ENUMERATION OF PLANAR CONFIGURATIONS
IN COMPUTATIONAL GEOMETRY

Philippe Flajolet
(Version of January 15, 1997)

Consider *n* points on a circle that define a convex polygon. The enumeration of geometric configurations that can be superimposed on these points has a dignified history. In 1753, Euler solved the problem of counting the number of possible triangulations of a convex *n*-gon while inventing the concept of generating function for this particular combinatorial enumeration problem. Since then, many configurations have been enumerated; see Comtet's *Advanced Combinatorics,* p. 74 for a discussion of works of Prouhet 1886, Jordan 1920, Guy 1967.

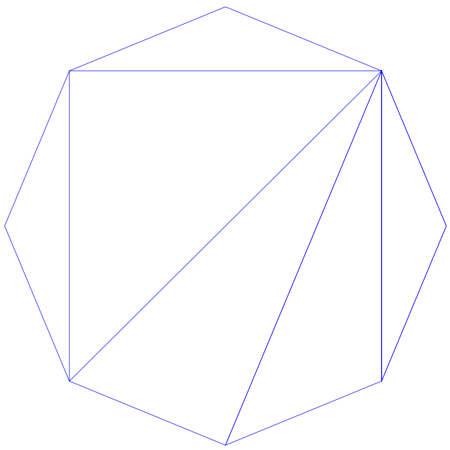
Till recently, any new enumerative result was basically a research paper, and some of the applications discussed below have required several pages of recurrence manipulations. In this report, we take inspiration from a recent memo of Noy (September 1996) and show that many such results can be derived automatically using the Maple system and the packages Combstruct and Gfun. This worsheet is meant as a simple introduction to Combstruct specifications as well as to basic experimental computations that may accompany them.

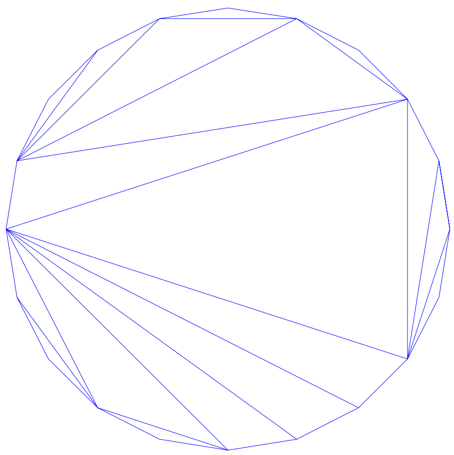
Euler's counting of triangulations

Problem

Find the number of ways of cutting up a convex $(n+2)$-gon into n triangles by means of $(n-1)$ non-intersecting diagonals.

Here are examples of triangulations of the octogon and of the icosagon.





Specification

A triangulation decomposes into three components:

- the "root" triangle uniquely defined uniquely by the fact that it contains the edge with smallest numbers (0, 1 initially);

- the left and right subtriangulations defined by their position with respect to the root triangle.

First, we load the combstruct package:

```
> with(combstruct);
```

[*allstructs*, *count*, *draw*, *finished*, *gfeqns*, *gfseries*, *gfsolve*, *iterstructs*, *nextstruct*, *prog_gfeqns*, *prog_gfseries*, *prog_gfsolve*]

In writing a specification, we take the atomic symbol Z to denote a traingle, T being an arbitrary triangulation. One must be careful to isolate one-sided triangulations, where either the left or the right subtriangulations may be empty. Thus, the grammar is

```
> triang:=[T,{T=Union(Z,Prod(Epsilon,Z,T),Prod(T,Z,Epsilon),
  Prod(T,Z,T))},unlabelled]:
```

This specification is clearly reminiscent of binary search trees.

Counting.

Now, we can count the number of triangulations of size 100:

```
> count(triang,size=100);
```

Here the system takes a couple of seconds to set up (once and for all!) complete counting tables till size equal to 100.

896519947090131496687170070074100632420837521538745909320

In particular, the first few values are

```
> seq(count(triang,size=i),i=0..20);
```

0, 1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796, 58786, 208012, 742900, 2674440, 9694845, 35357670, 129644790, 477638700, 1767263190, 6564120420

These numbers are known as the Catalan numbers in the honor of the French and Belgian mathematician who worked out their main properties in the 1850's.

Random generation.

We can draw a triangulation at random amongst those of any given size. Note that uniformity is granted a priori by the algorithms contained in the combstruct package. Naturally, the output is in the format dictated by the specification. Here is a triangulation of size 10, that is to say, comprised of 10 triangles, and corresponding to a dodecagon.

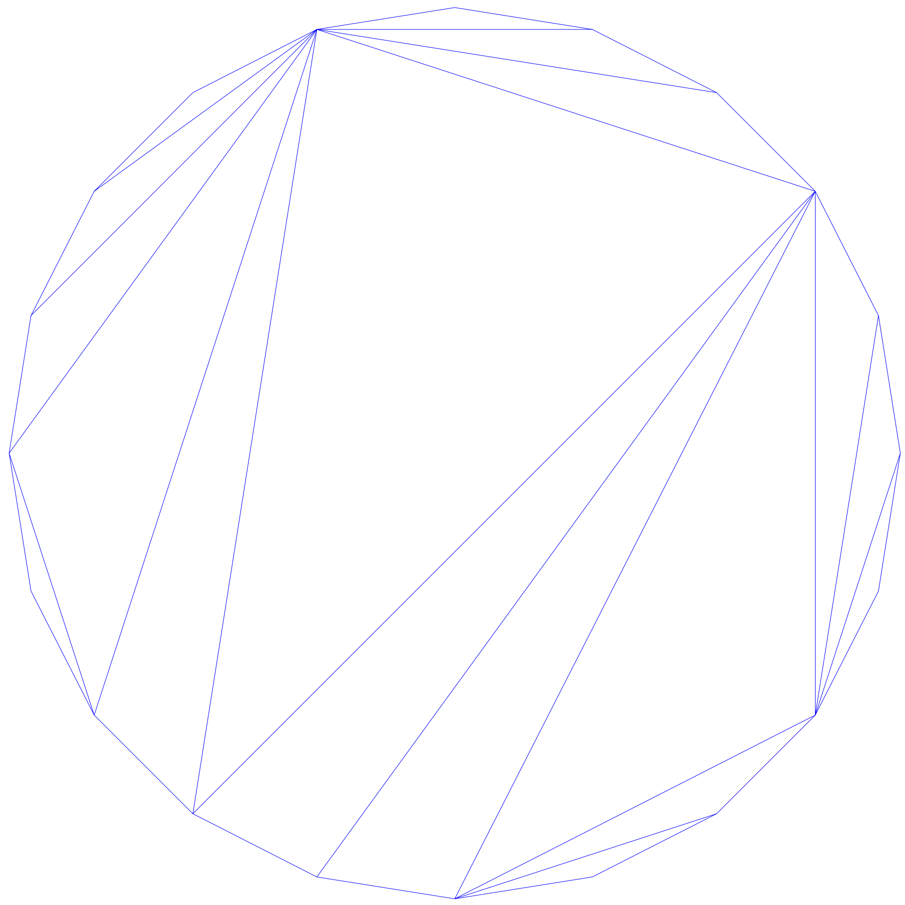
```
> draw(triang,size=10);
```

Prod(Prod(Z, Z, Prod(Prod(E, Z, Prod(E, Z, Z)), Z, E)), Z, Prod(Prod(E, Z, Z), Z, E))

Since the objects generated are standard Maple objects, we can manipulate them and in particular construct plots to visualise them. Here is a set of procedures to construct the list of edges of a triangulation.

```
> size:=proc(t) convert(map(size,t),`+`) end:
  size(Epsilon):=0: size(Z):=1:
> set_edges:=proc(tree,org,edge_set) local se1,se2;
      if tree=Epsilon then
          org,org,edge_set union {[org,org+1]}
      elif tree=Z then
          org,org+1,edge_set union
  {[org,org+1],[org+1,org+2],[org+2,org]}
```

```
        else
            se1:=set_edges(op(1,tree),org,edge_set);
            se2:=set_edges(op(3,tree),se1[2]+1,se1[3]);
            se1[1],se2[2],se2[3] union {[se1[1],se2[2]+1]}
        fi
    end:
> plott:=proc(n)
    plot([op(map2(map,[cos,sin],expand(map(`*`,set_edges(draw(
    triang,size=n-2),0,{})[3],2*Pi/n))))],color=blue,axes=NONE
    ) end:
> plott(20);
```



Exhaustive generation.

With the December 1996 release of Combstruct, we may also list exhautively all configurations of a given size. This proves useful for exhaustive testing of small cases, though there are physical limitations owing to the exponential growth of the number of cases.

```
> tr4:=allstructs(triang,size=4);
```

$tr4 := [\mathrm{Prod}(E, Z, \mathrm{Prod}(E, Z, \mathrm{Prod}(E, Z, Z))), \mathrm{Prod}(\mathrm{Prod}(\mathrm{Prod}(Z, Z, E), Z, E), Z, E),$
$\mathrm{Prod}(\mathrm{Prod}(E, Z, \mathrm{Prod}(Z, Z, E)), Z, E), \mathrm{Prod}(\mathrm{Prod}(E, Z, Z), Z, Z),$
$\mathrm{Prod}(Z, Z, \mathrm{Prod}(Z, Z, E)), \mathrm{Prod}(\mathrm{Prod}(Z, Z, E), Z, Z), \mathrm{Prod}(E, Z, \mathrm{Prod}(Z, Z, Z)),$
$\mathrm{Prod}(E, Z, \mathrm{Prod}(\mathrm{Prod}(E, Z, Z), Z, E)), \mathrm{Prod}(\mathrm{Prod}(E, Z, \mathrm{Prod}(E, Z, Z)), Z, E),$
$\mathrm{Prod}(Z, Z, \mathrm{Prod}(E, Z, Z)), \mathrm{Prod}(\mathrm{Prod}(Z, Z, Z), Z, E),$
$\mathrm{Prod}(E, Z, \mathrm{Prod}(\mathrm{Prod}(Z, Z, E), Z, E)), \mathrm{Prod}(E, Z, \mathrm{Prod}(E, Z, \mathrm{Prod}(Z, Z, E))),$
$\mathrm{Prod}(\mathrm{Prod}(\mathrm{Prod}(E, Z, Z), Z, E), Z, E)]$

We found 14 objects, as we should.

Generating function equations.

Again, this uses the December 1996 release of Combstruct. We may form the generating function equations associated to a given specification, as follows:

```
> gfeqns(op(2,triang),unlabelled,z);
```

$$[Z(z) = z, T(z) = Z(z) + 2\,Z(z)\,T(z) + T(z)^2\,Z(z)]$$

```
> gfsys:=gfsolve(op(2,triang),unlabelled,z);
```

$$gfsys := \{T(z) = \frac{1}{2}\,\frac{1 - 2\,z - \sqrt{1 - 4\,z}}{z}, Z(z) = z\}$$

Here, the solution could be obtained explicitly. Correctness can be checked by a Taylor expansion, comparing with what combstruct[count] gives us.

```
> op([1,2],"); series(",z=0,12);
```

$$\frac{1}{2}\,\frac{1 - 2\,z - \sqrt{1 - 4\,z}}{z}$$

$$z + 2\,z^2 + 5\,z^3 + 14\,z^4 + 42\,z^5 + 132\,z^6 + 429\,z^7 + 1430\,z^8 + 4862\,z^9 + 16796\,z^{10} + O(z^{11})$$

```
> seq(count(triang,size=n),n=0..10);
```

$$0, 1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796$$

Alternatively, an arbitrary system of generating function equations can be solved by means of combstruct[gfseries], even when no closed form is available.

```
> gfseries(op(2..3,triang),z,[[z]],6);
```

table([
$Z(z) = z$
$T(z) = z + 2\,z^2 + 5\,z^3 + 14\,z^4 + 42\,z^5 + O(z^6)$
])

Noy's counting of non-crossing trees

Problem.

Find the number of trees (called non-crossing trees) that can be built on the n vertices on the n-gon, assuming that no edges of the tree intersect.

Specification.

View node number 0 as the root, the vertices being number from 0 to $n-1$. An arbitrary number of edges connect the root to other nodes. At each such node m connected to the root, there is a "butterfly", defined by two noncrossing trees attached to m (one non-crossing tree to the left of the edges $\langle 0, m \rangle$, the other to the right). We let T denote non-crossing (NC) trees, F denote forests defined by pruning the root of a T-tree, and B denote a butterfly. The specification results.

```
> nctree:=[T,{T=Prod(Z,F),F=Sequence(B),B=Prod(F,Z,F)},unlabelled]:
> seq(count(nctree,size=i),i=0..20);
```

0, 1, 1, 3, 12, 55, 273, 1428, 7752, 43263, 246675, 1430715, 8414640, 50067108, 300830572, 1822766520, 11124755664, 68328754959, 422030545335, 2619631042665, 16332922290300

Random generation, exhaustive generation.

We draw objects at random with combstruct[draw]. Here is a random NC-tree.

```
> draw(nctree,size=8);
```

Prod(Z, Sequence(Prod(E, Z, Sequence(Prod(Sequence(Prod(Sequence(Prod(Sequence(Prod(E, Z, E)), Z, Sequence(Prod(E, Z, E)))), Z, E) , Z, E))), Prod(E, Z, E)))

Guessing counts with Maple.

At this stage, we ask ourselves whether a closed form expression may exist. First, let us examine the way we might proceed in a simple case, using Maple. Consider for instance the number of NC-trees of size 30 and try to factor it.

```
> ifactor(count(nctree,size=30));
```

$$(2)^3 (3)^2 (5) (17) (31) (37) (41) (43) (61) (67) (71) (73) (79) (83)$$

We observe that the count is a highly composite number that involves all the prime numbers between 31 and 83. This is a strong indication that a closed form involving factorials or binomial coefficients should exist. The pattern can then be guessed by taking successive quotients.

```
> seq([n,ifactor(count(nctree,size=n+1)/count(nctree,size=n))],n=1..15);
```

$$[1, 1], [2, (3)], [3, (2)^2], \left[4, \frac{(5)(11)}{(2)^2(3)}\right], \left[5, \frac{(3)(7)(13)}{(5)(11)}\right], \left[6, \frac{(2)^2(17)}{(13)}\right], \left[7, \frac{(2)(19)}{(7)}\right], \left[8, \frac{(3)(11)(23)}{(2)^3(17)}\right], \left[9, \frac{(5)^2(13)}{(3)(19)}\right], \left[10, \frac{(29)}{(5)}\right], \left[11, \frac{(2)^4(3)(31)}{(11)(23)}\right], \left[12, \frac{(7)(17)}{(2)^2(5)}\right], \left[13, \frac{(19)(37)}{(3)^2(13)}\right], \left[14, \frac{(2)(3)(5)(41)}{(7)(29)}\right], \left[15, \frac{(2)(11)(43)}{(5)(31)}\right]$$

The ratios clearly involve n and $2\,n+1$ in the denominator, at least whenever these are primes. We thus try:

```
> seq([n,ifactor(n*(2*n+1)*count(nctree,size=n+1)/count(nctr
  ee,size=n))],n=1..15);
```

$[1,(3)],[2,(2)(3)(5)],[3,(2)^2(3)(7)],[4,(3)(5)(11)],[5,(3)(7)(13)],$
$[6,(2)^3(3)(17)],[7,(2)(3)(5)(19)],[8,(3)(11)(23)],[9,(3)(5)^2(13)],$
$[10,(2)(3)(7)(29)],[11,(2)^4(3)(31)],[12,(3)(5)(7)(17)],$
$[13,(3)(19)(37)],[14,(2)^2(3)(5)(41)],[15,(2)(3)(11)(43)]$

The numerators now seem to involve $3\,n-1$ and $3\,m-2$. Thus we try next

```
> seq([n,ifactor(n*(2*n+1)/(3*n-1)/(3*n-2)*count(nctree,size
  =n+1)/count(nctree,size=n))],n=1..15);
```

$\left[1,\frac{(3)}{(2)}\right],\left[2,\frac{(3)}{(2)}\right],\left[3,\frac{(3)}{(2)}\right],\left[4,\frac{(3)}{(2)}\right],\left[5,\frac{(3)}{(2)}\right],\left[6,\frac{(3)}{(2)}\right],\left[7,\frac{(3)}{(2)}\right],\left[8,\frac{(3)}{(2)}\right],$
$\left[9,\frac{(3)}{(2)}\right],\left[10,\frac{(3)}{(2)}\right],\left[11,\frac{(3)}{(2)}\right],\left[12,\frac{(3)}{(2)}\right],\left[13,\frac{(3)}{(2)}\right],\left[14,\frac{(3)}{(2)}\right],\left[15,\frac{(3)}{(2)}\right]$

We have thus found empirically evidence for the conjecture that the counts satisfy

```
> u(n+1)=3/2*n*(2*n+1)/(3*n-1)/(3*n-2)*u(n);
```

$$\mathrm{u}(n+1)=\frac{3}{2}\,\frac{n\,(2\,n+1)\,\mathrm{u}(n)}{(3\,n-1)\,(3\,n-2)}$$

This can be solved by means of the Maple command rsolve, suggesting a closed binomial form for the coefficients.

```
> rsolve({",u(1)=1},u);
```

$$8\,\frac{\sqrt{3}\,\pi\,\Gamma(2\,n)\,12^{(-n)}}{\Gamma\left(n-\frac{1}{3}\right)\Gamma\left(n-\frac{2}{3}\right)}$$

Naturally, we succeeded here only because the counts admit a direct product form.

Guessing counts with Gfun.

Our guessing task is greatly simplified if we appeal to the Gfun package that determines automatically recurrences for sequences and differential equations for the corresponding generating functions that match a given set of initial data.

```
> with(gfun);
```

[*Laplace*, *algebraicsubs*, *algeqtodiffeq*, *algeqtoseries*, *algfuntoalgeq*, *borel*, *cauchyproduct*, *diffeq*diffeq*, *diffeq+diffeq*, *diffeqtorec*, *guesseqn*, *guessgf*, *hadamardproduct*, *holexprtodiffeq*, *invborel*, *listtoalgeq*, *listtodiffeq*, *listtohypergeom*, *listtolist*, *listtoratpoly*, *listtorec*, *listtoseries*, *listtoseries/Laplace*, *listtoseries/egf*, *listtoseries/lgdegf*, *listtoseries/lgdogf*, *listtoseries/ogf*, *listtoseries/revegf*, *listtoseries/revogf*, *maxdegcoeff*, *maxdegeqn*, *maxordereqn*, *mindegcoeff*, *mindegeqn*, *minordereqn*, *optionsgf*, *poltodiffeq*, *poltorec*, *ratpolytocoeff*, *rec*rec*, *rec+rec*, *rectodiffeq*, *rectoproc*, *seriestoalgeq*, *seriestodiffeq*, *seriestohypergeom*, *seriestolist*, *seriestoratpoly*, *seriestorec*, *seriestoseries*]

First, we build a list of small values:

```
> l:=[seq(count(nctree,size=n),n=0..20)];
```

l := [0, 1, 1, 3, 12, 55, 273, 1428, 7752, 43263, 246675, 1430715, 8414640, 50067108, 300830572, 1822766520, 11124755664, 68328754959, 422030545335, 2619631042665, 16332922290300]

The procedure gfun[listtorec] finds automatically a plausible recurrence with polynomial coefficients, provided such a form exists; similarly, the procedure gfun[listtodiffeq] finds automatically a plausible differential equation with polynomial coefficients, provided such a form exists.

```
> listtorec(l,u(n));
```

$$[\{\mathrm{u}(1)=1, (-6+27\,n-27\,n^2)\,\mathrm{u}(n)+(2\,n+4\,n^2)\,\mathrm{u}(n+1), \mathrm{u}(0)=0\}, ogf]$$

```
> listtodiffeq(l,U(z));
```

$$\left[\left\{\mathrm{U}(0)=0, (-2+6\,z)\,\mathrm{U}(z)+2\,z\left(\frac{\partial}{\partial z}\mathrm{U}(z)\right)+(27\,z^3-4\,z^2)\left(\frac{\partial^2}{\partial z^2}\mathrm{U}(z)\right), \mathrm{D}(U)(0)=1\right\}, ogf\right]$$

This candidate equation can be passed for possible solution to Maple's dsolve.

```
> dsolve(op(1,"),U(z));
```

$$\mathrm{U}(z)=z\,\mathrm{hypergeom}\left(\left[\frac{2}{3},\frac{1}{3}\right],\left[\frac{3}{2}\right],\frac{27}{4}\,z\right)$$

Proving solutions with Combstruct.

With the version of December 1996, we can find automatically a system of equations and in this particular case, an explicit solution.

```
> gfeqns(op(2..3,nctree),z);
```

$$\left[\mathrm{Z}(z)=z, \mathrm{F}(z)=\frac{1}{1-\mathrm{B}(z)}, \mathrm{B}(z)=\mathrm{F}(z)^2\,\mathrm{Z}(z), \mathrm{T}(z)=\mathrm{Z}(z)\,\mathrm{F}(z)\right]$$

```
> nctree_sys:=gfsolve(op(2..3,nctree),z);
```

$$nctree\_sys := \{\mathrm{Z}(z)=z, \mathrm{B}(z)=\mathrm{RootOf}(-\_Z+\_Z^3\,z+1)^2\,z, \mathrm{T}(z)=z\,\mathrm{RootOf}(-\_Z+\_Z^3\,z+1), \mathrm{F}(z)=\mathrm{RootOf}(-\_Z+\_Z^3\,z+1)\}$$

This tells us in particular that the generating function of non-crossing trees is

```
> nctree_gf:=subs(nctree_sys,B(z));
```

$$nctree\_gf := \mathrm{RootOf}(-\_Z+\_Z^3\,z+1)^2\,z$$

```
> series(nctree_gf,z=0,12);
```

$$z+2\,z^2+7\,z^3+30\,z^4+143\,z^5+728\,z^6+3876\,z^7+21318\,z^8+120175\,z^9+690690\,z^{10}+4032015\,z^{11}+\mathrm{O}(z^{12})$$

Conclusion.

In particular, we have obtained automatically one of the theorems of Noy (1994):

Theorem. *The generating function of non crossing trees satisfies the equation*

$$T(z) = 1 + z\,T(z)^3.$$

The number of trees with n nodes is

$$\frac{\mathrm{binomial}(3\,n+1,\,n)}{3\,n+1}.$$

The result of combstruct[gfsolve] gives a valid proof of the generating function equation. The result of gfun[guessgf] followed by gfun[listtodiffeq] is a priori only heuristic. However, once a candidate recurrence or differential equation has been found, it may then rigorously be checked from withing gfun itself.

First, we compute automatically a differential equation from the generating function.

```
> algfuntoalgeq(nctree_gf,Y(z));
```

$$-z + Y - 2\,Y^2 + Y^3$$

```
> algeqtodiffeq(",Y(z));
```

$$-2 + 3\,Y(z) + (2 - 27\,z)\left(\frac{\partial}{\partial z}Y(z)\right) + (-27\,z^2 + 4\,z)\left(\frac{\partial^2}{\partial z^2}Y(z)\right)$$

This result has proven value and it coincides with the differential equation that was previously "guessed". Thus, the theorem has been established "automatically"...

Comtet's counting of slicings.

Problem.

Given n points on a circle, count the number of slicings, that is to say, graphs consisting of $\mathrm{floor}\left(\frac{n}{2}\right)$ *edges that do not intersect inside the circle.*

Specification, random generation, and counting.

Consider first the case of an even number of vertices. A slicing decomposes into a central edge (initially, the edge that contains vertex 0) and into a left and right slicing that are each of a similar nature. This is readily specified as follows.

```
> sl0:={Edge=Prod(Z,Z),S=Union(Epsilon,Prod(S,Edge,S))}:
  slicing:=[S,sl0,unlabelled]:
> seq(count(slicing,size=n),n=0..20);
```

1, 0, 1, 0, 2, 0, 5, 0, 14, 0, 42, 0, 132, 0, 429, 0, 1430, 0, 4862, 0, 16796

Our old friends, the Catalan numbers strike again.
An interest of the method is to allow for the counting of odd slicings, where one has a free unattached point. The following specification just says that the "free point" lies somewhere either left or right or at the root of the slicing.

```
> sl1:=sl0 union {Free=Z,
  S1=Union(Prod(S1,Edge,S),Prod(S,Edge,S1),Prod(S,Free,S))}:
  slicing1:=[S1,sl1,unlabelled]:
> seq(count(slicing1,size=n),n=0..20);
```

0, 1, 0, 4, 0, 15, 0, 56, 0, 210, 0, 792, 0, 3003, 0, 11440, 0, 43758, 0, 167960, 0

```
> draw(slicing1,size=20);
Error, (in combstruct/drawgrammar) there is no structure of this size
```

There, the system has recognized the absence of any structure of even size. Everything goes well with an odd size.

```
> draw(slicing1,size=19);
```

Prod(E, Prod(Z, Z), Prod(Prod(Prod(Prod(E, Prod(Z, Z), Prod(Prod(E, Prod(Z, Z), E), Prod(Z, Z), Prod(E, Prod(Z, Z), E))), Prod(Z, Z), E), Z, E), Prod(Z, Z), Prod(E, Prod(Z, Z), Prod(E, Prod(Z, Z), E))))

Jordan's couting of non-crossing Hamiltonian paths.

Problem.

Given n points on a circle, determine the number of hamiltonian paths that have no crossing edges.

Specification, random generation and couting.

We discuss here the simpler problem of specifying non-crossing hamiltonian paths that start at the designated root node of index 0. An initial segment of such a path can be continued in two ways, either by moving to the "next" point (an N-move) away from the root, or by crossing over to the opposite side (an X-move). The final step (the F-step) is fully determined by the context. Thus, the specification is:

```
> ham_path:=[H,{H=Prod(Sequence(Union(N,X)),F),F=Z,N=Z,X=Z},
  unlabelled]:
```

We count, draw, etc. For instance:

```
> seq(count(ham_path,size=n),n=0..20);
```

0, 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536, 131072, 262144, 524288

Naturally, the result is the powers of two. We leave it as an exercise to prove that the number of unconstrained hamiltonian paths is $n\,2^{(n-3)}$. The idea is to decompose such paths as either starting right, or left, or as a gluing (with suitable constraints) of two paths that emenate from the root node.

Other non-crossing configurations.

Discussion.

By the same principles, it is possible to enumerate, list exhaustively, draw at random, etc, a large number of geometrical configurations. Examples are general non-crossing graphs, forests, dissections, trees and graphs of bounded degree (Noy , September1996). We discuss here the case of forest.

Problem.

How many forests of trees with no crossing edges can be built from n points on a circle?

Specification, counting and generating functions.

A forest has a skeletton that is defined as the non-crossing tree that contains the vertex of smallest index. A (possibly empty) forest is then attached to each node of the skeleton. Thus forests are recursively definable as non-crossing trees of forests.
The corresponding specification is thus derived from that of NC-trees.

```
> forest:=[F,{F=Union(Epsilon,T),T=Prod(Z1,P),P=Sequence(B),
  B=Prod(P,Z1,P),Z1=Prod(Z,F)},unlabelled]:
> seq(count(forest,size=n),n=0..20);
```

1, 1, 2, 7, 33, 181, 1083, 6854, 45111, 305629, 2117283, 14929212, 106790500, 773035602, 5652275723, 41683912721, 309691336359, 2315772552485, 17415395593371, 131632335068744, 999423449413828

```
> draw(forest,size=20);
```

Prod(Prod(Z, Prod(Prod(Z, E), E)), Sequence(Prod(E, Prod(Z, E), E), Prod(E, Prod(Z, Prod(Prod(Z, Prod(Prod(Z, E), E)), E)), Sequence(Prod(Sequence(Prod(E, Prod(Z, E), Sequence(Prod(E, Prod(Z, E), E), Prod(Sequence(Prod(Sequence(Prod(E, Prod(Z, E), E)), Prod(Z, E), E), Prod(E, Prod(Z, E), E)), Prod(Z, E), E), Prod(E, Prod(Z, E), E), Prod(Sequence(Prod(Sequence(Prod(E, Prod(Z, E), E), Prod(E, Prod(Z, E), E)), Prod(Z, E), Sequence(Prod(E, Prod(Z, E), E)))), Prod(Z, E), E), Prod(E, Prod(Z, E), E))), Prod(Z, E), E)))))

```
> forest_sys:=gfsolve(op(2..3,forest),z);
```

$$forest\_sys := \{ Z(z) = z, Z1(z) = -\frac{z}{\mathrm{RootOf}(\_Z^3 z + \_Z^2 z + (-1-z)\_Z + 1)\, z - 1},$$
$$F(z) = -\frac{1}{\mathrm{RootOf}(\_Z^3 z + \_Z^2 z + (-1-z)\_Z + 1)\, z - 1},$$
$$B(z) = -\frac{\mathrm{RootOf}(\_Z^3 z + \_Z^2 z + (-1-z)\_Z + 1)^2\, z}{\mathrm{RootOf}(\_Z^3 z + \_Z^2 z + (-1-z)\_Z + 1)\, z - 1},$$
$$P(z) = \mathrm{RootOf}(\_Z^3 z + \_Z^2 z + (-1-z)\_Z + 1),$$
$$T(z) = -\frac{z\,\mathrm{RootOf}(\_Z^3 z + \_Z^2 z + (-1-z)\_Z + 1)}{\mathrm{RootOf}(\_Z^3 z + \_Z^2 z + (-1-z)\_Z + 1)\, z - 1} \}$$

```
> evala(subs(forest_sys,F(z)));
```

$$\mathrm{RootOf}(\_Z^3 z + \_Z^2 z + (-1-z)\_Z + 1)^2 z^2 + \mathrm{RootOf}(\_Z^3 z + \_Z^2 z + (-1-z)\_Z + 1)\, z^2 + \mathrm{RootOf}(\_Z^3 z + \_Z^2 z + (-1-z)\_Z + 1)\, z - z^2 + 1$$

We automatically obtain a theorem of Noy (1996).

```
> series(",z);
```

$$1 + z + 2z^2 + 7z^3 + 33z^4 + 181z^5 + O(z^6)$$

Conclusions.

We have demonstrated the way *Combstruct* can be used to do experimental combinatorics. In conjunction with *Gfun*, it can help generate conjectures and even derive automatic proofs of theorems in combinatorics.