9 216	101 376

Open questions

- Is the generating function of planar Eulerian orientations algebraic?
- Can we find an other (simpler?) decomposition for PEO?

Thank you for your attention !