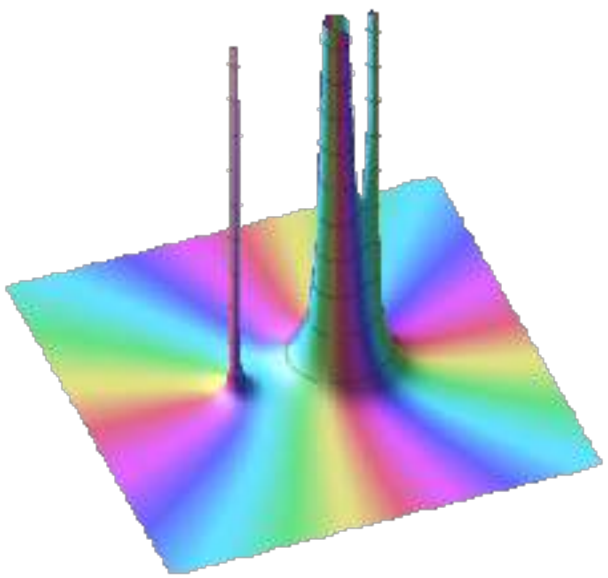
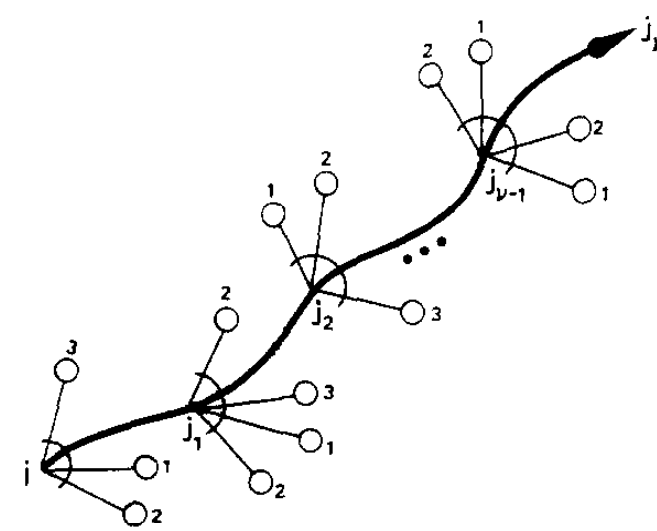
On (Effective) Analytic Combinatorics



Bruno Salvy

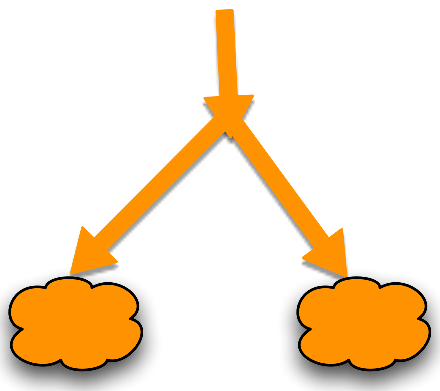
Inria & ENS de Lyon

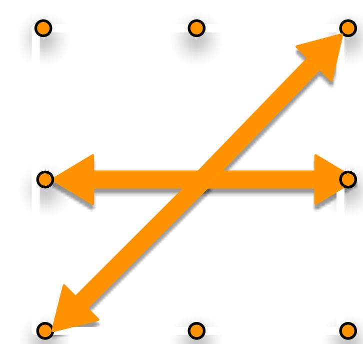


Séminaire Algorithmes

Dec. 2017

Combinatorics, Randomness and Analysis

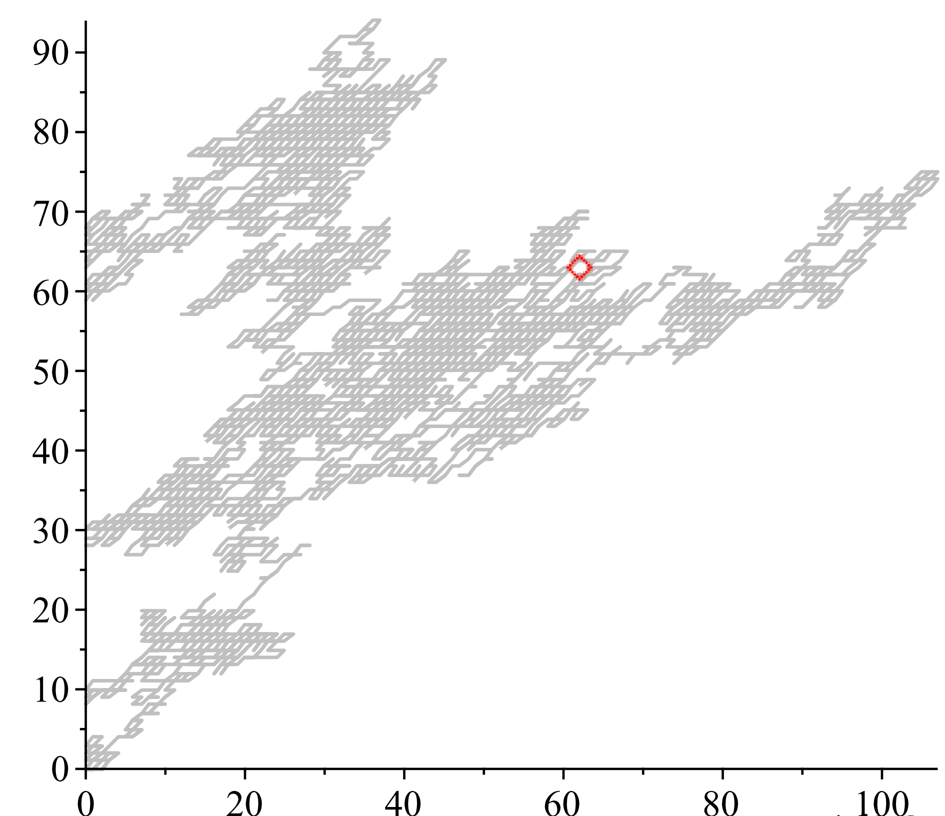




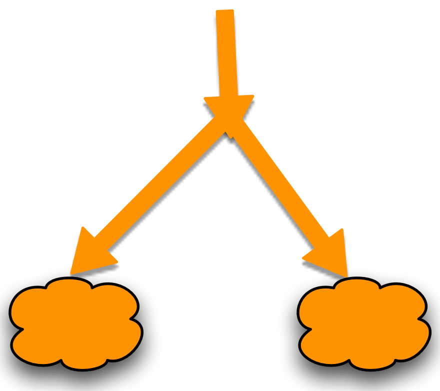
From **simple local** rules, a **global** structure arises**.**

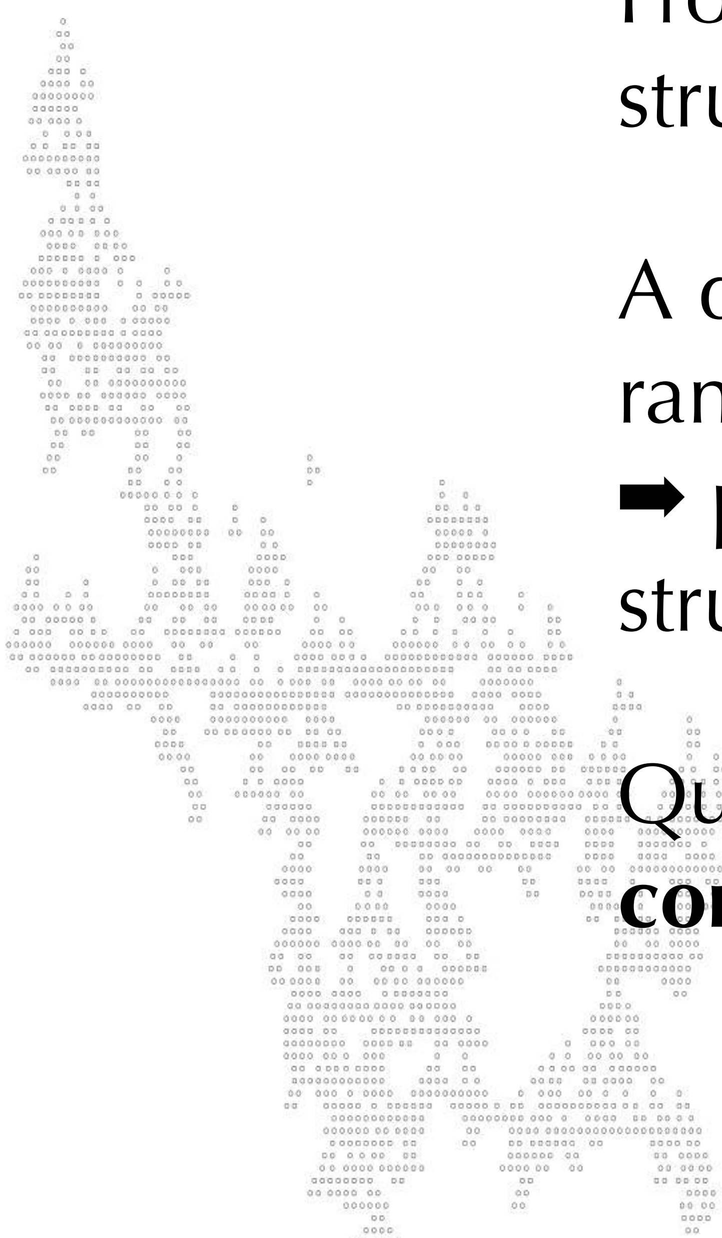
A quest for **universality** in random discrete structures**:**
➡ **probabilistic complexity** of structures and algorithms.

Quantitative results using **complex analysis.**



Combinatorics, Randomness and Analysis



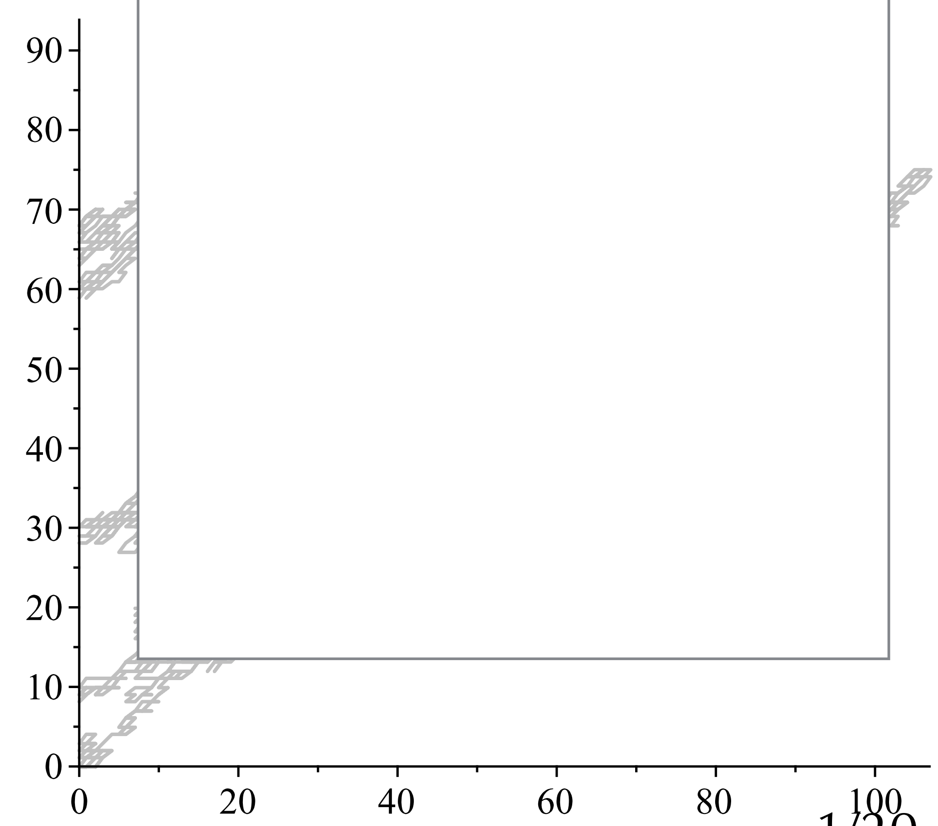


From **simple local** rules, a **global** structure arises**.**

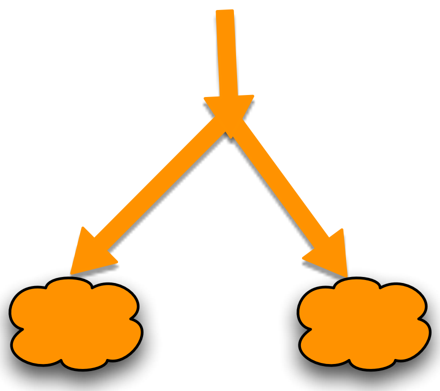
A quest for **universality** in random discrete structures**:**
➡ **probabilistic complexity** of structures and algorithms.

Quantitative results using **complex analysis.**

Typical questions



Combinatorics, Randomness and Analysis



From **simple local** rules, a **global** structure arises**.**

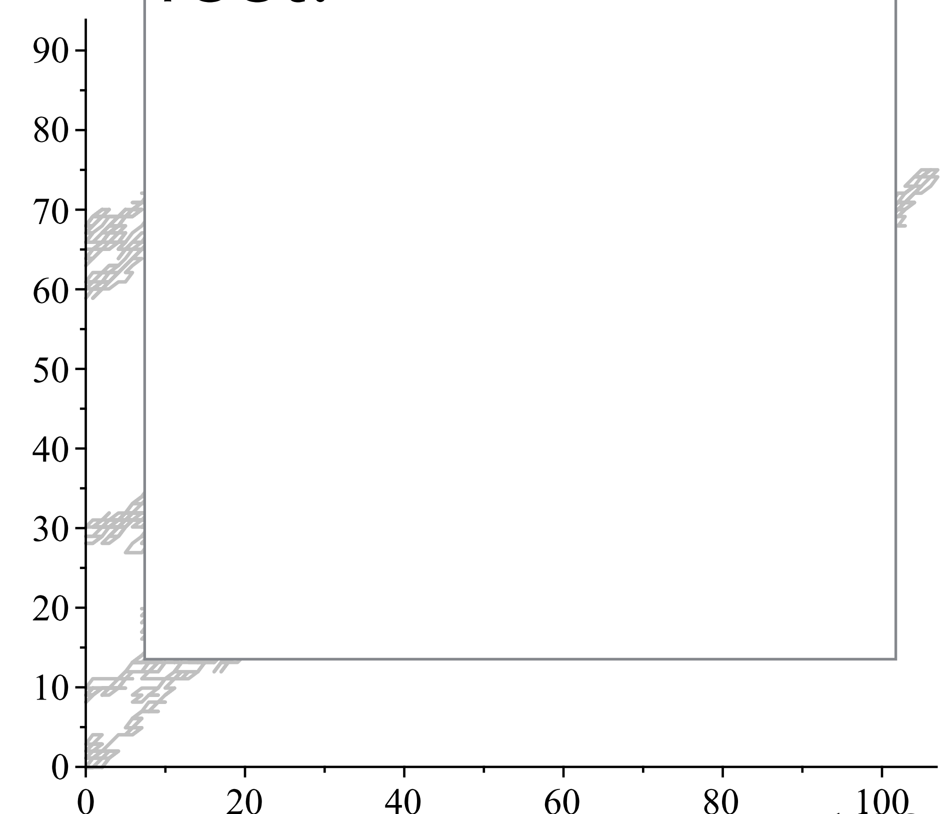
A quest for **universality** in random discrete structures**:**
➡ **probabilistic complexity** of structures and algorithms.

Quantitative results using **complex analysis.**

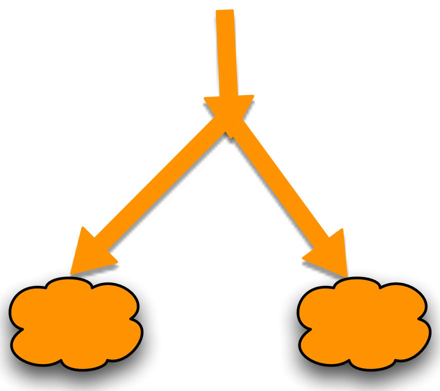
Typical questions

For a tree of size *n*,

. probability that a leaf is a child of the root?



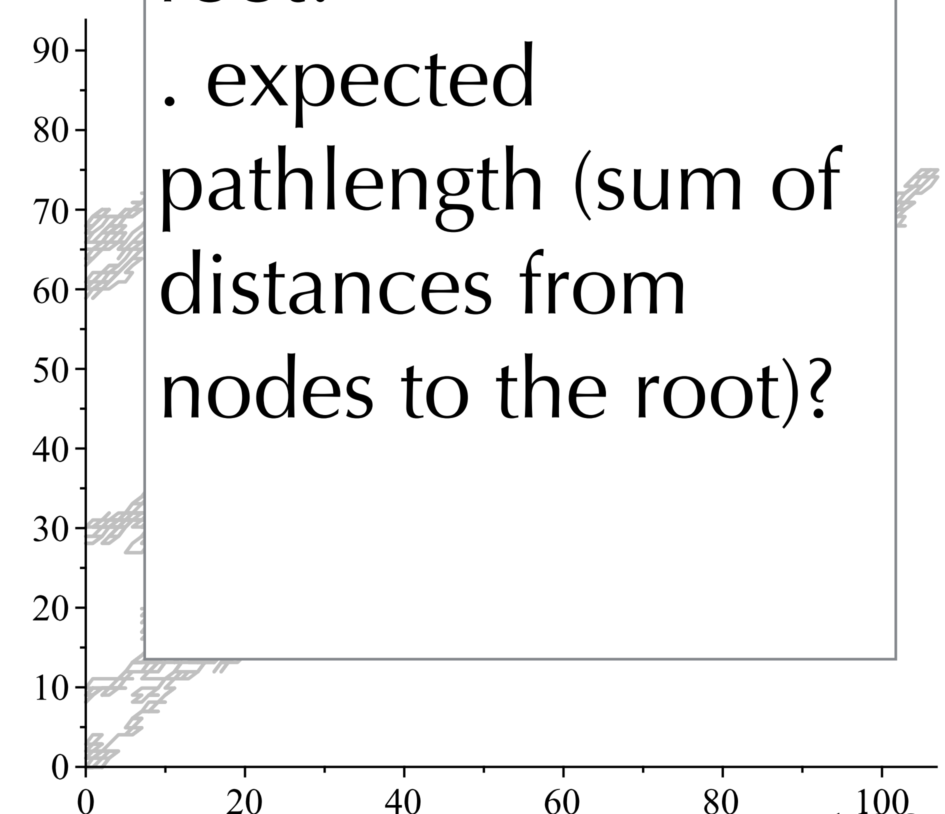
Combinatorics, Randomness and Analysis



From **simple local** rules, a **global** structure arises**.**

A quest for **universality** in random discrete structures**:**
➡ **probabilistic complexity** of structures and algorithms.

Quantitative results using **complex analysis.**

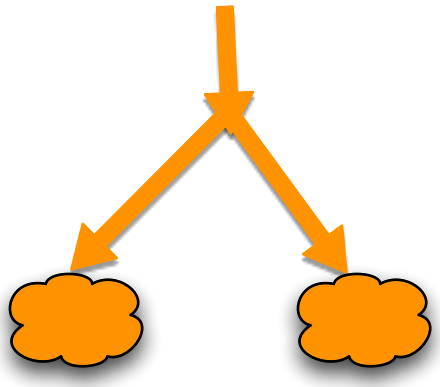


Typical questions

For a tree of size *n*,

. probability that a leaf is a child of the root?
. expected pathlength (sum of distances from nodes to the root)?

Combinatorics, Randomness and Analysis



From **simple local** rules, a **global** structure arises**.**

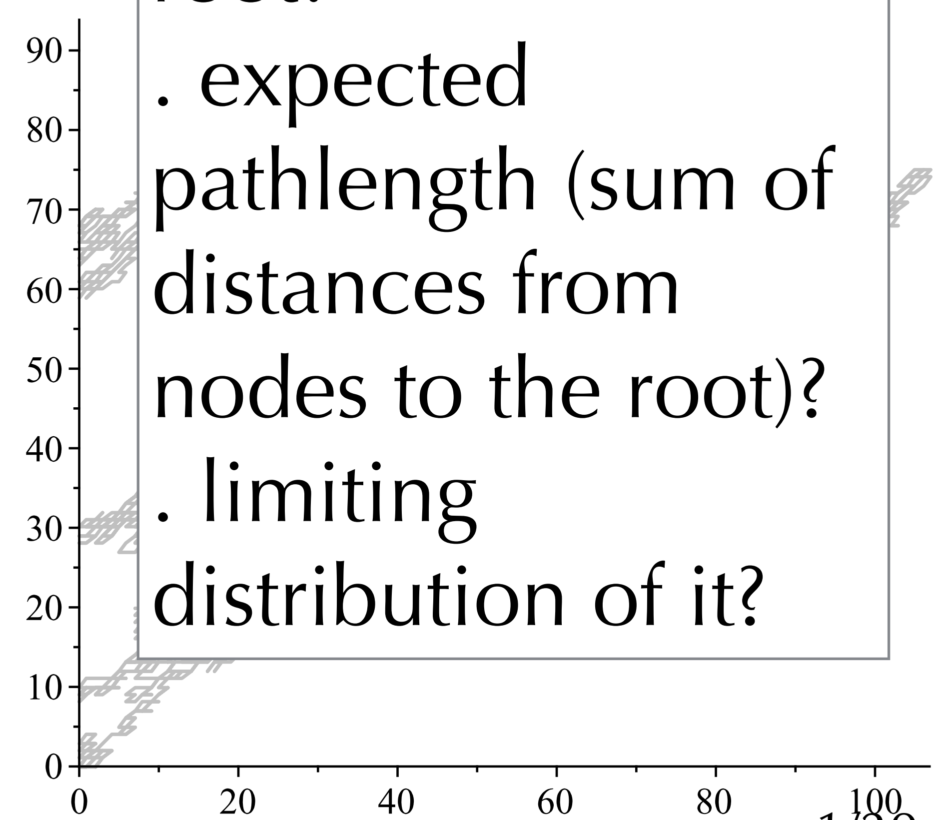
A quest for **universality** in random discrete structures**:**
➡ **probabilistic complexity** of structures and algorithms.

Quantitative results using **complex analysis.**

Typical questions

For a tree of size *n*,

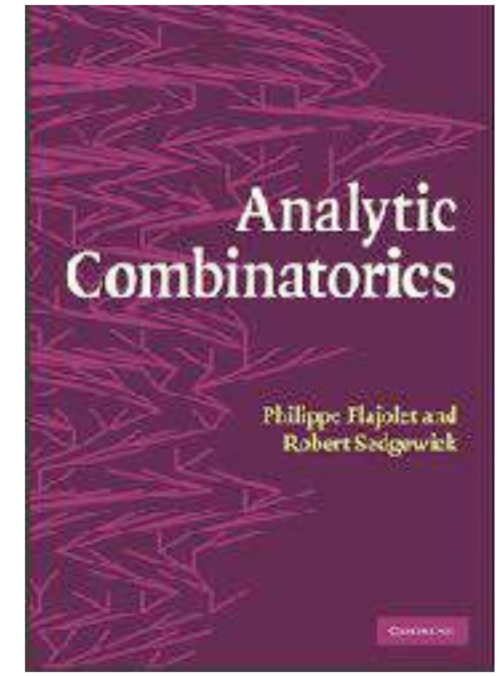
. probability that a leaf is a child of the root?
. expected pathlength (sum of distances from nodes to the root)?
. limiting distribution of it?



Philippe Flajolet,
the Father of Analytic Combinatorics

1948-2011

Planned complete works span
7 volumes of approx 600 pp. each.



2009

Combinatorial specification → Generating functions →
- counting;
- random generation;
- asymptotic analysis.

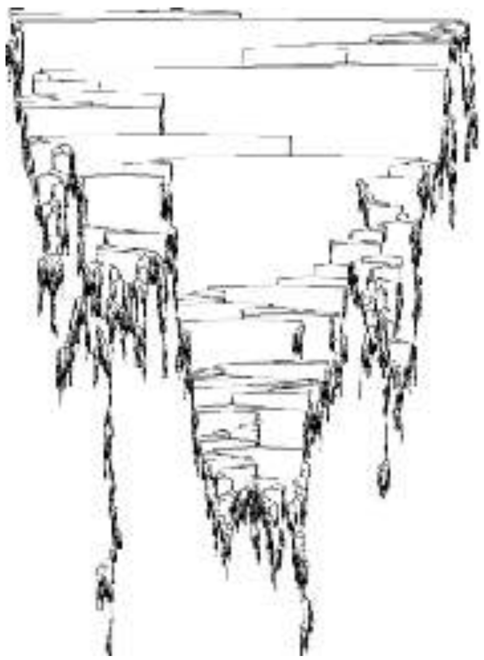
If you can specify, you can analyze.

Constructible Structures

Language: $1, \mathcal{Z}, +, \times,$ SEQ, SET, CYC
and recursion.

Constructible Structures

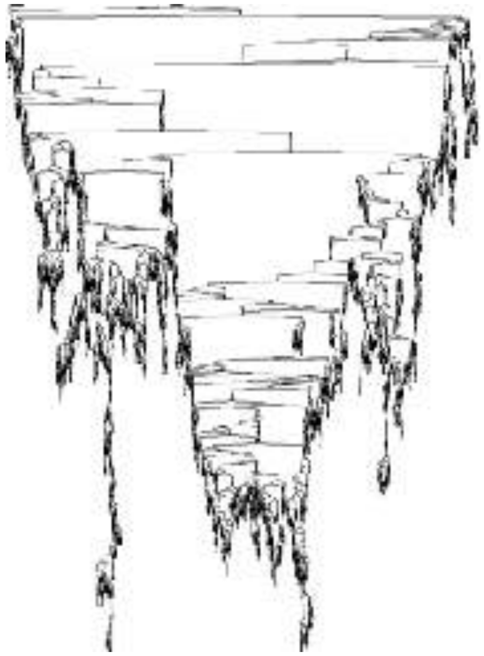
Language: $1, \mathcal{Z}, +, \times,$ SEQ, SET, CYC
and recursion.



- Binary trees: $\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$

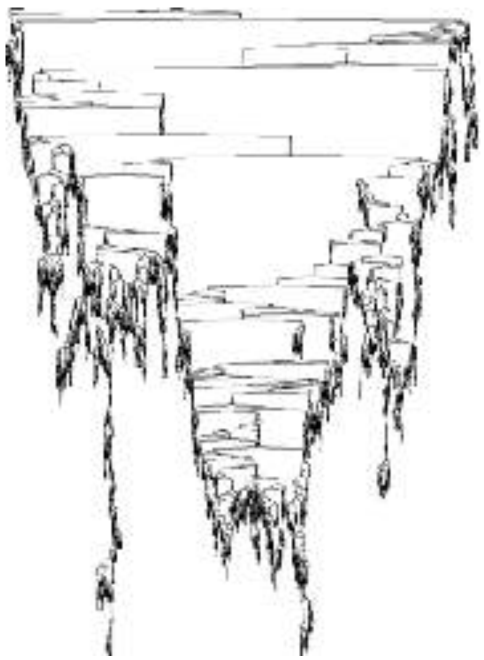
Constructible Structures

Language: $1, \mathcal{Z}, +, \times, \textsc{Seq}, \textsc{Set}, \textsc{Cyc}$ and recursion.



- Binary trees: $\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$
- Permutations: $\mathrm{Perm}=\textsc{Set}(\textsc{Cyc}(\mathcal{Z}))$;

Constructible Structures

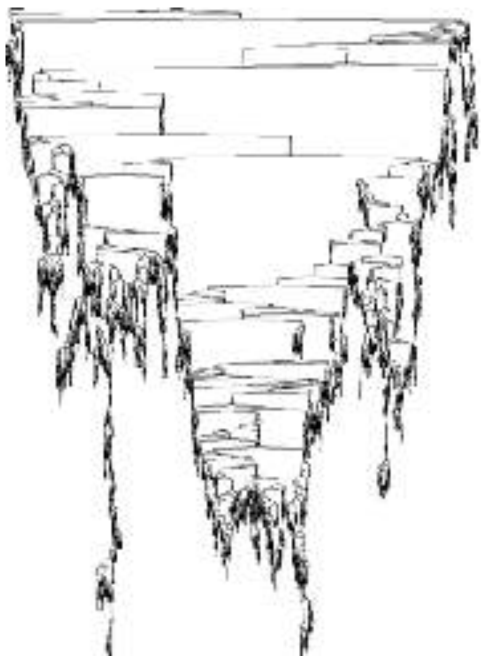


Language: $1, \mathcal{Z}, +, \times,$ SEQ, SET, CYC and recursion.

- Binary trees: $\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$
- Permutations: Perm=SET(CYC($\mathcal{Z}$));
- Trees: $\mathcal{T}=\mathcal{Z}\times$SET($\mathcal{T}$);

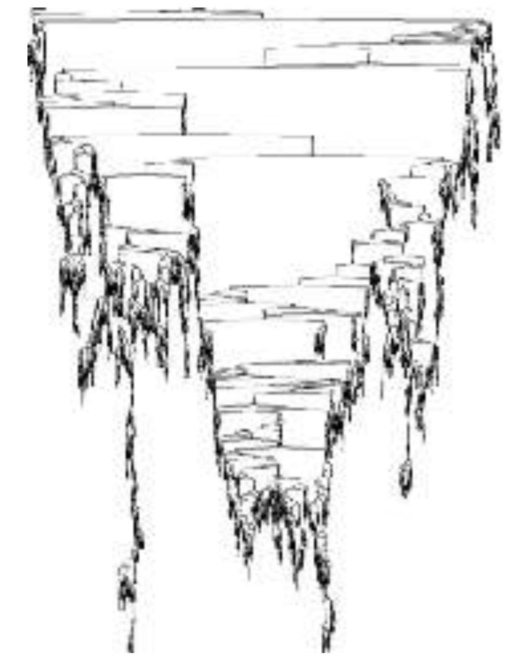
Constructible Structures

Language: $1, \mathcal{Z}, +, \times$, SEQ, SET, CYC and recursion.



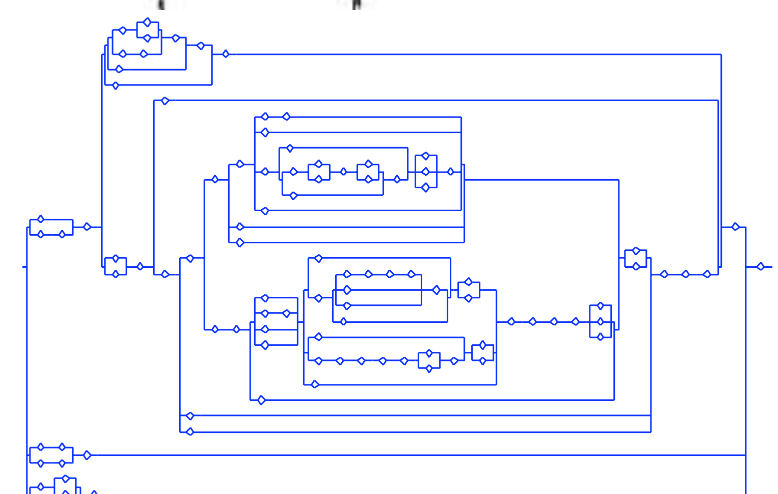
- Binary trees: $\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$
- Permutations: Perm=SET(CYC($\mathcal{Z}$));
- Trees: $\mathcal{T} = \mathcal{Z} \times$SET($\mathcal{T}$);
- Functional graphs: $\mathcal{F}$=SET(CYC($\mathcal{T}$));

Constructible Structures



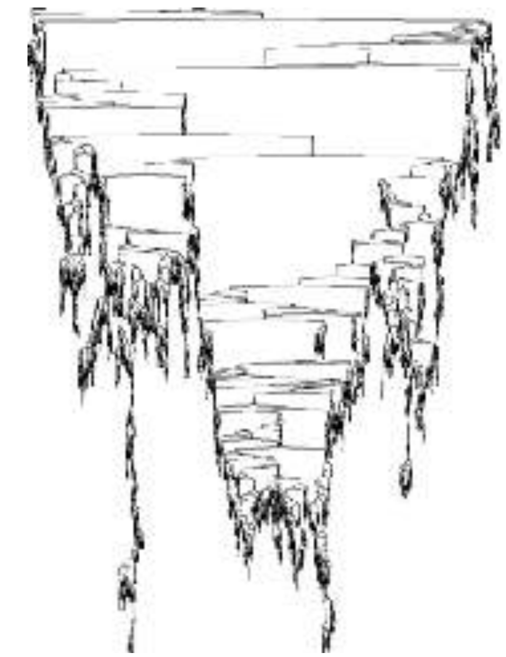
Language: $1, \mathcal{Z}, +, \times, \textsc{Seq}, \textsc{Set}, \textsc{Cyc}$ and recursion.

- Binary trees: $\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$
- Permutations: $\mathrm{Perm} = \textsc{Set}(\textsc{Cyc}(\mathcal{Z}))$;
- Trees: $\mathcal{T} = \mathcal{Z} \times \textsc{Set}(\mathcal{T})$;
- Functional graphs: $\mathcal{F} = \textsc{Set}(\textsc{Cyc}(\mathcal{T}))$;
- Series-parallel graphs:
  $\mathcal{G} = \mathcal{Z} + \mathcal{S} + \mathcal{P}$, $\mathcal{S} = \textsc{Seq}_{>0}(\mathcal{Z} + \mathcal{P})$, $\mathcal{P} = \textsc{Set}_{>0}(\mathcal{Z} + \mathcal{S})$;

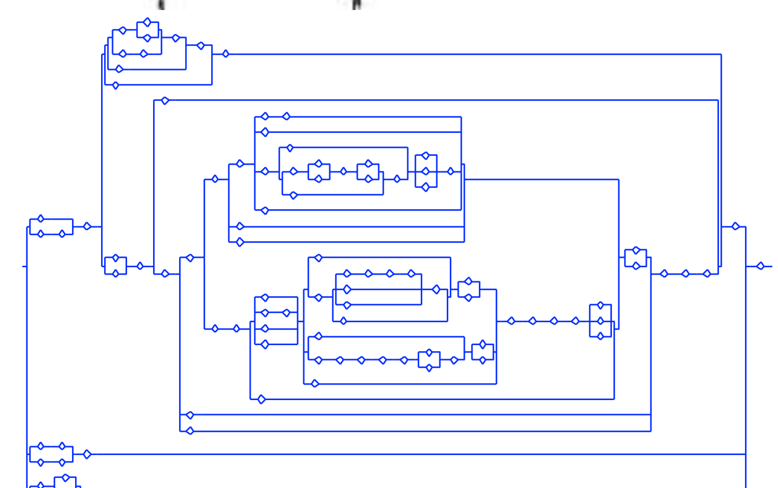


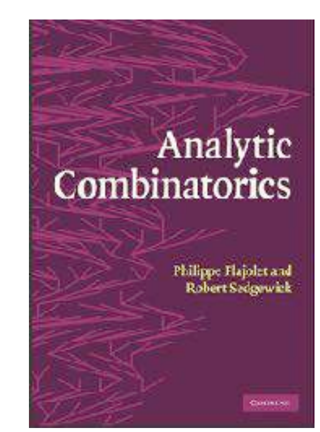
Constructible Structures

Language: $1, \mathcal{Z}, +, \times, \textsc{Seq}, \textsc{Set}, \textsc{Cyc}$
and recursion.



- Binary trees: $\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$
- Permutations: $\mathrm{Perm} = \textsc{Set}(\textsc{Cyc}(\mathcal{Z}))$;
- Trees: $\mathcal{T} = \mathcal{Z} \times \textsc{Set}(\mathcal{T})$;
- Functional graphs: $\mathcal{F} = \textsc{Set}(\textsc{Cyc}(\mathcal{T}))$;
- Series-parallel graphs:
  $\mathcal{G} = \mathcal{Z} + \mathcal{S} + \mathcal{P}$, $\mathcal{S} = \textsc{Seq}_{>0}(\mathcal{Z} + \mathcal{P})$, $\mathcal{P} = \textsc{Set}_{>0}(\mathcal{Z} + \mathcal{S})$;
- ...hundreds of examples in "the purple book".





I. Enumeration and Generating Functions

I. Enumeration and Generating Functions

$$F(z) = \sum_{n=0}^{\infty} f_n z^n$$

Example: Binary Trees

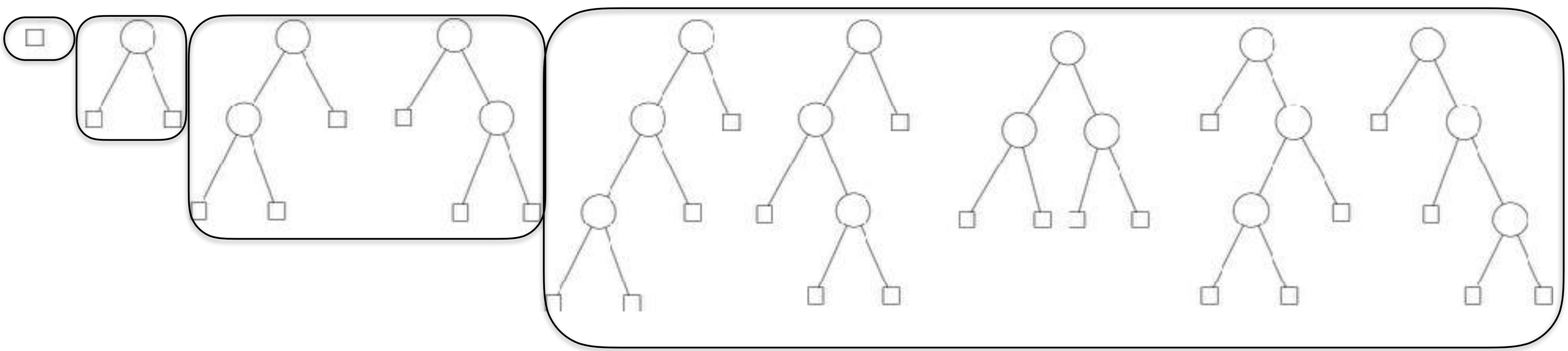
$$\mathscr{B} = 1 + \mathscr{Z} \times \mathscr{B} \times \mathscr{B}$$

$$B(z) = \sum_{n=0}^{\infty} B_n z^n$$

number of binary trees with *n* nodes

(Catalan)

$B_0 = B_1 = 1$ $B_2 = 2$ $B_3 = 5$



Example: Binary Trees

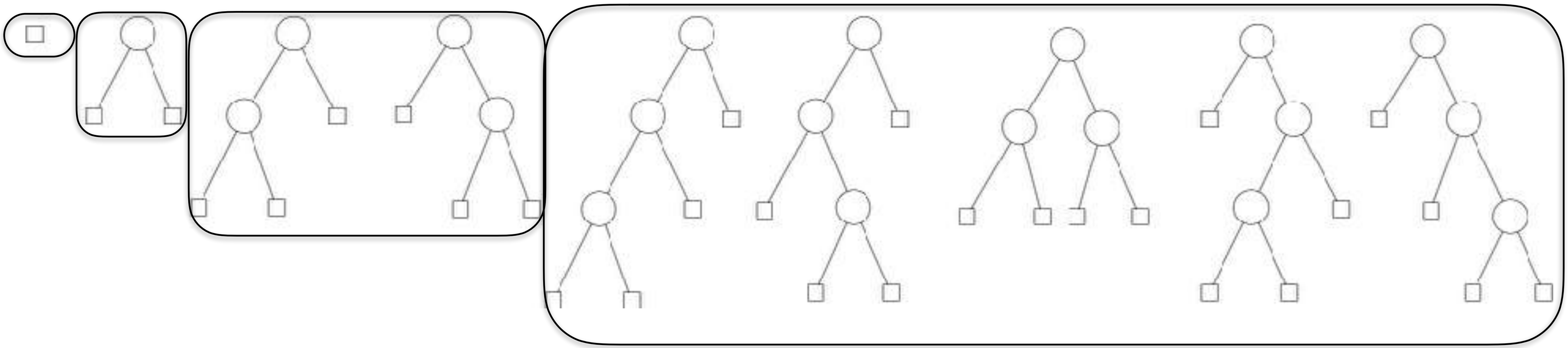
$$\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$$

$$B(z) = \sum_{n=0}^{\infty} B_n z^n$$

number of binary trees with *n* nodes

(Catalan)

$B_0 = B_1 = 1 \quad B_2 = 2 \qquad B_3 = 5$



For $n \geq 1, \quad B_n = \sum_{k=0}^{n-1} B_k B_{n-k-1}$

Example: Binary Trees

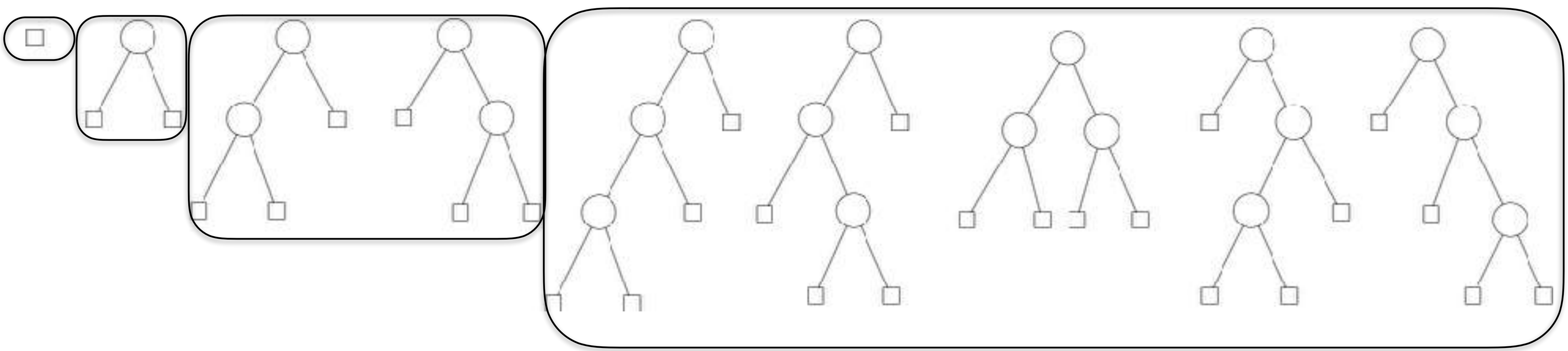
$$\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$$

$$B(z) = \sum_{n=0}^{\infty} B_n z^n$$

number of binary trees with *n* nodes

(Catalan)

$B_0 = B_1 = 1 \quad B_2 = 2 \quad\quad B_3 = 5$



$$\text{For } n \geq 1, \quad B_n = \sum_{k=0}^{n-1} B_k B_{n-k-1} \Rightarrow B_n z^n = \sum_{k=0}^{n-1} z(B_k z^k)(B_{n-k-1} z^{n-k-1})$$

Example: Binary Trees

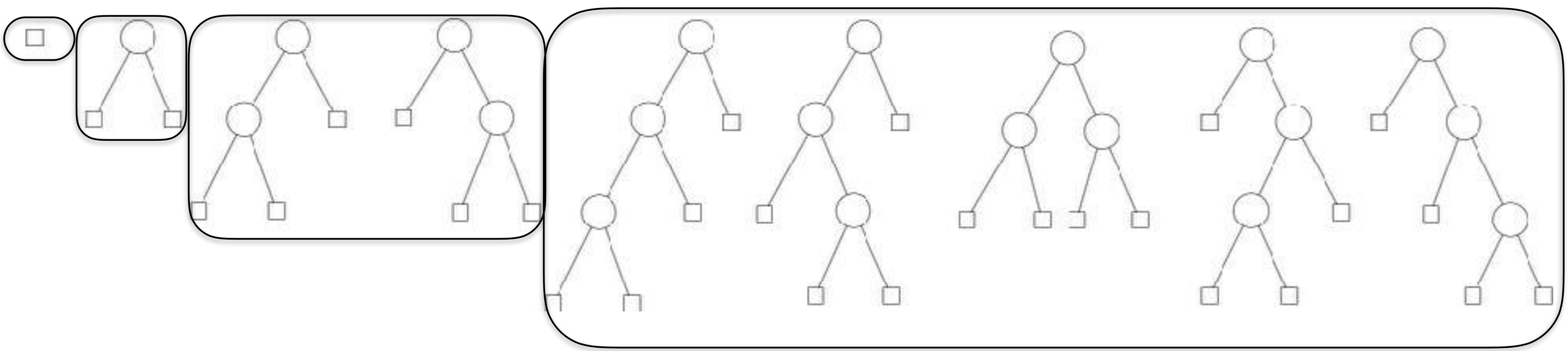
$$\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$$

$$B(z) = \sum_{n=0}^{\infty} B_n z^n$$

number of binary trees with *n* nodes

(Catalan)

$B_0 = B_1 = 1 \quad B_2 = 2 \qquad B_3 = 5$



$$\text{For } n \geq 1, \quad B_n = \sum_{k=0}^{n-1} B_k B_{n-k-1} \Rightarrow B_n z^n = \sum_{k=0}^{n-1} z(B_k z^k)(B_{n-k-1} z^{n-k-1})$$

$$B(z) = 1 + zB(z)^2$$

Example: Binary Trees

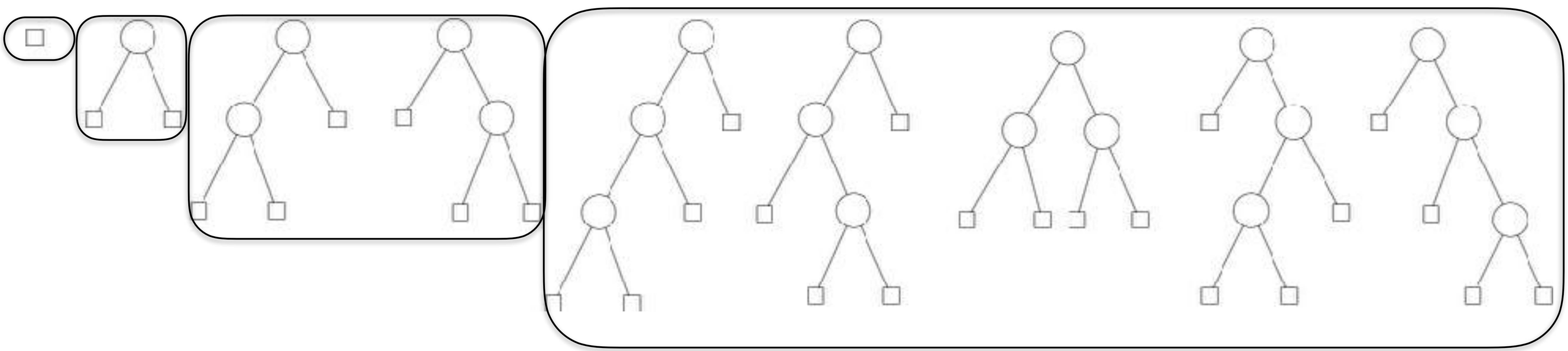
$$\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$$

$$B(z) = \sum_{n=0}^{\infty} B_n z^n$$

number of binary trees with n nodes

(Catalan)

$B_0 = B_1 = 1 \quad B_2 = 2 \qquad B_3 = 5$



$$\text{For } n \geq 1, \quad B_n = \sum_{k=0}^{n-1} B_k B_{n-k-1} \Rightarrow B_n z^n = \sum_{k=0}^{n-1} z(B_k z^k)(B_{n-k-1} z^{n-k-1})$$

$$B(z) = 1 + zB(z)^2 = 1 + z + 2z^2 + 5z^3 + \dots$$

Example: Binary Trees

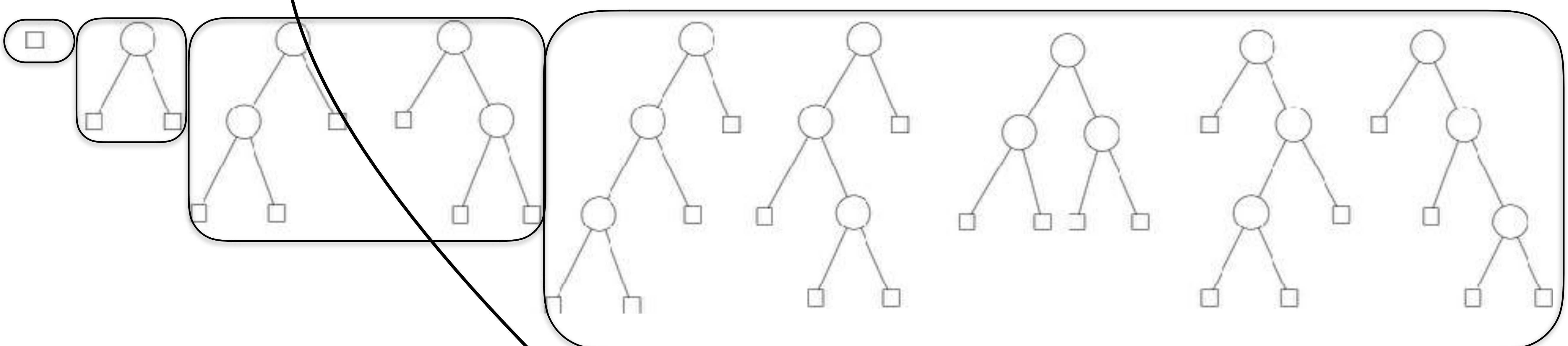
$$\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$$

$$B(z) = \sum_{n=0}^{\infty} B_n z^n$$

number of binary trees with *n* nodes

(Catalan)

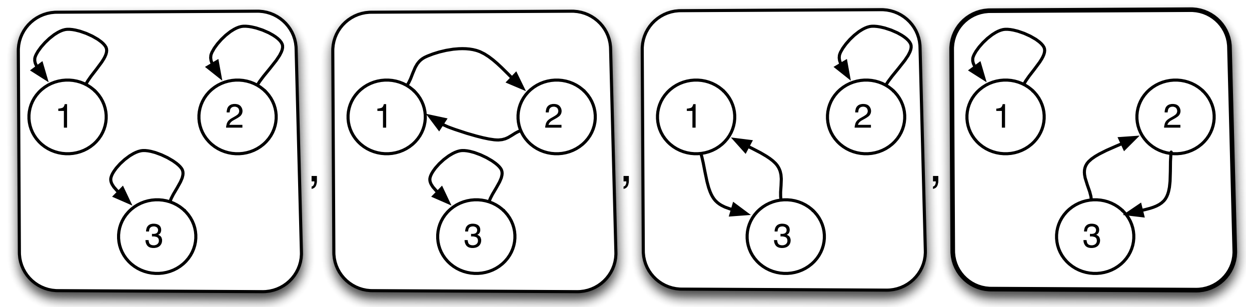
$B_0 = B_1 = 1$ $\quad B_2 = 2$ $\quad B_3 = 5$



For $n \geq 1, \quad B_n = \sum_{k=0}^{n-1} B_k B_{n-k-1} \Rightarrow B_n z^n = \sum_{k=0}^{n-1} z(B_k z^k)(B_{n-k-1} z^{n-k-1})$

$$B(z) = 1 + zB(z)^2 = 1 + z + 2z^2 + 5z^3 + \cdots$$

What do we count?

$\mathrm{Inv}[\{1,2,3\}] = \{$  $\}$

4 involutions;

3 of them permuted by $\mathfrak{S}_3 \rightarrow$ 2 *unlabelled structures*.

Exponential generating series (EGF):

$$F(z) = \sum_{n=0}^{\infty} f_n \frac{z^n}{n!}, \quad f_n = \text{nb. labelled structs of size } n. \qquad \mathrm{Inv}_3(z) = \frac{2}{3} z^3$$

Ordinary generating series (OGF):

$$\tilde{F}(z) = \sum_{n=0}^{\infty} \tilde{f}_n z^n, \quad \tilde{f}_n = \text{nb. unlabelled structs of size } n. \qquad \widetilde{\mathrm{Inv}}_3(z) = 2z^3$$

A Dictionary for Generating Series

Language: $1, \mathcal{Z}, +, \times, \textsc{Seq}, \textsc{Set}, \textsc{Cyc}$ and recursion.

describes ***constructible structures***

A Dictionary for Generating Series

Language: $1, \mathcal{Z}$, $+$, $\times$, SEQ, SET, CYC and recursion.

describes ***constructible structures***

Structure	EGF	OGF
$\mathcal{A}+\mathcal{B}$	$A(z)+B(z)$	$\tilde{A}(z)+\tilde{B}(z)$
$\mathcal{A}\times\mathcal{B}$	$A(z)\times B(z)$	$\tilde{A}(z)\times\tilde{B}(z)$

A Dictionary for Generating Series

Language: $1, \mathcal{Z}, +, \times$, Seq, Set, Cyc and recursion.

describes ***constructible structures***

Structure	EGF	OGF
$\mathcal{A}+\mathcal{B}$	$A(z)+B(z)$	$\tilde{A}(z)+\tilde{B}(z)$
$\mathcal{A}\times\mathcal{B}$	$A(z)\times B(z)$	$\tilde{A}(z)\times\tilde{B}(z)$
Seq($\mathcal{A}$)	$\frac{1}{1-A(z)}$	$\frac{1}{1-\tilde{A}(z)}$

A Dictionary for Generating Series

Language: $1, \mathcal{Z}, +, \times,$ SEQ, SET, CYC and recursion.

describes ***constructible structures***

Structure	EGF	OGF
$\mathcal{A}+\mathcal{B}$	$A(z)+B(z)$	$\tilde{A}(z)+\tilde{B}(z)$
$\mathcal{A}\times\mathcal{B}$	$A(z)\times B(z)$	$\tilde{A}(z)\times\tilde{B}(z)$
SEQ($\mathcal{A}$)	$\frac{1}{1-A(z)}$	$\frac{1}{1-\tilde{A}(z)}$
SET($\mathcal{A}$)	$\exp(A(z))$	$\exp(\sum \tilde{A}(z^i)/i)$

A Dictionary for Generating Series

Language: 1, $\mathcal{Z}$, +, ×, SEQ, SET, CYC and recursion.

describes ***constructible structures***

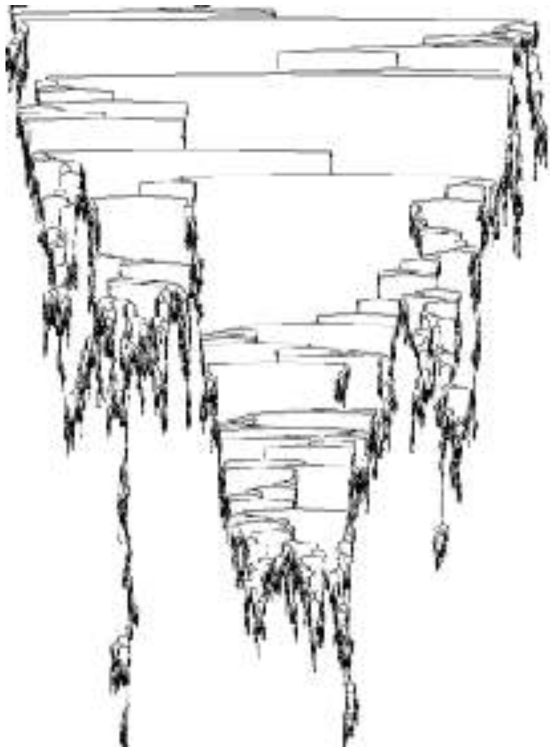
Structure	EGF	OGF
$\mathcal{A}+\mathcal{B}$	$A(z)+B(z)$	$\tilde{A}(z)+\tilde{B}(z)$
$\mathcal{A}\times\mathcal{B}$	$A(z)\times B(z)$	$\tilde{A}(z)\times\tilde{B}(z)$
SEQ($\mathcal{A}$)	$\frac{1}{1-A(z)}$	$\frac{1}{1-\tilde{A}(z)}$
SET($\mathcal{A}$)	$\exp(A(z))$	$\exp(\sum\tilde{A}(z^i)/i)$
CYC($\mathcal{A}$)	$\log\frac{1}{1-A(z)}$	$\sum\frac{\phi(i)}{i}\log\frac{1}{1-\tilde{A}(z^i)}$

Examples

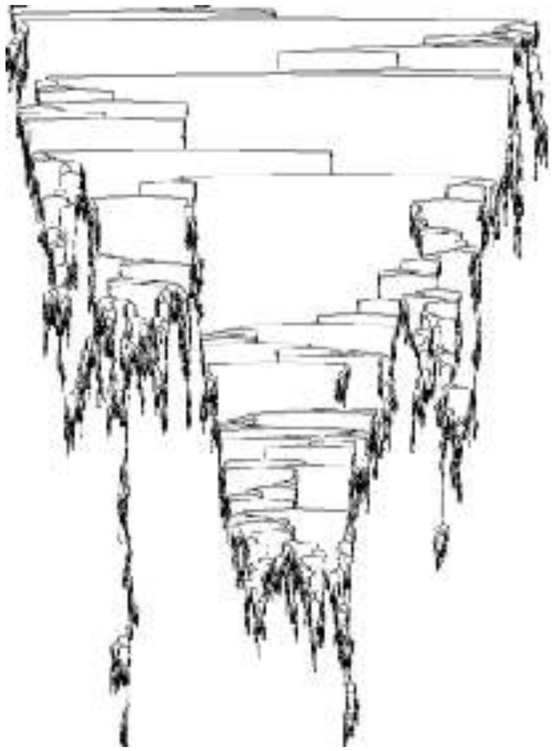
Examples

Binary trees: $\mathscr{B}$=1+$\mathscr{Z}$x$\mathscr{B}$x$\mathscr{B}$

$$\longrightarrow B(z) = 1 + zB(z)^2 = \tilde{B}(z)$$



Examples



Binary trees: $\mathscr{B}=1+\mathscr{Z}\times\mathscr{B}\times\mathscr{B}$

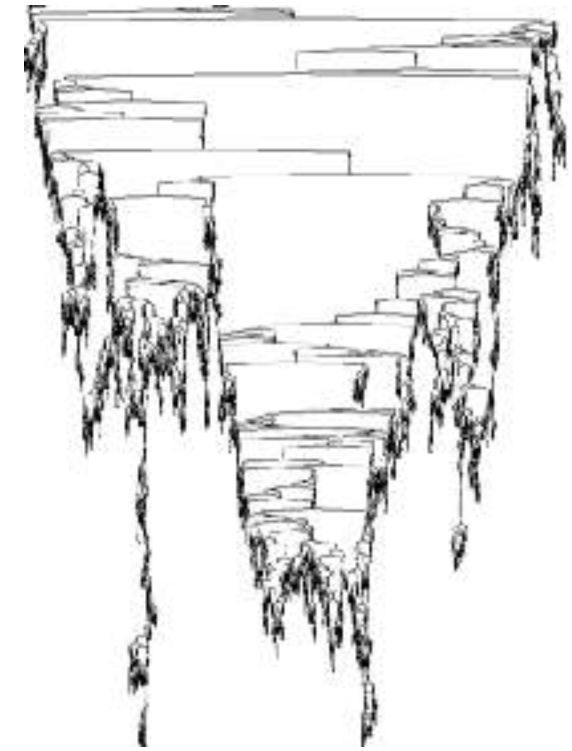
$$\longrightarrow B(z) = 1 + zB(z)^2 = \tilde{B}(z)$$

Cayley trees: $\mathscr{T}=\mathscr{Z}\times\textsc{Set}(\mathscr{T})$

$$\longrightarrow T(z) = z\exp(T(z));$$

$$\longrightarrow \tilde{T}(z) = z\exp\left(\tilde{T}(z) + \frac{1}{2}\tilde{T}(z^2) + \frac{1}{3}\tilde{T}(z^3) + \cdots\right)$$

Examples



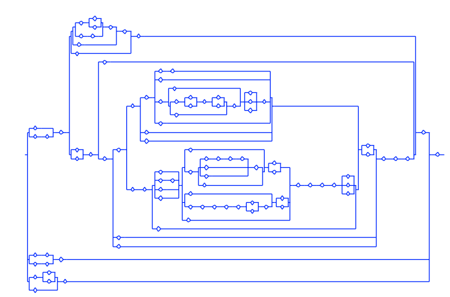
Binary trees: $\mathscr{B}=1+\mathscr{Z}\times\mathscr{B}\times\mathscr{B}$

$$\longrightarrow B(z) = 1 + zB(z)^2 = \tilde{B}(z)$$

Cayley trees: $\mathscr{T}=\mathscr{Z}\times\textsc{Set}(\mathscr{T})$

$$\longrightarrow T(z) = z\exp(T(z));$$

$$\longrightarrow \tilde{T}(z) = z\exp\left(\tilde{T}(z) + \frac{1}{2}\tilde{T}(z^2) + \frac{1}{3}\tilde{T}(z^3) + \cdots\right)$$

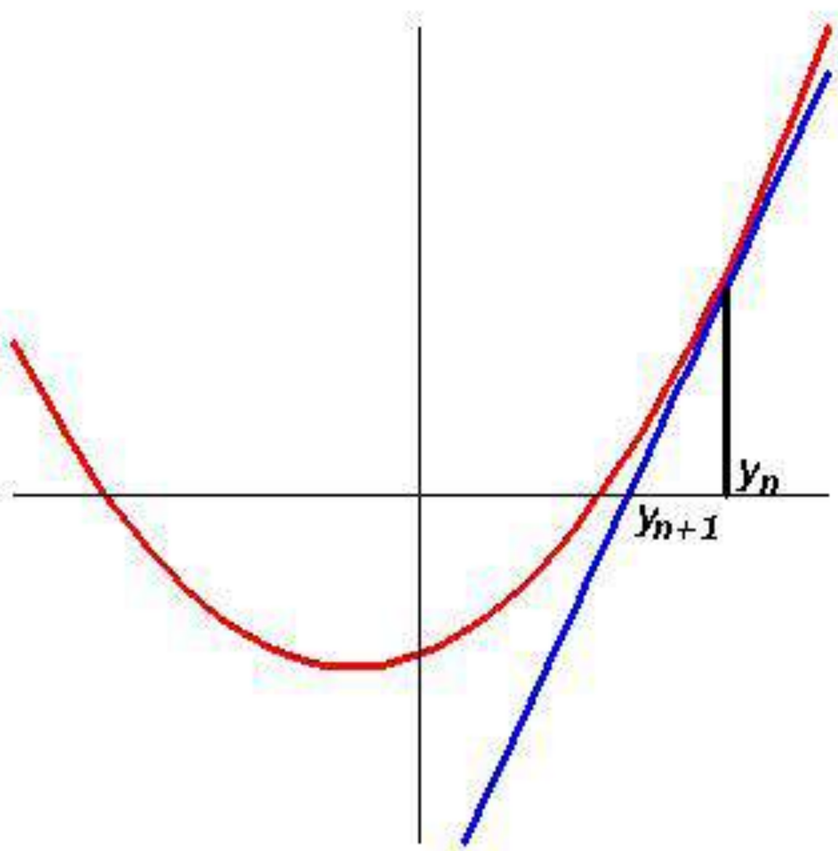


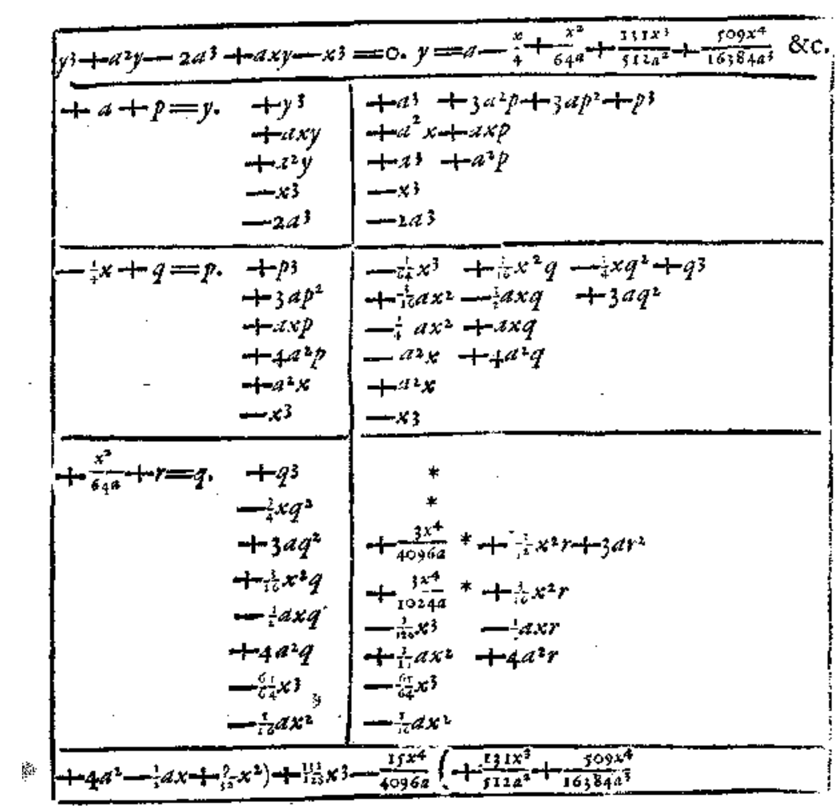
Series-parallel graphs:

$$\mathcal{G} = \mathcal{Z} + \mathcal{S} + \mathcal{P}, \mathcal{S} = \textsc{Seq}_{>0}(\mathcal{Z}+\mathcal{P}), \mathcal{P} = \textsc{Set}_{>0}(\mathcal{Z}+\mathcal{S})$$

$$\longrightarrow \left\{ G(z) = z + S(z) + P(z), S(z) = \frac{1}{1-z-P(z)} - 1, P(z) = e^{z+S(z)} - 1 \right\}$$

II. Newton Iteration and Fast Enumeration





1671

Numerical Newton Iteration

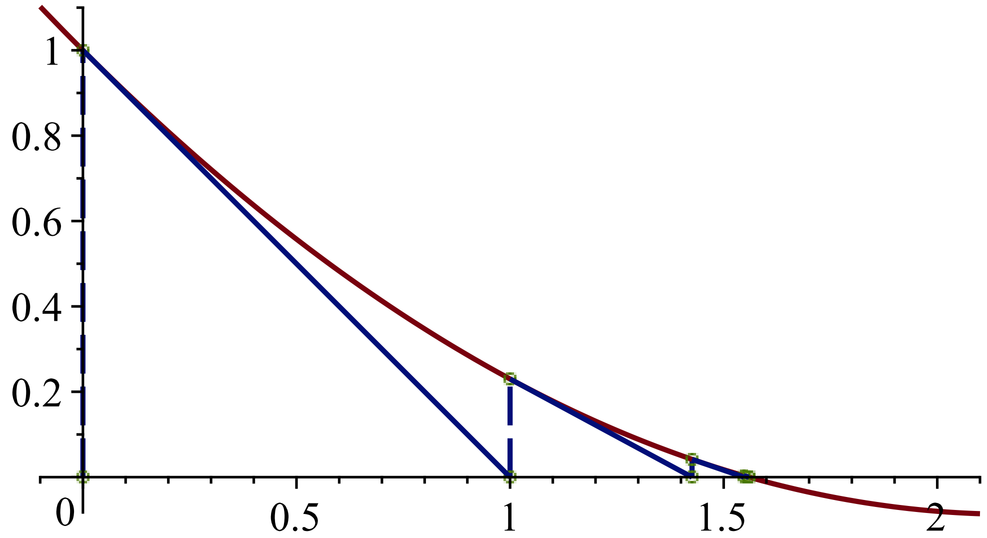
To solve $\phi(y) = 0$, iterate

$$y^{[n+1]} = y^{[n]} + u^{[n]},$$

with $\phi(y^{[n]}) + \phi'(y^{[n]})u^{[n]} = 0.$

$$\phi(y) = 1 + zy^2 - y$$

$$z = 0.23$$



Numerical Newton Iteration

To solve $\phi(y) = 0$, iterate

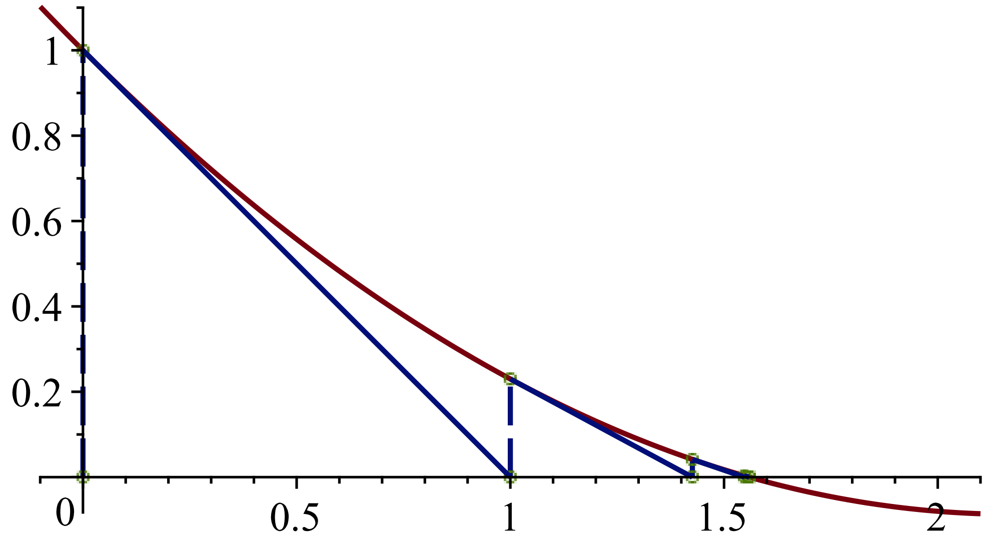
$$y^{[n+1]} = y^{[n]} + u^{[n]},$$

with $\phi(y^{[n]}) + \phi'(y^{[n]})u^{[n]} = 0.$

$$\phi(y) = 1 + zy^2 - y$$

$$y^{[n+1]} = \mathcal{N}(y^{[n]}) = y^{[n]} + \frac{1 + z{y^{[n]}}^2 - y^{[n]}}{1 - 2zy^{[n]}}$$

$$z = 0.23$$



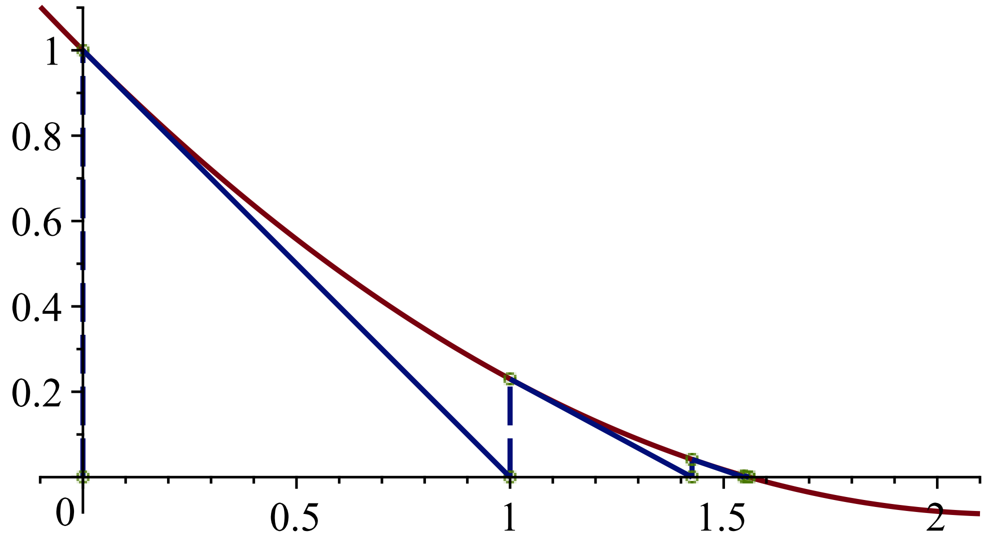
Numerical Newton Iteration

To solve $\phi(y) = 0$, iterate
$$y^{[n+1]} = y^{[n]} + u^{[n]},$$
with $\phi(y^{[n]}) + \phi'(y^{[n]})u^{[n]} = 0.$

$$\phi(y) = 1 + zy^2 - y$$

$$y^{[n+1]} = \mathcal{N}(y^{[n]}) = y^{[n]} + \frac{1 + zy^{[n]^2} - y^{[n]}}{1 - 2zy^{[n]}}$$

$z = 0.23$



$$y^{[0]} = 0,$$
$$y^{[1]} = 1.0000000000000000,$$
$$y^{[2]} \simeq 1.4259259259259259,$$
$$y^{[3]} \simeq 1.5471933181836303,$$
$$y^{[4]} \simeq 1.5589256602748822,$$
$$y^{[5]} \simeq 1.5590375713926592,$$
$$y^{[6]} \simeq 1.5590375815769151$$

Quadratic convergence

Newton Iteration for Power Series

Same Newton Iteration

$$y^{[n+1]} = \mathcal{N}(y^{[n]}) = y^{[n]} + \frac{1 + z{y^{[n]}}^2 - y^{[n]}}{1 - 2zy^{[n]}}$$

Newton Iteration for Power Series

Same Newton Iteration

$$y^{[n+1]} = \mathcal{N}(y^{[n]}) = y^{[n]} + \frac{1 + z{y^{[n]}}^2 - y^{[n]}}{1 - 2zy^{[n]}}$$

$$y^{[0]} = 0$$

$$y^{[1]} = 1$$

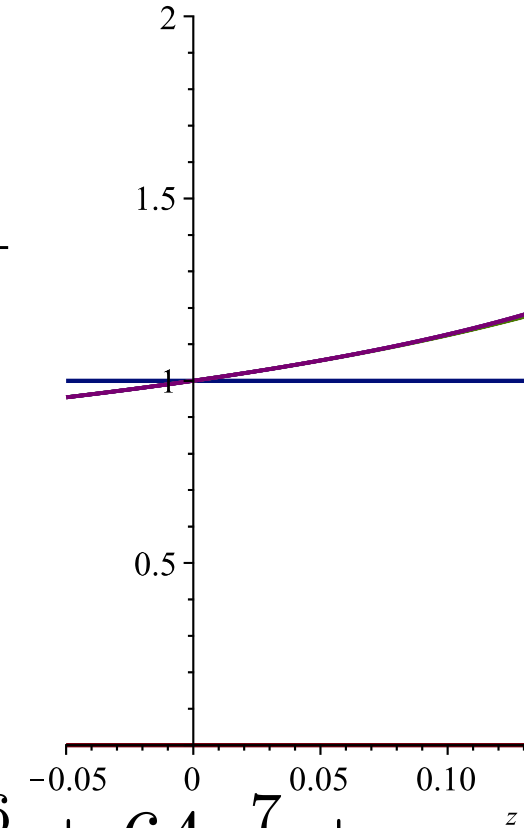
$$y^{[2]} = 1 + z + 2z^2 + 4z^3 + 8z^4 + 16z^5 + 32z^6 + 64z^7 + \cdots$$

$$y^{[3]} = 1 + z + 2z^2 + 5z^3 + 14z^4 + 42z^5 + 132z^6 + 428z^7 + \cdots$$

Newton Iteration for Power Series

Same Newton Iteration

$$y^{[n+1]} = \mathcal{N}(y^{[n]}) = y^{[n]} + \frac{1 + z{y^{[n]}}^2 - y^{[n]}}{1 - 2zy^{[n]}}$$



$$y^{[0]} = 0$$

$$y^{[1]} = 1$$

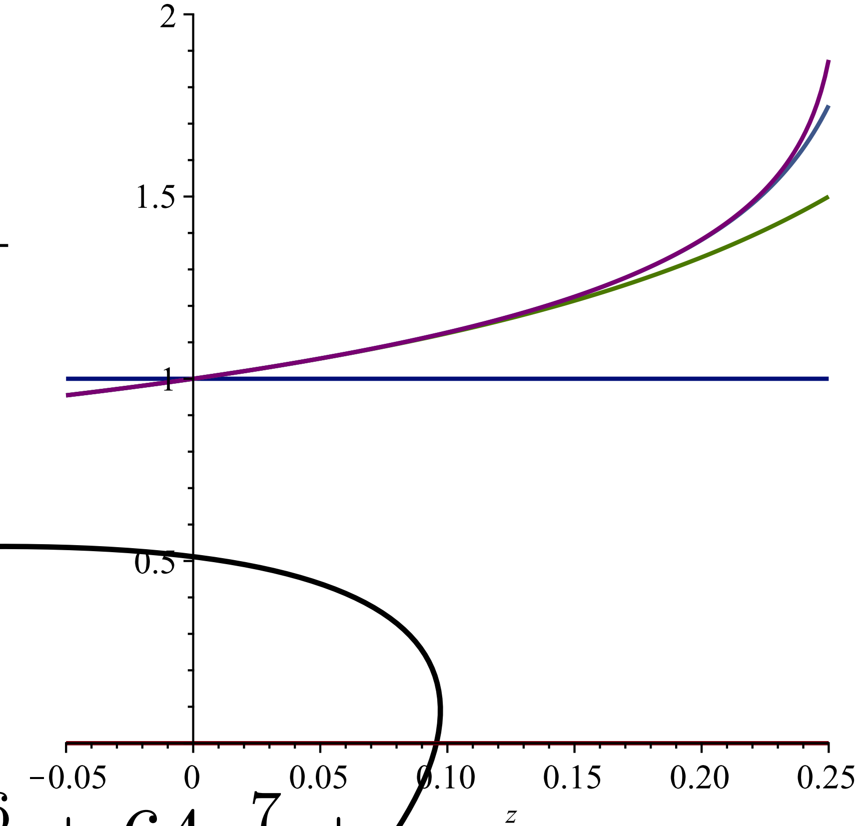
$$y^{[2]} = 1 + z + 2z^2 + 4z^3 + 8z^4 + 16z^5 + 32z^6 + 64z^7 + \cdots$$

$$y^{[3]} = 1 + z + 2z^2 + 5z^3 + 14z^4 + 42z^5 + 132z^6 + 428z^7 + \cdots$$

Newton Iteration for Power Series

Same Newton Iteration

$$y^{[n+1]} = \mathcal{N}(y^{[n]}) = y^{[n]} + \frac{1 + z{y^{[n]}}^2 - y^{[n]}}{1 - 2zy^{[n]}}$$



$$y^{[0]} = 0$$

$$y^{[1]} = 1$$

$$y^{[2]} = 1 + z + 2z^2 + 4z^3 + 8z^4 + 16z^5 + 32z^6 + 64z^7 + \cdots$$

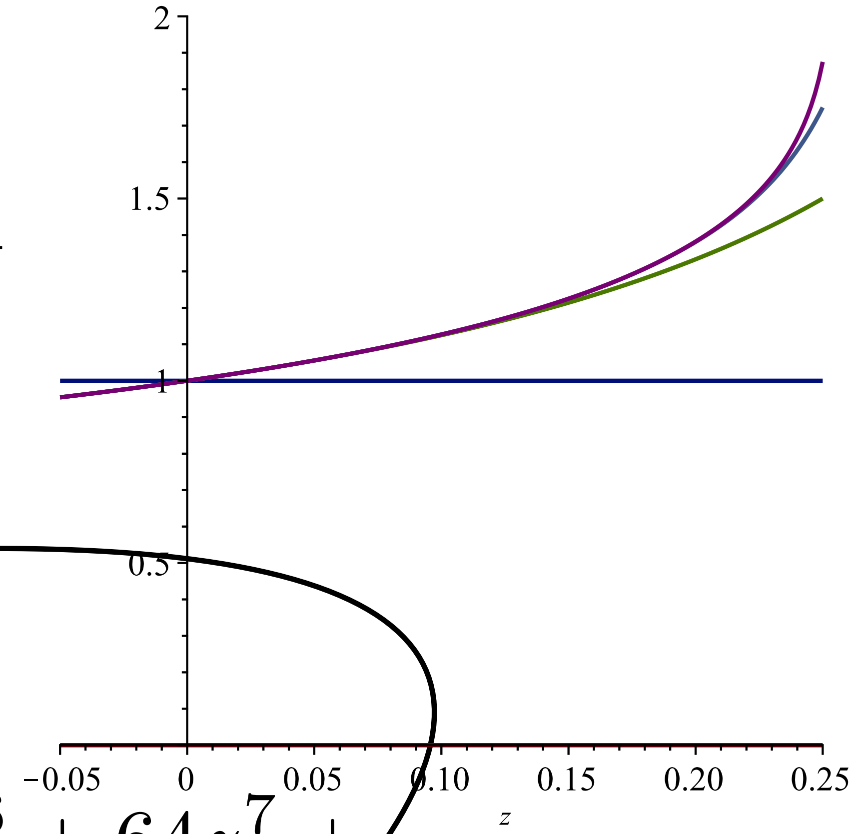
$$y^{[3]} = 1 + z + 2z^2 + 5z^3 + 14z^4 + 42z^5 + 132z^6 + 428z^7 + \cdots$$

Proving numerical convergence requires control over the tails

Newton Iteration for Power Series

Same Newton Iteration

$$y^{[n+1]} = \mathcal{N}(y^{[n]}) = y^{[n]} + \frac{1 + z{y^{[n]}}^2 - y^{[n]}}{1 - 2zy^{[n]}}$$



Proving numerical convergence requires control over the tails

$$y^{[0]} = 0$$

$$y^{[1]} = 1$$

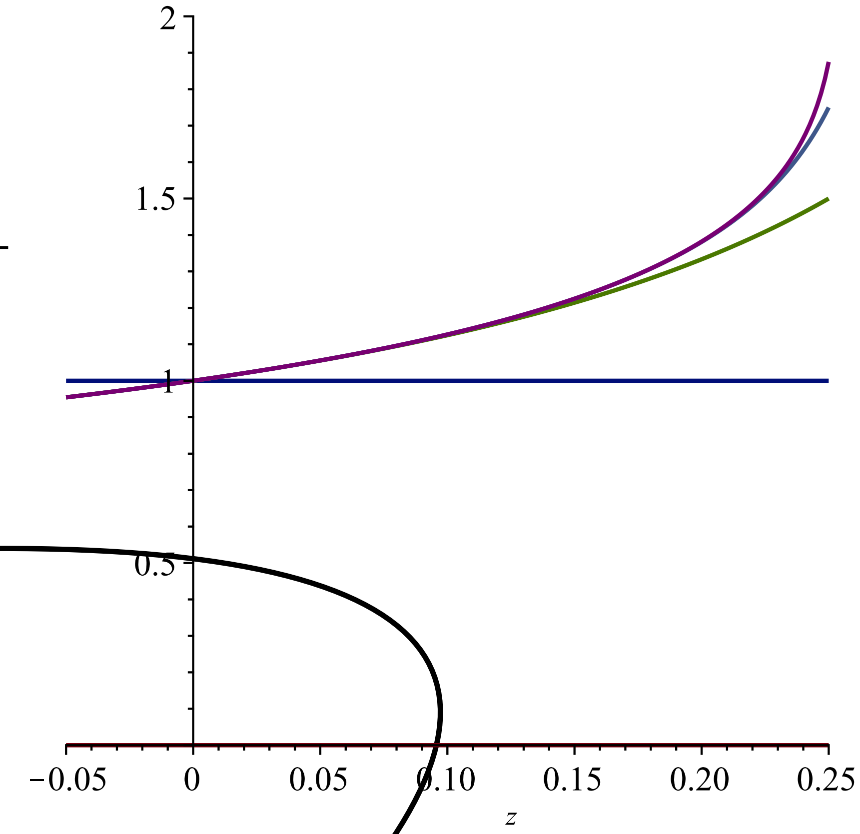
$$y^{[2]} = 1 + z + 2z^2 + 4z^3 + 8z^4 + 16z^5 + 32z^6 + 64z^7 + \cdots$$

$$y^{[3]} = 1 + z + 2z^2 + 5z^3 + 14z^4 + 42z^5 + 132z^6 + 428z^7 + \cdots$$

On power series: $y - y^{[\infty]} = O(z^m) \Longrightarrow \mathcal{N}(y) - y^{[\infty]} = O(z^{2m(+1)})$

Newton Iteration for Power Series

Same Newton Iteration $\quad y^{[n+1]} = \mathcal{N}(y^{[n]}) = y^{[n]} + \dfrac{1 + z{y^{[n]}}^2 - y^{[n]}}{1 - 2zy^{[n]}}$



$y^{[0]} = 0$

$y^{[1]} = 1$

$y^{[2]} = 1 + z + 2z^2$

$y^{[3]} = 1 + z + 2z^2 + 5z^3 + 14z^4 + 42z^5 + 132z^6$

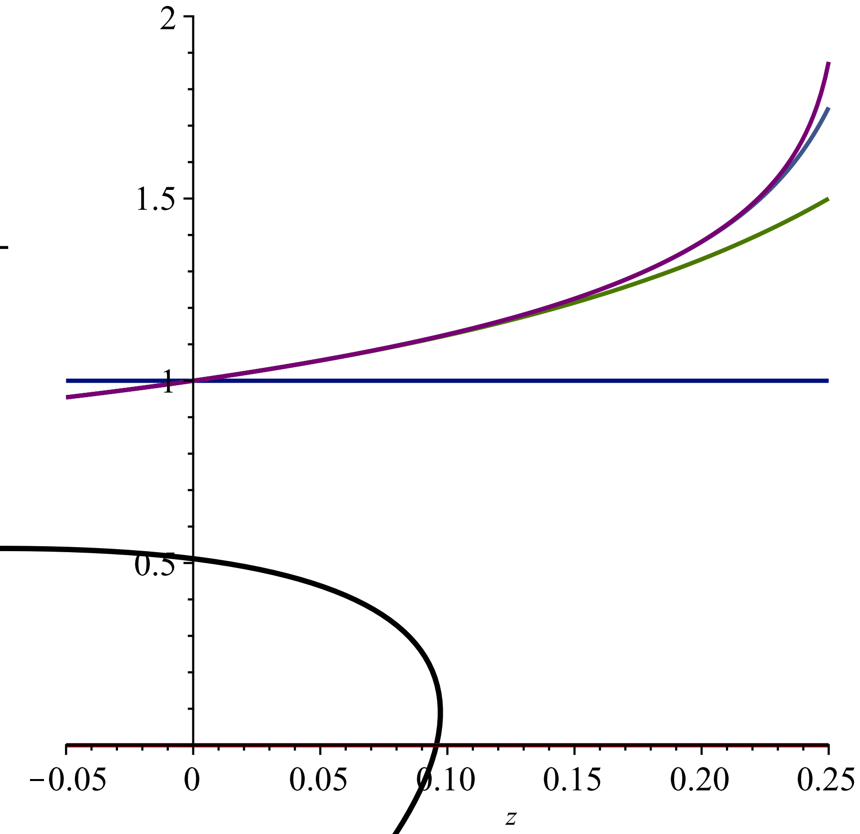
Proving numerical convergence requires control over the tails

On power series: $\quad y - y^{[\infty]} = O(z^m) \Longrightarrow \mathcal{N}(y) - y^{[\infty]} = O(z^{2m(+1)})$

Newton Iteration for Power Series

Same Newton Iteration

$$y^{[n+1]} = \mathcal{N}(y^{[n]}) = y^{[n]} + \frac{1 + zy^{[n]^2} - y^{[n]}}{1 - 2zy^{[n]}}$$



Proving numerical convergence requires control over the tails

$$y^{[0]} = 0$$

$$y^{[1]} = 1$$

$$y^{[2]} = 1 + z + 2z^2$$

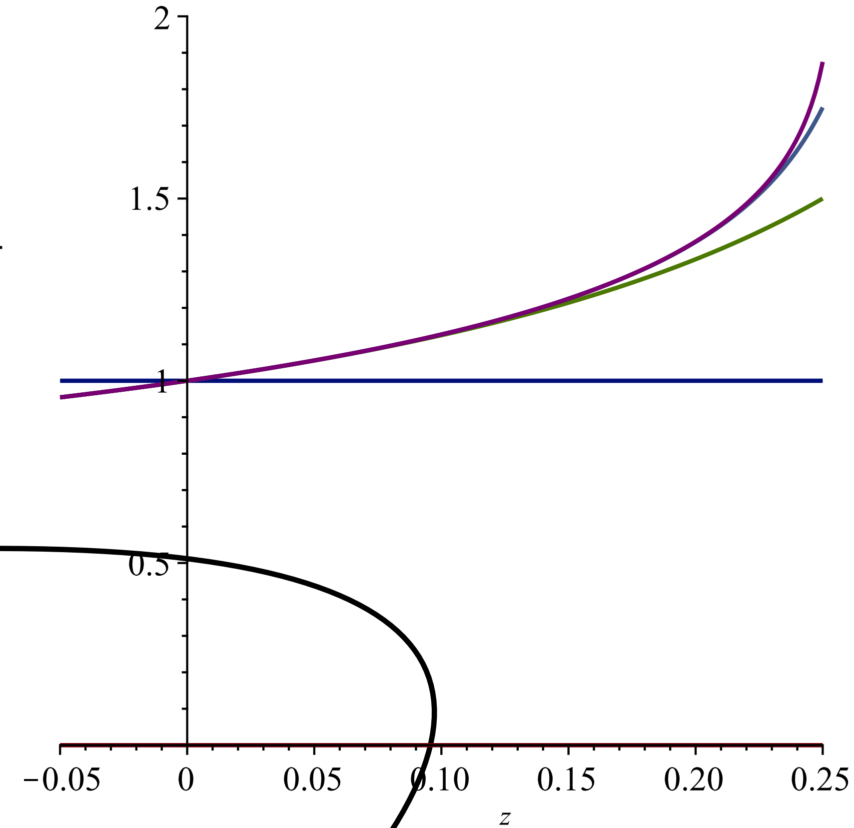
$$y^{[3]} = 1 + z + 2z^2 + 5z^3 + 14z^4 + 42z^5 + 132z^6$$

On power series: $y - y^{[\infty]} = O(z^m) \Longrightarrow \mathcal{N}(y) - y^{[\infty]} = O(z^{2m(+1)})$

```
Expand(N) = {
  res=Expand(N/2);
  a=ϕ(res); b=ϕ'(res);
  u=Solve(a+bx,x);
  return res+u; }
```

Newton Iteration for Power Series

Same Newton Iteration $y^{[n+1]} = \mathcal{N}(y^{[n]}) = y^{[n]} + \dfrac{1 + z{y^{[n]}}^2 - y^{[n]}}{1 - 2zy^{[n]}}$



Proving numerical convergence requires control over the tails

$y^{[0]} = 0$

$y^{[1]} = 1$

$y^{[2]} = 1 + z + 2z^2$

$y^{[3]} = 1 + z + 2z^2 + 5z^3 + 14z^4 + 42z^5 + 132z^6$

On power series: $y - y^{[\infty]} = O(z^m) \Longrightarrow \mathcal{N}(y) - y^{[\infty]} = O(z^{2m(+1)})$

```
Expand(N) = {
   res=Expand(N/2);
   a=ϕ(res); b=ϕ'(res);
   u=Solve(a+bx,x);
   return res+u; }
```

Cost(N) ≤ ct × Cost(last step)

Fast Enumeration

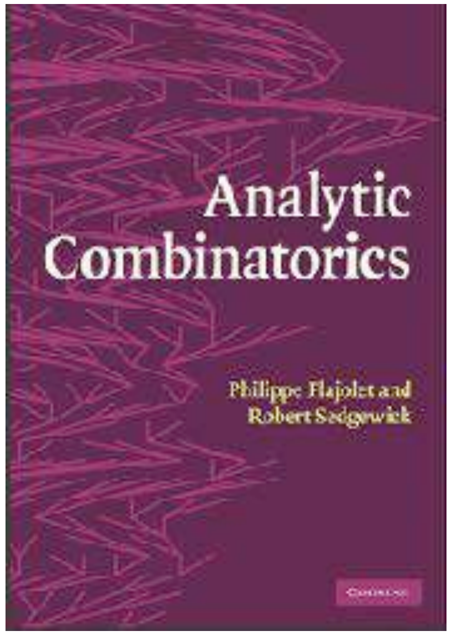
Thm. First N coefficients of GFs of all *constructible* structures:

1. arithmetic complexity $O(N \log N)$ (both ogf & egf);

2. bit complexity

$O(N^2 \log^2 N \log\log N)$ (ogf); $O(N^2 \log^3 N \log\log N)$ (egf).

Ingredients: Newton iteration & FFT.



[Pivoteau-S-Soria 2012]

Fast Enumeration

Thm. First N coefficients of GFs of all *constructible* structures:

1. arithmetic complexity $O(N \log N)$ (both ogf & egf);

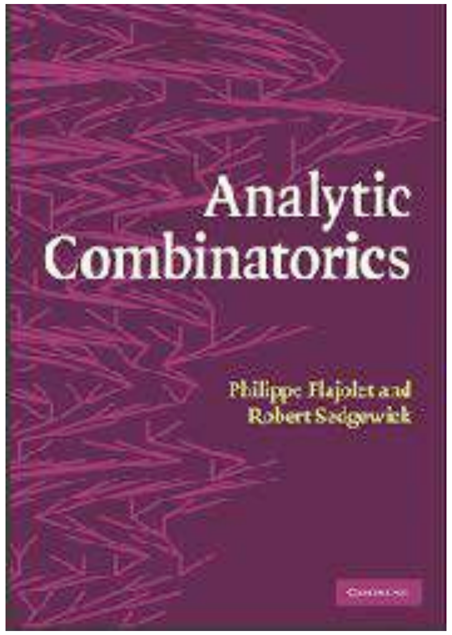
2. bit complexity

$O(N^2 \log^2 N \log\log N)$ (ogf); $O(N^2 \log^3 N \log\log N)$ (egf).

Ingredients: Newton iteration & FFT.

```
> with(NewtonGF):
> BinTrees:={B=Union(Epsilon,Prod(Z,B,B))}:
> GFSeries(BinTrees,labelled,z,21)[2];
```

$$B = 1 + z + 2z^2 + 5z^3 + 14z^4 + 42z^5 + 132z^6 + 429z^7 + 1430z^8 + 4862z^9 + 16796z^{10} + 58786z^{11} + 208012z^{12} + 742900z^{13} + 2674440z^{14} + 9694845z^{15} + 35357670z^{16} + 129644790z^{17} + 477638700z^{18} + 1767263190z^{19} + 6564120420z^{20} + O(z^{21})$$

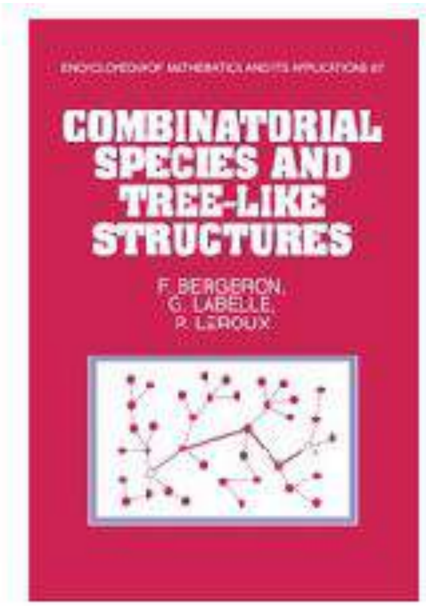


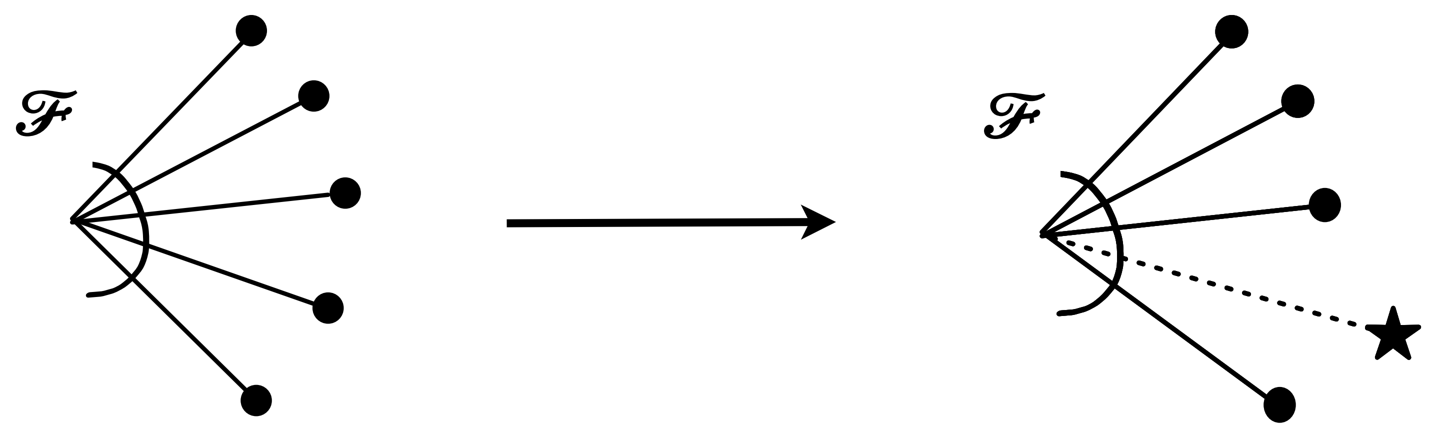
Demo
NewtonGF

[Pivoteau-S-Soria 2012]

Derivative of Combinatorial Structures

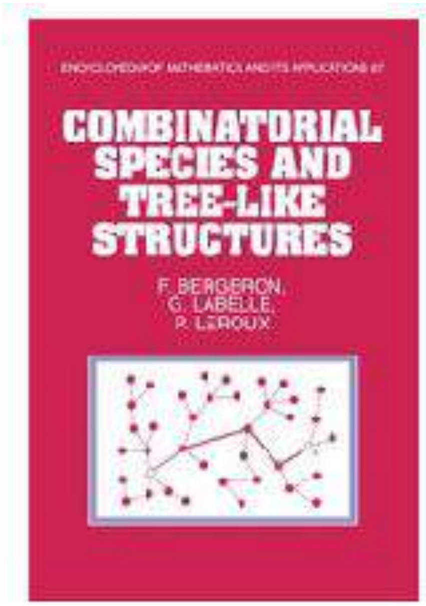
(origin: species theory)

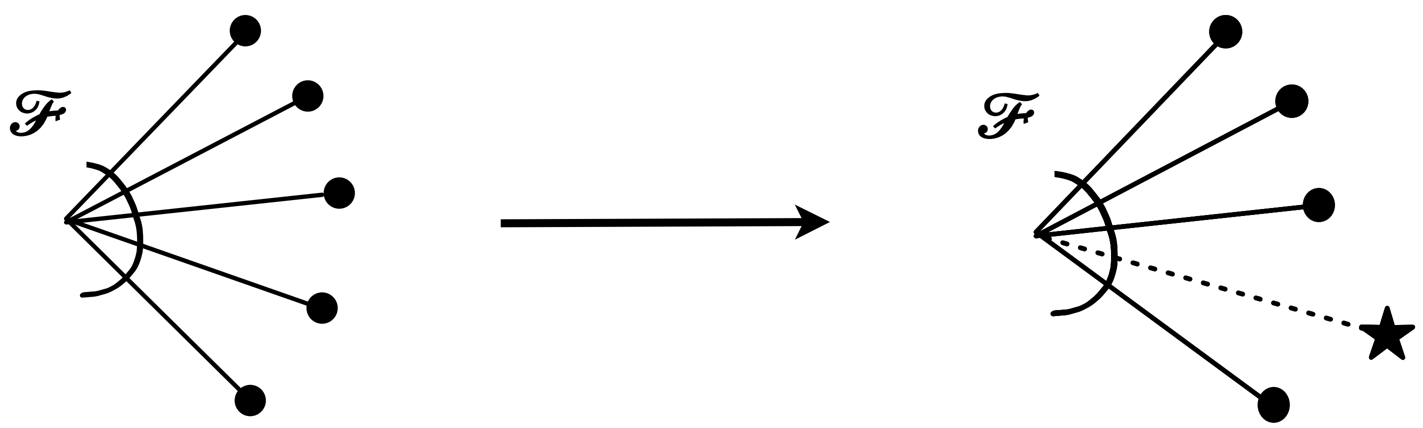




Derivative of Combinatorial Structures

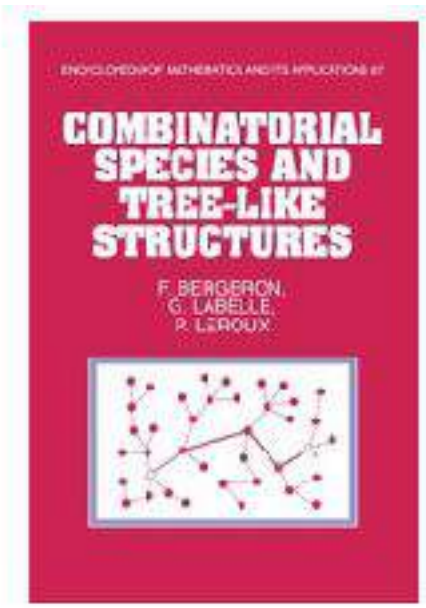
(origin: species theory)



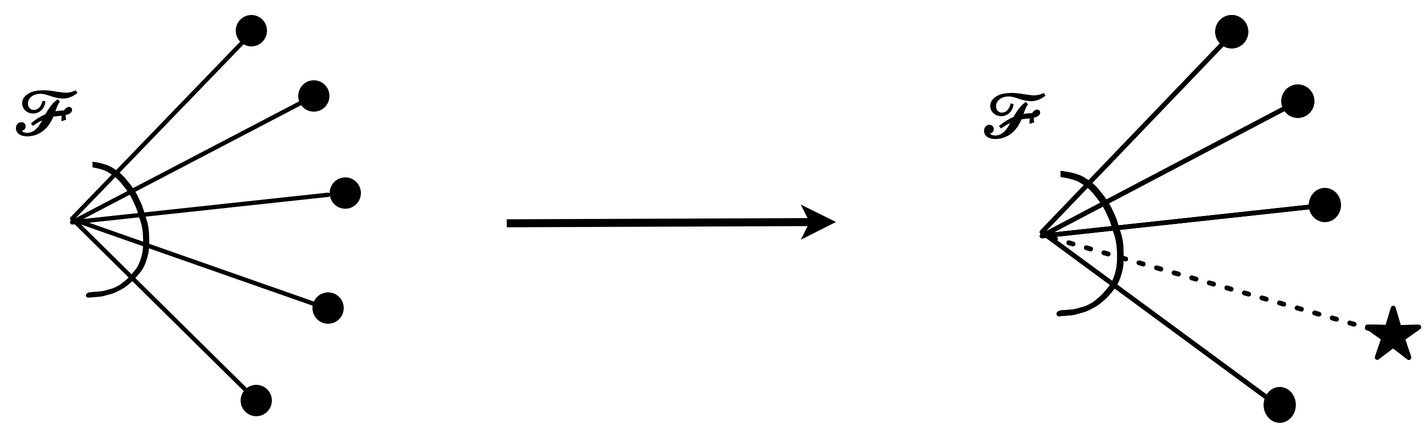


- $(\mathscr{F}+\mathscr{G})'=\mathscr{F}'+\mathscr{G}'$; $(\mathscr{F}\times\mathscr{G})'=\mathscr{F}'\times\mathscr{G}+\mathscr{F}\times\mathscr{G}'$;

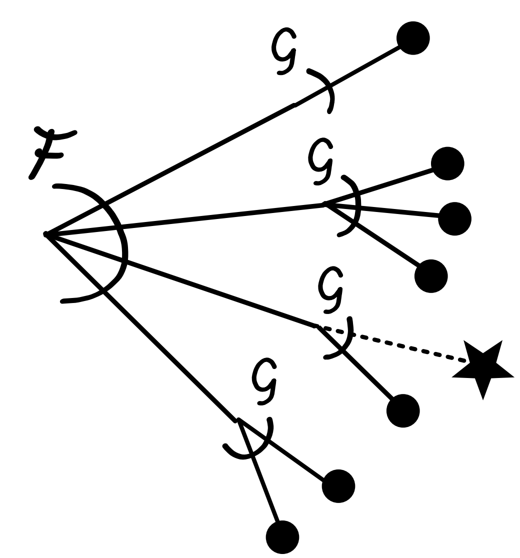
Derivative of Combinatorial Structures



(origin: species theory)

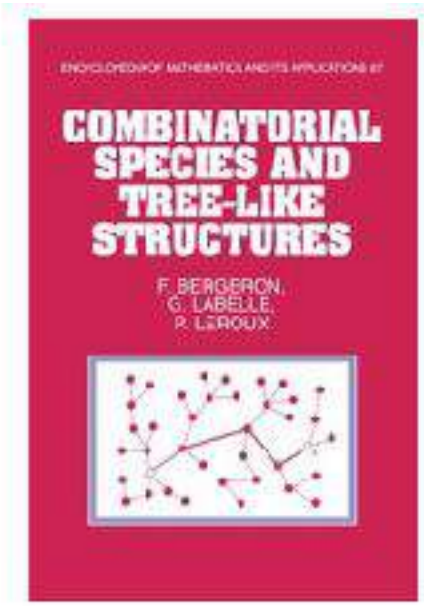


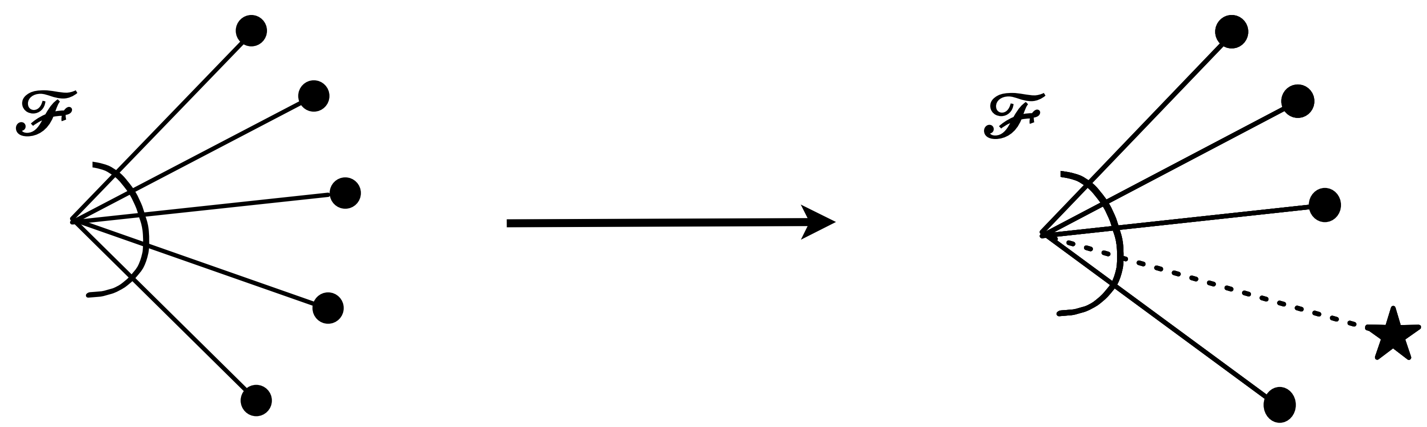
- $(\mathscr{F}+\mathscr{G})'=\mathscr{F}'+\mathscr{G}'$;$(\mathscr{F}\times\mathscr{G})'=\mathscr{F}'\times\mathscr{G}+\mathscr{F}\times\mathscr{G}'$;
- $\mathscr{F}(\mathscr{G})'=\mathscr{F}'(\mathscr{G})\times\mathscr{G}'$;



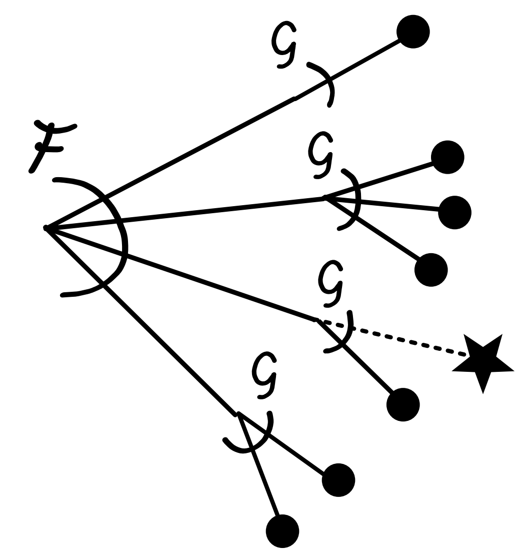
Derivative of Combinatorial Structures

(origin: species theory)



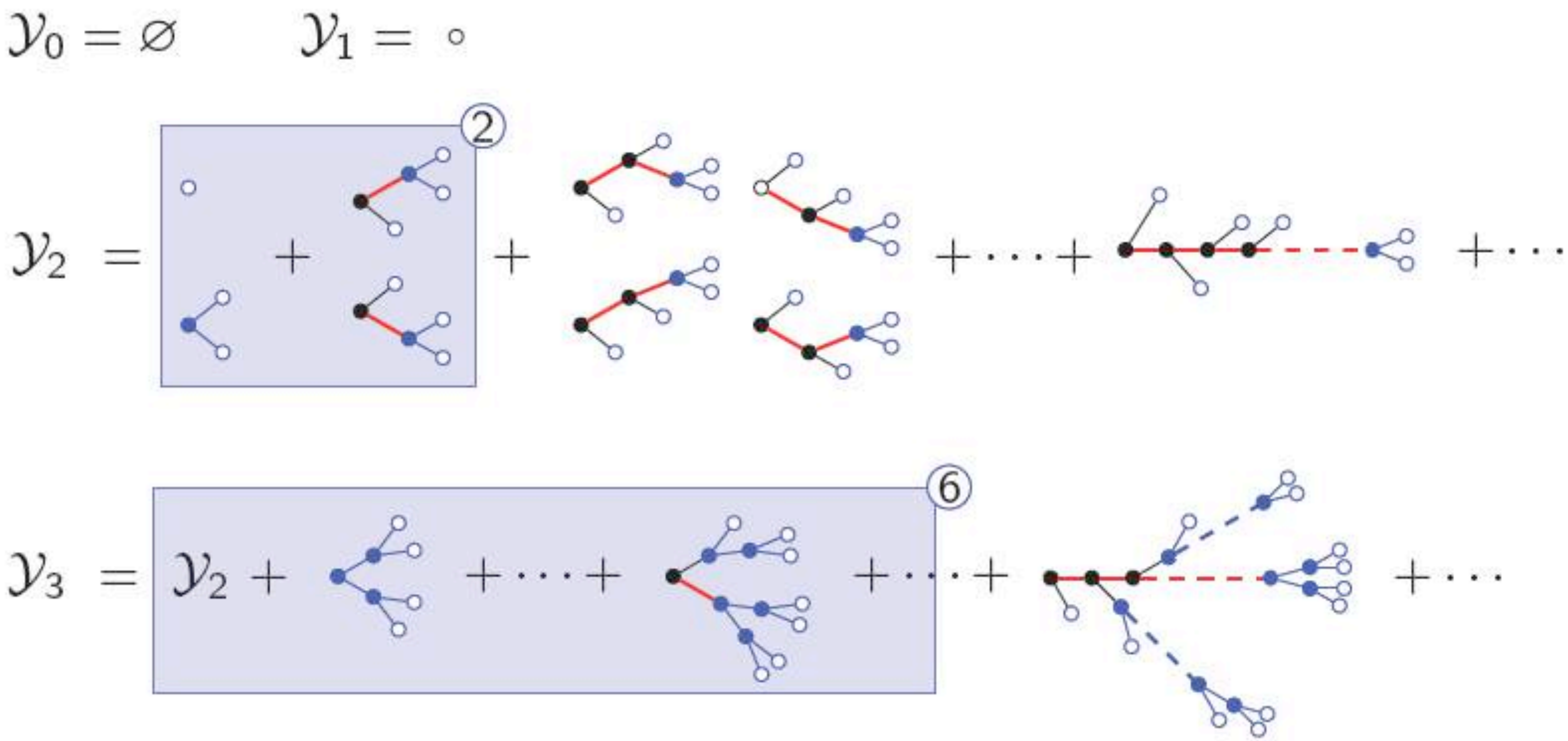


- $(\mathscr{F}+\mathscr{G})'=\mathscr{F}'+\mathscr{G}'$; $(\mathscr{F}\times\mathscr{G})'=\mathscr{F}'\times\mathscr{G}+\mathscr{F}\times\mathscr{G}'$;
- $\mathscr{F}(\mathscr{G})'=\mathscr{F}'(\mathscr{G})\times\mathscr{G}'$;
- $0'=1'=0$; $\mathscr{Z}'=1$;
  SET′=SET; CYC′=SEQ; SEQ′=SEQ×SEQ.



Combinatorial Newton Iteration

$$\mathcal{Y} = 1 + \mathcal{Z} \times \mathcal{Y} \times \mathcal{Y} =: \mathcal{H}(\mathcal{Z}, \mathcal{Y})$$



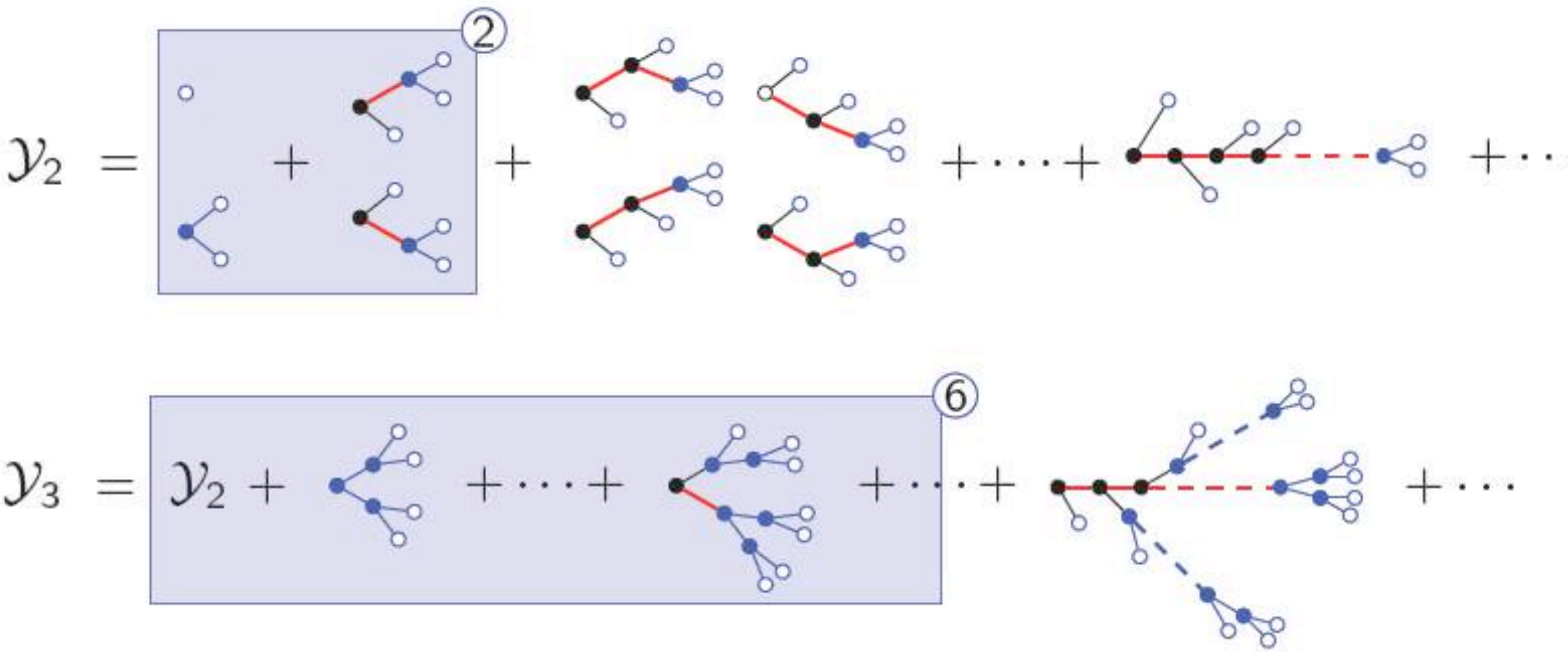
[Décoste, Labelle, Leroux 1982]

Combinatorial Newton Iteration

$$\mathcal{Y} = 1 + \mathcal{Z} \times \mathcal{Y} \times \mathcal{Y} =: \mathcal{H}(\mathcal{Z}, \mathcal{Y})$$

$$\mathcal{Y}^{[n+1]} = \mathcal{Y}^{[n]} + \textsc{Seq}(\mathcal{Z} \cdot \mathcal{Y}^{[n]} \cdot \star + \mathcal{Z} \cdot \star \cdot \mathcal{Y}^{[n]}) \cdot ((1 + \mathcal{Z} \cdot \mathcal{Y}^{[n]} \cdot \mathcal{Y}^{[n]}) \setminus \mathcal{Y}^{[n]}).$$

$\mathcal{Y}_0 = \varnothing \qquad \mathcal{Y}_1 = \circ$



[Décoste, Labelle, Leroux 1982]

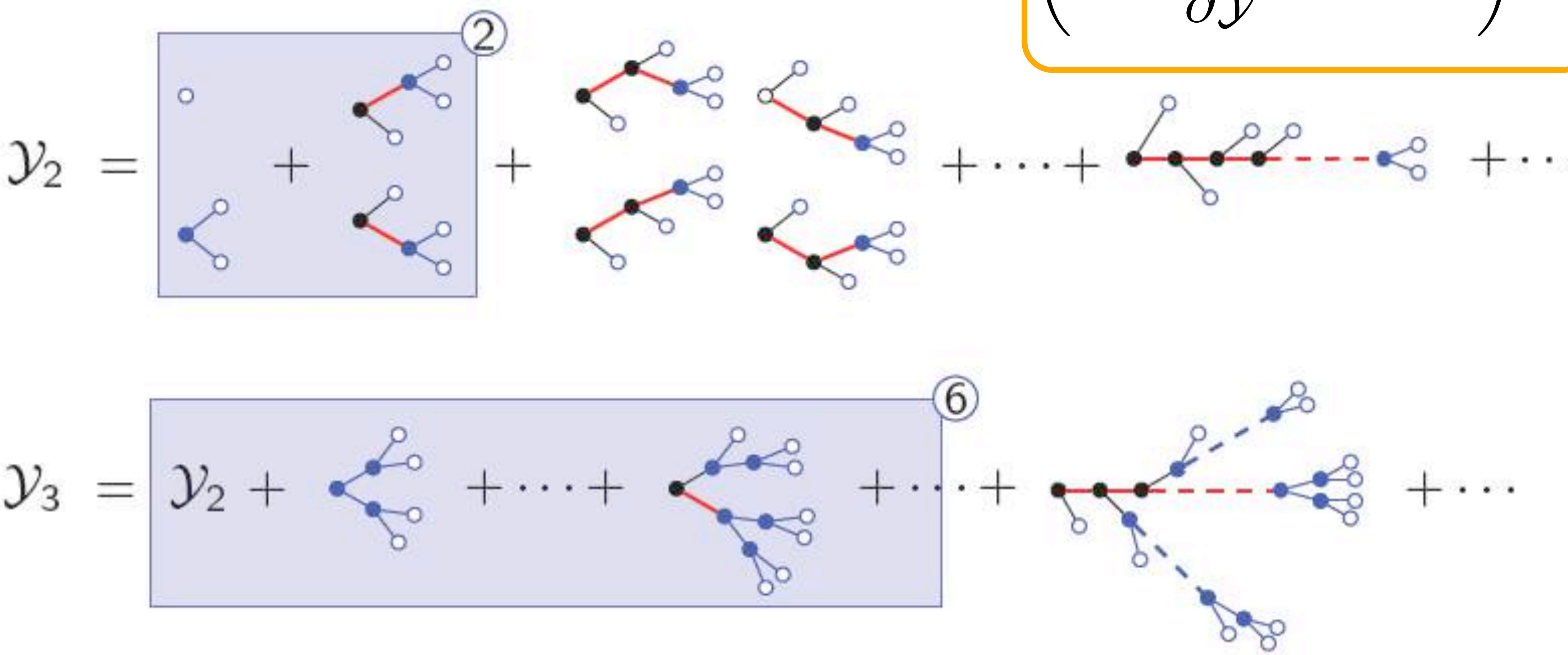
Combinatorial Newton Iteration

$$\mathcal{Y} = 1 + \mathcal{Z} \times \mathcal{Y} \times \mathcal{Y} =: \mathcal{H}(\mathcal{Z}, \mathcal{Y})$$

$$\mathcal{Y}^{[n+1]} = \mathcal{Y}^{[n]} + \textsc{Seq}(\mathcal{Z} \cdot \mathcal{Y}^{[n]} \cdot \star + \mathcal{Z} \cdot \star \cdot \mathcal{Y}^{[n]}) \cdot ((1 + \mathcal{Z} \cdot \mathcal{Y}^{[n]} \cdot \mathcal{Y}^{[n]}) \setminus \mathcal{Y}^{[n]}).$$

$$\left(\mathrm{Id} - \frac{\partial \mathcal{H}}{\partial \mathcal{Y}}(\mathcal{Z}, \mathcal{Y}^{[n]})\right)^{-1}$$

$$\mathcal{Y}_0 = \varnothing \qquad \mathcal{Y}_1 = \circ$$



[Décoste, Labelle, Leroux 1982]

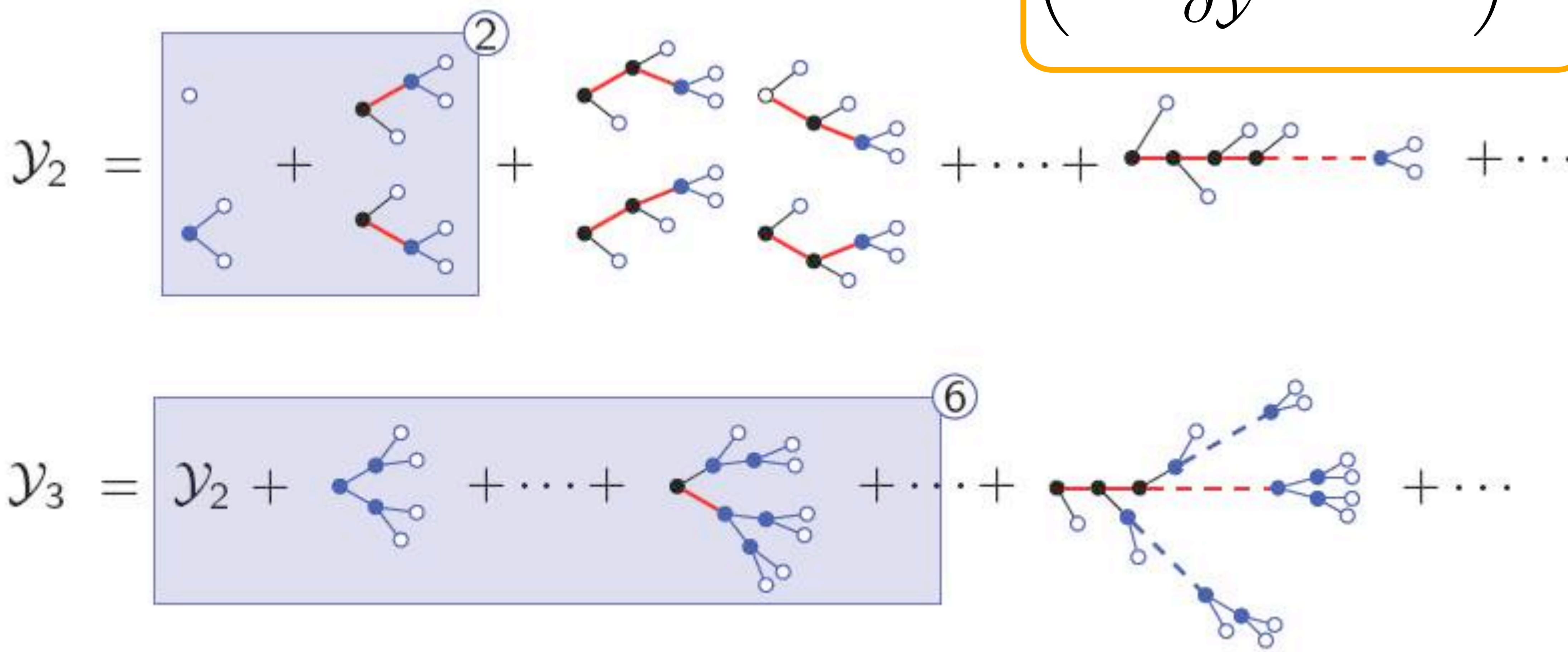
Combinatorial Newton Iteration

$$\mathcal{Y} = 1 + \mathcal{Z} \times \mathcal{Y} \times \mathcal{Y} =: \mathcal{H}(\mathcal{Z}, \mathcal{Y})$$

$$\mathcal{Y}^{[n+1]} = \mathcal{Y}^{[n]} + \boxed{\textsc{Seq}(\mathcal{Z} \cdot \mathcal{Y}^{[n]} \cdot \star + \mathcal{Z} \cdot \star \cdot \mathcal{Y}^{[n]})} \cdot ((1 + \mathcal{Z} \cdot \mathcal{Y}^{[n]} \cdot \mathcal{Y}^{[n]}) \setminus \mathcal{Y}^{[n]}).$$

$$\boxed{\left(\mathrm{Id} - \frac{\partial \mathcal{H}}{\partial \mathcal{Y}}(\mathcal{Z}, \mathcal{Y}^{[n]})\right)^{-1}}$$

$\mathcal{Y}_0 = \varnothing \qquad \mathcal{Y}_1 = \circ$

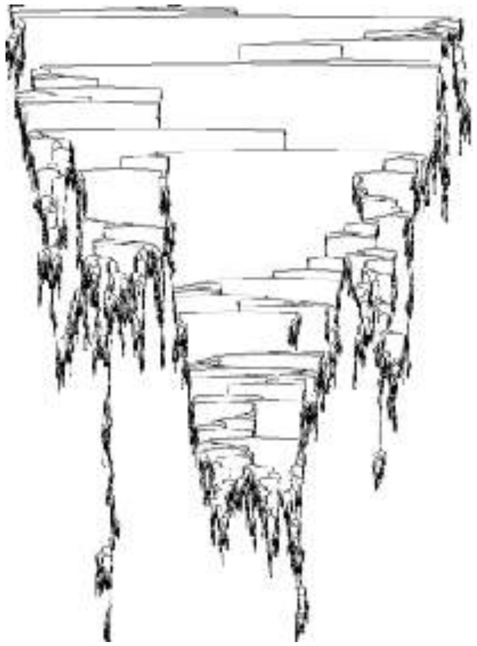


Ccl: numerical convergence of the Newton iteration starting from 0.

[Décoste, Labelle, Leroux 1982]

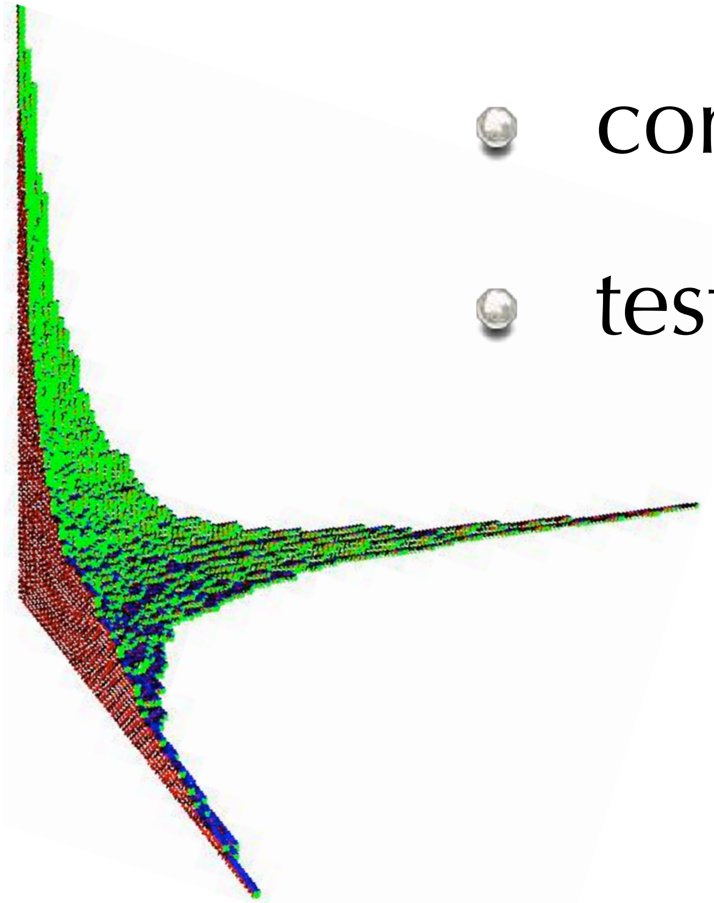
III. Random Generation

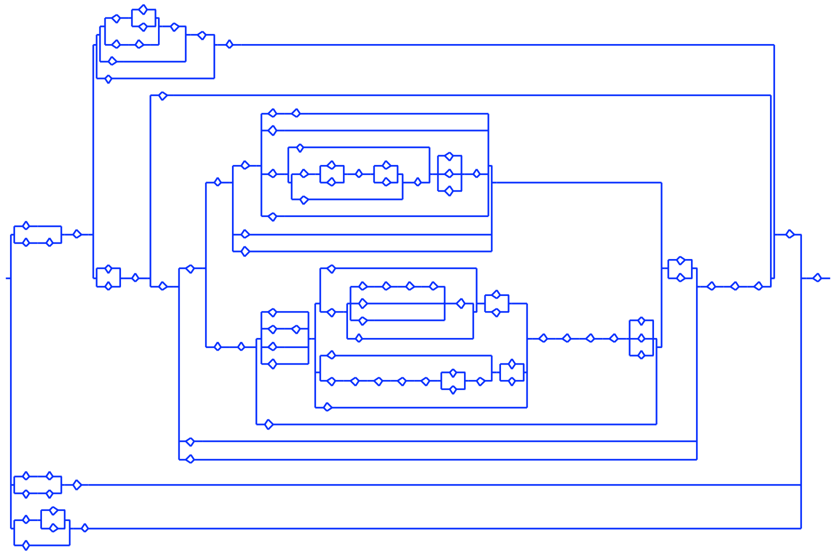
Why Random Generation?



Simulation in the discrete world; helps

- evaluate parameters;
- compare/refine models;
- test software.





Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

x is a parameter of the algorithm

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size:

$$\frac{xT'(x)}{T(x)}$$

x is a parameter of the algorithm

Boltzmann Sampling

$$\text{Proba}(t) = \frac{x^{|t|}}{T(x)} \quad \text{with} \quad T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈ℋ) = {
   if ℋ=1 return 1;
   if ℋ=𝒵 return 𝒵;
   if ℋ=ℱxℊ {
       Generate(f∈ℱ);
       Generate(g∈𝒢);
       return (f,g); }
   if ℋ=ℱ+𝒢 {
       b:=Bernoulli(F(x)/H(x));
       if b=1 return Generate(f∈ℱ);
       else return Generate(g∈𝒢);
   if ℋ=Set(ℱ) {
       …
```

[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

$$\text{Proba}(t) = \frac{x^{|t|}}{T(x)} \quad \text{with} \quad T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$$

Expected size:

$$\frac{xT'(x)}{T(x)}$$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝓩 return 𝓩;
   if 𝓗=𝓕x𝓖 {
       Generate(f∈𝓕);
       Generate(g∈𝓖);
       return (f,g); }
   if 𝓗=𝓕+𝓖 {
       b:=Bernoulli(F(x)/H(x));
       if b=1 return Generate(f∈𝓕);
       else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
       …
```

F(x),*H(x)* by Newton iteration

[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

$$\text{Proba}(t) = \frac{x^{|t|}}{T(x)} \quad \text{with} \quad T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝒽) = {
   if 𝓗=1 return 1;
   if 𝓗=𝒵 return 𝒵;
   if 𝓗=𝓕x𝓖 {
       Generate(f∈𝓕);
       Generate(g∈𝓖);
       return (f,g); }
   if 𝓗=𝓕+𝓖 {
       b:=Bernoulli(F(x)/H(x));
       if b=1 return Generate(f∈𝓕);
       else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
       ...
```

F(x),H(x) by Newton iteration

Example: binary trees

$$\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…

[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x)=\sum_{t\in\mathcal{T}} x^{|t|}=\sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈ℋ) = {
   if ℋ=1 return 1;
   if ℋ=𝒵 return 𝒵;
   if ℋ=ℱx𝒢 {
      Generate(f∈ℱ);
      Generate(g∈𝒢);
      return (f,g); }
   if ℋ=ℱ+𝒢 {
      b:=Bernoulli(F(x)/H(x));
      if b=1 return Generate(f∈ℱ);
      else return Generate(g∈𝒢);
   if ℋ=Set(ℱ) {
      …
```

F(x),H(x) by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…

[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈ℋ) = {
   if ℋ=1 return 1;
   if ℋ=𝒵 return 𝒵;
   if ℋ=ℱx𝒢 {
      Generate(f∈ℱ);
      Generate(g∈𝒢);
      return (f,g); }
   if ℋ=ℱ+𝒢 {
      b:=Bernoulli(F(x)/H(x));
      if b=1 return Generate(f∈ℱ);
      else return Generate(g∈𝒢);
   if ℋ=Set(ℱ) {
      …
```

F(x),*H(x)* by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…

[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t\in\mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈ℋ) = {
   if ℋ=1 return 1;
   if ℋ=𝒵 return 𝒵;
   if ℋ=ℱxG {
      Generate(f∈ℱ);
      Generate(g∈𝒢);
      return (f,g); }
   if ℋ=ℱ+𝒢 {
      b:=Bernoulli(F(x)/H(x));
      if b=1 return Generate(f∈ℱ);
      else return Generate(g∈𝒢);
   if ℋ=Set(ℱ) {
      …
```

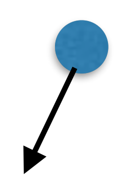
F(x),H(x) by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝓩 return 𝓩;
   if 𝓗=𝓕x𝓖 {
       Generate(f∈𝓕);
       Generate(g∈𝓖);
       return (f,g); }
   if 𝓗=𝓕+𝓖 {
       b:=Bernoulli(F(x)/H(x));
       if b=1 return Generate(f∈𝓕);
       else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
       …
```

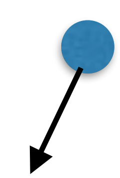
F(x),H(x) by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈ℋ) = {
   if ℋ=1 return 1;
   if ℋ=𝒵 return 𝒵;
   if ℋ=ℱxℊ {
      Generate(f∈ℱ);
      Generate(g∈ℊ);
      return (f,g); }
   if ℋ=ℱ+ℊ {
      b:=Bernoulli(F(x)/H(x));
      if b=1 return Generate(f∈ℱ);
      else return Generate(g∈ℊ);
   if ℋ=Set(ℱ) {
      …
```

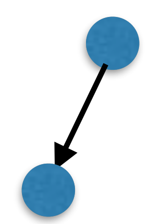
F(x),H(x) by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t\in\mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈ℋ) = {
   if ℋ=1 return 1;
   if ℋ=𝒵 return 𝒵;
   if ℋ=ℱx𝒢 {
      Generate(f∈ℱ);
      Generate(g∈𝒢);
      return (f,g); }
   if ℋ=ℱ+𝒢 {
      b:=Bernoulli(F(x)/H(x));
      if b=1 return Generate(f∈ℱ);
      else return Generate(g∈𝒢);
   if ℋ=Set(ℱ) {
      …
```

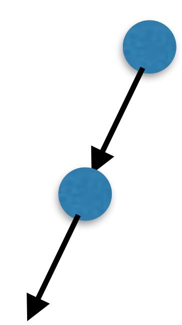
F(x),H(x) by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
  if 𝓗=1 return 1;
  if 𝓗=𝓩 return 𝓩;
  if 𝓗=𝓕x𝓖 {
     Generate(f∈𝓕);
     Generate(g∈𝓖);
     return (f,g); }
  if 𝓗=𝓕+𝓖 {
     b:=Bernoulli(F(x)/H(x));
     if b=1 return Generate(f∈𝓕);
     else return Generate(g∈𝓖);
  if 𝓗=Set(𝓕) {
     …
```

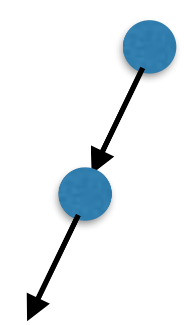
F(x),*H(x)* by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝓩 return 𝓩;
   if 𝓗=𝓕x𝓖 {
      Generate(f∈𝓕);
      Generate(g∈𝓖);
      return (f,g); }
   if 𝓗=𝓕+𝓖 {
      b:=Bernoulli(F(x)/H(x));
      if b=1 return Generate(f∈𝓕);
      else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
      …
```

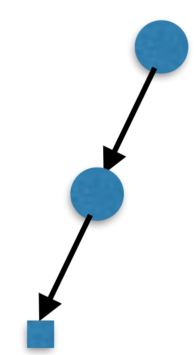
F(x),*H(x)* by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝓩 return 𝓩;
   if 𝓗=𝓕x𝓖 {
       Generate(f∈𝓕);
       Generate(g∈𝓖);
       return (f,g); }
   if 𝓗=𝓕+𝓖 {
       b:=Bernoulli(F(x)/H(x));
       if b=1 return Generate(f∈𝓕);
       else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
       …
```

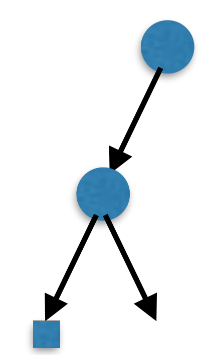
F(x),*H(x)* by Newton iteration

Example: binary trees

$\mathcal{B}$=1+$\mathcal{Z}$x$\mathcal{B}$x$\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

$$\text{Proba}(t) = \frac{x^{|t|}}{T(x)} \quad \text{with} \quad T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝓩 return 𝓩;
   if 𝓗=𝓕x𝓖 {
       Generate(f∈𝓕);
       Generate(g∈𝓖);
       return (f,g); }
   if 𝓗=𝓕+𝓖 {
       b:=Bernoulli(F(x)/H(x));
       if b=1 return Generate(f∈𝓕);
       else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
       …
```

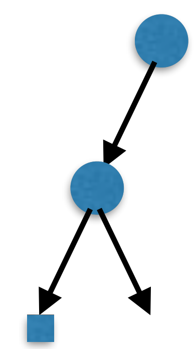
F(x),H(x) by Newton iteration

Example: binary trees

$\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t\in\mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝓩 return 𝓩;
   if 𝓗=𝓕x𝓖 {
       Generate(f∈𝓕);
       Generate(g∈𝓖);
       return (f,g); }
   if 𝓗=𝓕+𝓖 {
       b:=Bernoulli(F(x)/H(x));
       if b=1 return Generate(f∈𝓕);
       else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
       …
```

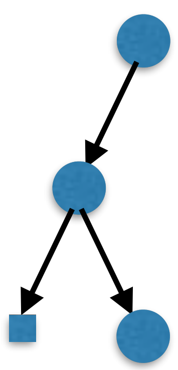
F(x),*H(x)* by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

$$\text{Proba}(t) = \frac{x^{|t|}}{T(x)} \quad \text{with} \quad T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$$

x is a parameter of the algorithm

Expected size: $\dfrac{xT'(x)}{T(x)}$

```
Generate(h∈ℋ) = {
   if ℋ=1 return 1;
   if ℋ=𝒵 return 𝒵;
   if ℋ=ℱxℊ {
      Generate(f∈ℱ);
      Generate(g∈ℊ);
      return (f,g); }
   if ℋ=ℱ+ℊ {
      b:=Bernoulli(F(x)/H(x));
      if b=1 return Generate(f∈ℱ);
      else return Generate(g∈ℊ);
   if ℋ=Set(ℱ) {
      …
```

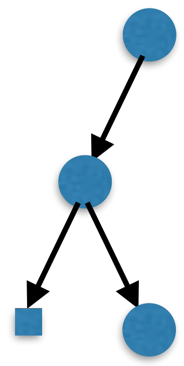
F(x),H(x) by Newton iteration

Example: binary trees

$$\mathcal{B} = 1 + \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝓩 return 𝓩;
   if 𝓗=𝓕x𝓖 {
      Generate(f∈𝓕);
      Generate(g∈𝓖);
      return (f,g); }
   if 𝓗=𝓕+𝓖 {
      b:=Bernoulli(F(x)/H(x));
      if b=1 return Generate(f∈𝓕);
      else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
      …
```

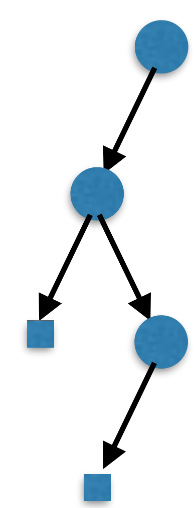
F(x),H(x) by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(*t*)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝓩 return 𝓩;
   if 𝓗=𝓕x𝓖 {
       Generate(f∈𝓕);
       Generate(g∈𝓖);
       return (f,g); }
   if 𝓗=𝓕+𝓖 {
       b:=Bernoulli(F(x)/H(x));
       if b=1 return Generate(f∈𝓕);
       else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
       …
```

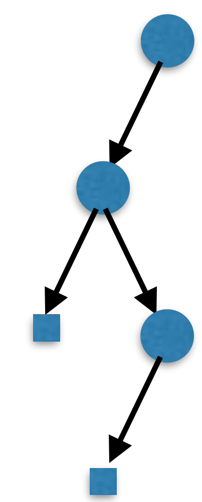
F(x),H(x) by Newton iteration

Example: binary trees

$\mathcal{B}$=1+$\mathcal{Z}$x$\mathcal{B}$x$\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝓩 return 𝓩;
   if 𝓗=𝓕x𝓖 {
       Generate(f∈𝓕);
       Generate(g∈𝓖);
       return (f,g); }
   if 𝓗=𝓕+𝓖 {
       b:=Bernoulli(F(x)/H(x));
       if b=1 return Generate(f∈𝓕);
       else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
       …
```

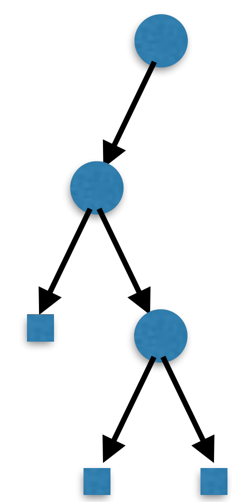
F(x),*H(x)* by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t \in \mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
  if 𝓗=1 return 1;
  if 𝓗=𝓩 return 𝓩;
  if 𝓗=𝓕x𝓖 {
     Generate(f∈𝓕);
     Generate(g∈𝓖);
     return (f,g); }
  if 𝓗=𝓕+𝓖 {
     b:=Bernoulli(F(x)/H(x));
     if b=1 return Generate(f∈𝓕);
     else return Generate(g∈𝓖);
  if 𝓗=Set(𝓕) {
     …
```

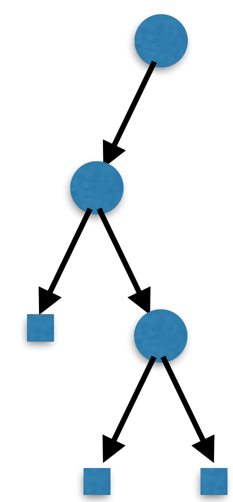
F(x),*H(x)* by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t\in\mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝒵 return 𝒵;
   if 𝓗=𝓕x𝓖 {
       Generate(f∈𝓕);
       Generate(g∈𝓖);
       return (f,g); }
   if 𝓗=𝓕+𝓖 {
       b:=Bernoulli(F(x)/H(x));
       if b=1 return Generate(f∈𝓕);
       else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
       …
```

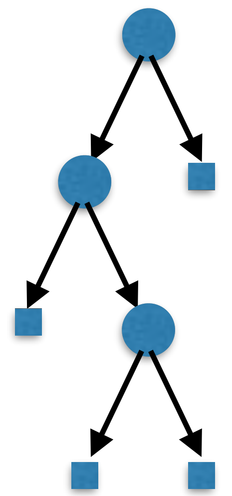
F(x),H(x) by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Boltzmann Sampling

Proba(t)= $\dfrac{x^{|t|}}{T(x)}$ with $T(x) = \sum_{t\in\mathcal{T}} x^{|t|} = \sum_n T_n x^n$

Expected size: $\dfrac{xT'(x)}{T(x)}$

x is a parameter of the algorithm

```
Generate(h∈𝓗) = {
   if 𝓗=1 return 1;
   if 𝓗=𝓩 return 𝓩;
   if 𝓗=𝓕x𝓖 {
      Generate(f∈𝓕);
      Generate(g∈𝓖);
      return (f,g); }
   if 𝓗=𝓕+𝓖 {
      b:=Bernoulli(F(x)/H(x));
      if b=1 return Generate(f∈𝓕);
      else return Generate(g∈𝓖);
   if 𝓗=Set(𝓕) {
      …
```

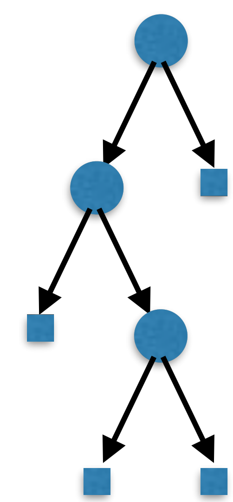
F(x),*H(x)* by Newton iteration

Example: binary trees

$\mathcal{B}=1+\mathcal{Z}\times\mathcal{B}\times\mathcal{B}$

with 1/B(.23)≃1/1.559≃.6414

Coin: 0,0,1,0,1,1,1,1,0,1,…



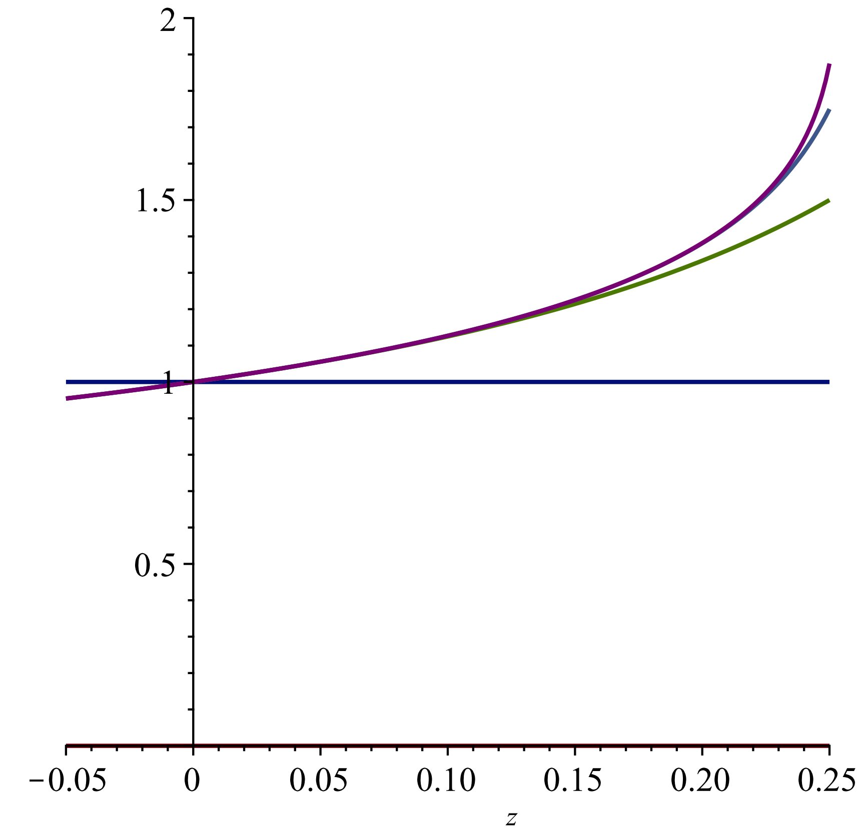
Exp. size

$\sim \dfrac{1}{2\sqrt{1-4x}}$

[Duchon, Flajolet, Louchard, Schaeffer 2002-04]

Example: XML-trees

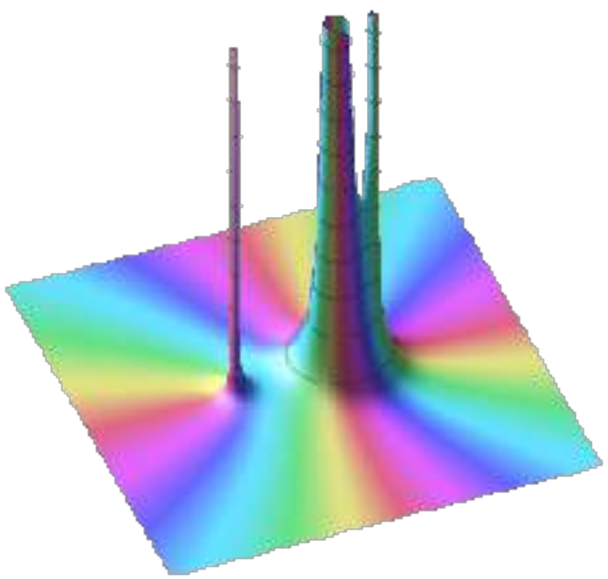
Grammar	File size	#eq.	Newton
rss	9.5k	16	0.02s.
Relax NG	124k	114	0.10s.
XSLT	168k	122	0.12s.
XHTML Basic	284k	96	0.32s.
SVG	6.3M	232	0.23s.
OpenDocument	2.8M	814	0.34s.
DocBook	11M	977	23s.



Time for *x* s.t. E(size)=10,000

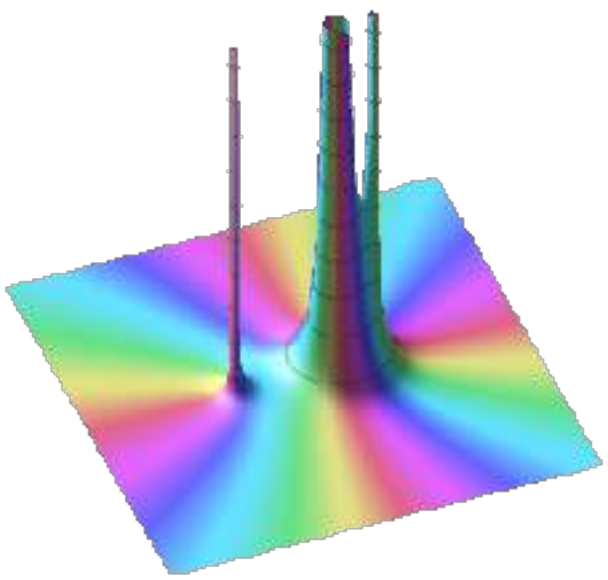
[Darrasse2010]

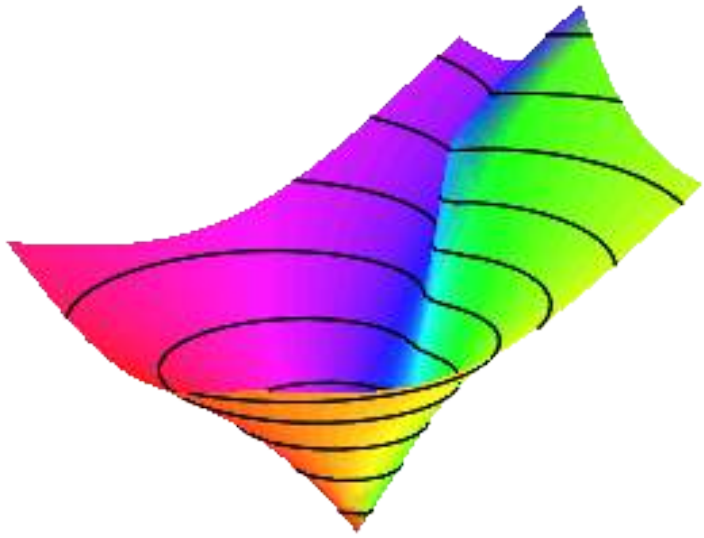
IV. Asymptotic Analysis



$$F(z) = \sum_{n=0}^{\infty} f_n z^n \longrightarrow f_n \sim \cdots, \quad n \to \infty$$

IV. Asymptotic Analysis





$$F(z) = \sum_{n=0}^{\infty} f_n z^n \longrightarrow f_n \sim \cdots, \quad n \to \infty$$

Singularity Analysis

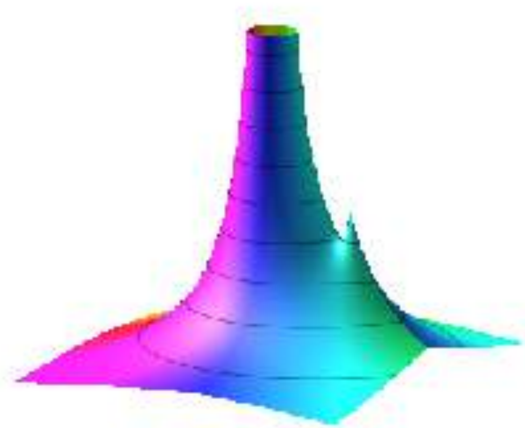
counts the number of objects of size *n*

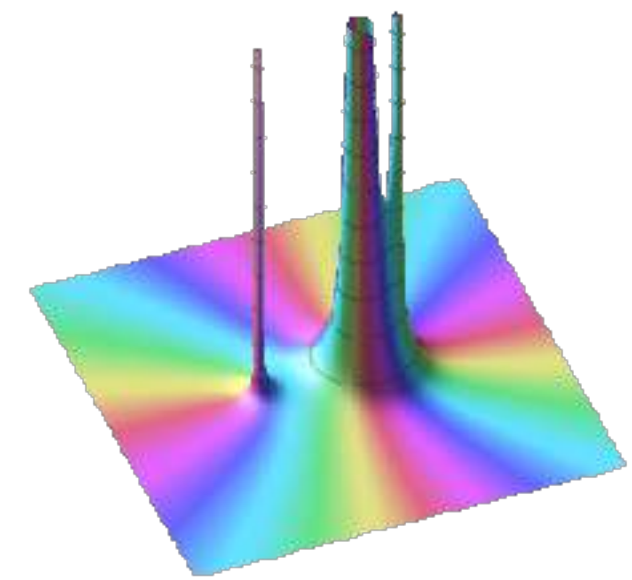
$$(a_n) \mapsto A(z) := \sum_{n \geq 0} a_n z^n$$

captures some structure

A 3-Step Method:

$$a_n = \frac{1}{2\pi i} \oint \frac{A(z)}{z^{n+1}} \, dz$$





Singularity Analysis

counts the number of objects of size n

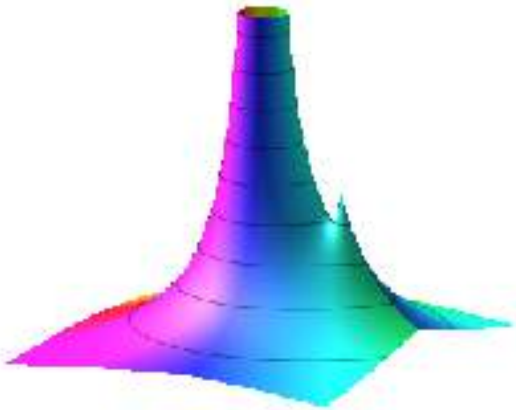
$$(a_n) \mapsto A(z) := \sum_{n \geq 0} a_n z^n$$

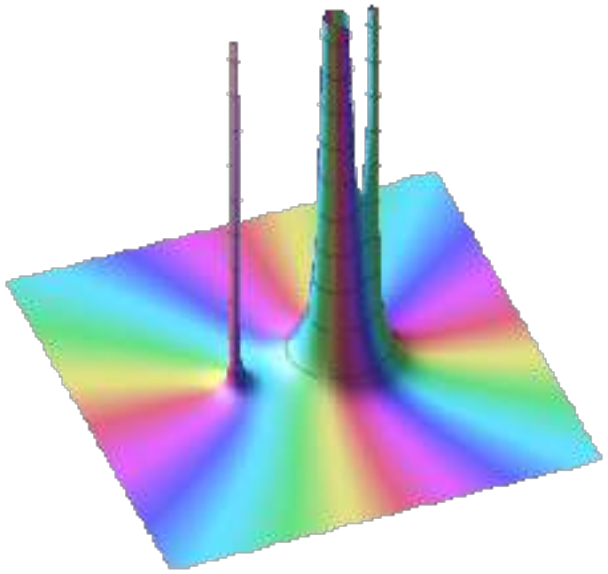
captures some structure

A 3-Step Method:

1. Locate dominant singularities
2. Compute local behaviour
3. Translate into asymptotics

$$a_n = \frac{1}{2\pi i} \oint \frac{A(z)}{z^{n+1}} \, dz$$





Singularity Analysis

counts the number of objects of size *n*

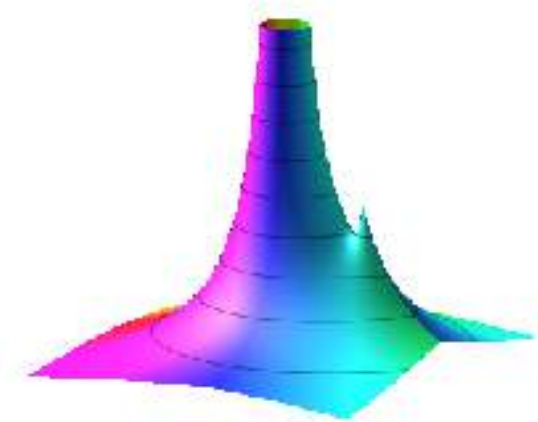
$$(a_n) \mapsto A(z) := \sum_{n\geq 0} a_n z^n$$

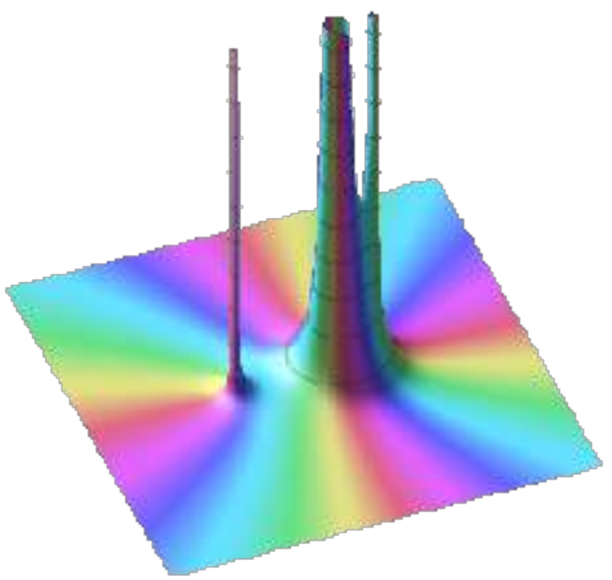
captures some structure

A 3-Step Method:

1. Locate dominant singularities
2. Compute local behaviour
3. Translate into asymptotics

$$a_n = \frac{1}{2\pi i}\oint \frac{A(z)}{z^{n+1}}\,dz$$





$$A(z) \underset{z\to\rho}{\sim} c\left(1-\frac{z}{\rho}\right)^{\alpha}\log^m\frac{1}{1-\frac{z}{\rho}}$$

$$a_n \underset{n\to\infty}{\sim} c\,\rho^{-n}\frac{n^{-\alpha-1}}{\Gamma(-\alpha)}\log^m n$$

$(\alpha \notin \mathbb{N})$

Singularity Analysis

counts the number of objects of size n

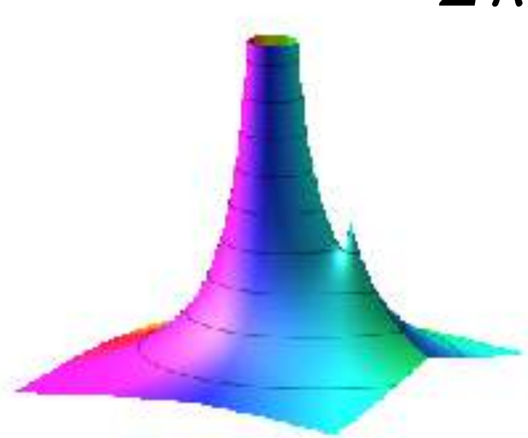
$$(a_n) \mapsto A(z) := \sum_{n \geq 0} a_n z^n$$

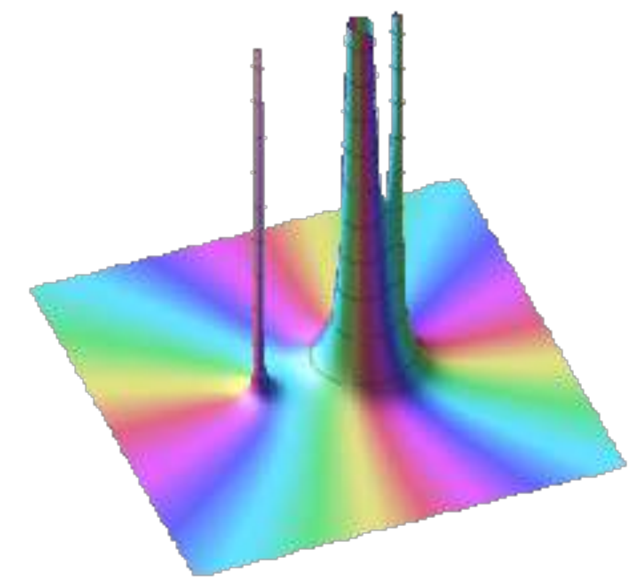
captures some structure

A 3-Step Method:

1. Locate dominant singularities
2. Compute local behaviour
3. Translate into asymptotics

$$a_n = \frac{1}{2\pi i} \oint \frac{A(z)}{z^{n+1}}\, dz$$



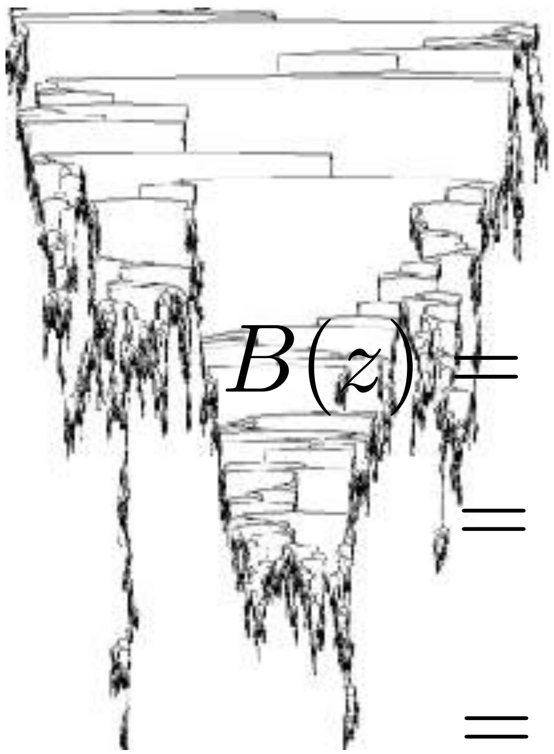


$$A(z) \underset{z \to \rho}{\sim} c \left(1 - \frac{z}{\rho}\right)^{\alpha} \log^{m} \frac{1}{1 - \frac{z}{\rho}} \qquad a_n \underset{n \to \infty}{\sim} c\, \rho^{-n} \frac{n^{-\alpha-1}}{\Gamma(-\alpha)} \log^{m} n$$

$(\alpha \notin \mathbb{N})$

full asymptotic expansion available

Example: Binary Trees (Again!)



$$B(z) = 1 + zB(z)^2$$
$$= 1 + z + 2z^2 + 5z^3 + \cdots$$
$$= \frac{1-\sqrt{1-4z}}{2z}$$

A 3-Step Method:

1. Locate dominant singularities;
2. Compute local behaviour;
3. Translate into asymptotics.

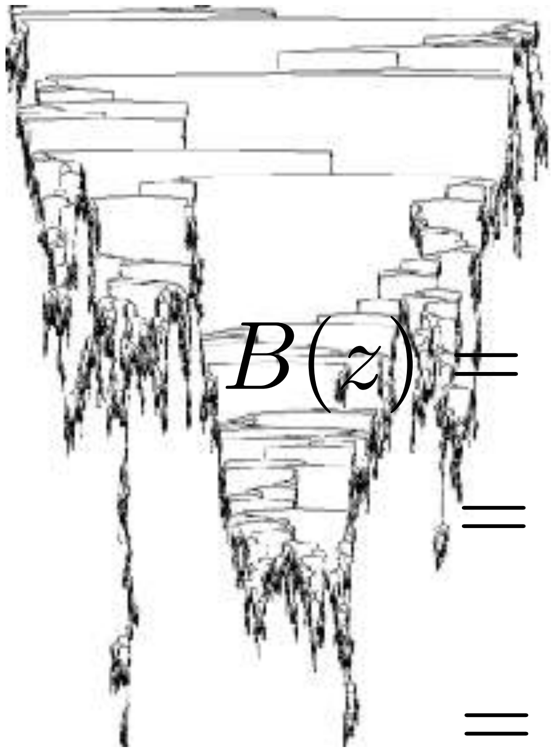
1. sing. en $z = 1/4$
2. local behaviour:

$$B(z) \underset{z \to 1/4}{=} 2 - 2(1-4z)^{1/2} + \cdots$$

3. translation:

$$B_n \sim \frac{4^n n^{-3/2}}{\sqrt{\pi}} \left(1 - \frac{9}{8n} + \cdots\right)$$

Example: Binary Trees (Again!)



$$B(z) = 1 + zB(z)^2$$
$$= 1 + z + 2z^2 + 5z^3 + \cdots$$
$$= \frac{1 - \sqrt{1-4z}}{2z}$$

A 3-Step Method:

1. Locate dominant singularities;
2. Compute local behaviour;
3. Translate into asymptotics.

1. sing. en $z = 1/4$
2. local behaviour:

$$B(z) \underset{z \to 1/4}{=} 2 - 2(1-4z)^{1/2} + \cdots$$

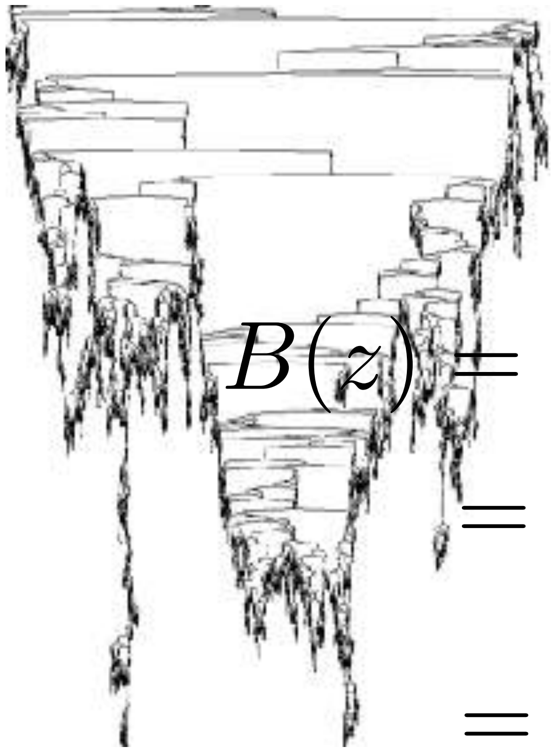
3. translation:

$$B_n \sim \frac{4^n n^{-3/2}}{\sqrt{\pi}} \left(1 - \frac{9}{8n} + \cdots \right)$$

Probability that a leaf is a child of the root?

$$\frac{[z^n]2zB(z)}{B_n} = \frac{1}{2} + \frac{3}{4n} + O\left(\frac{1}{n^2}\right)$$

Example: Binary Trees (Again!)



$$B(z) = 1 + zB(z)^2$$
$$= 1 + z + 2z^2 + 5z^3 + \cdots$$
$$= \frac{1-\sqrt{1-4z}}{2z}$$

A 3-Step Method:

1. Locate dominant singularities;
2. Compute local behaviour;
3. Translate into asymptotics.

1. sing. en $z = 1/4$
2. local behaviour:

$$B(z) \underset{z\to 1/4}{=} 2 - 2(1-4z)^{1/2} + \cdots$$

3. translation:

$$B_n \sim \frac{4^n n^{-3/2}}{\sqrt{\pi}}\left(1 - \frac{9}{8n} + \cdots\right)$$

Probability that a leaf is a child of the root?

$$\frac{[z^n]2zB(z)}{B_n} = \frac{1}{2} + \frac{3}{4n} + O\left(\frac{1}{n^2}\right)$$

Demo.

Parameters

Parameters

- Equations over combinatorial structures + parameters

Ex.: path length in binary trees

Parameters

- Equations over combinatorial structures + parameters
- Multivariate generating series

$$F(z,u) = \sum_{n,k} f_{n,k} u^k z^n$$

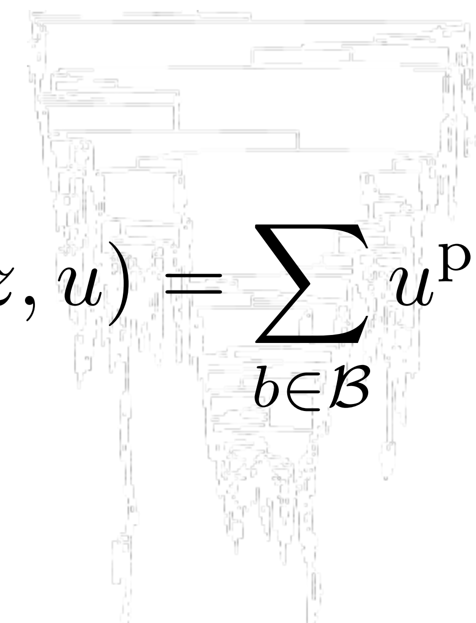
Ex.: path length in binary trees

Parameters

- Equations over combinatorial structures + parameters
- Multivariate generating series

$$F(z,u) = \sum_{n,k} f_{n,k} u^k z^n$$

Ex.: path length in binary trees



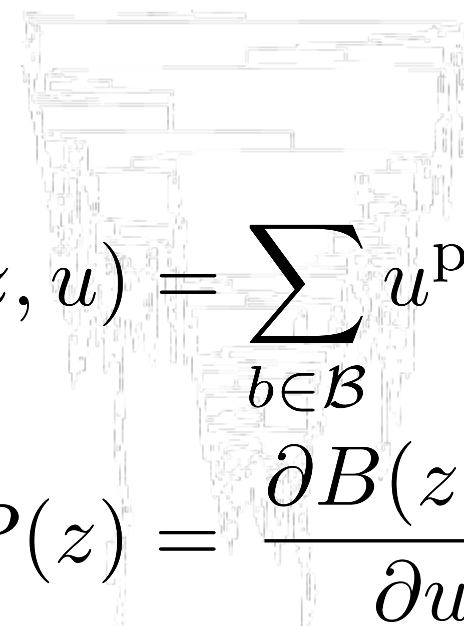
$$B(z,u) = \sum_{b \in \mathcal{B}} u^{\mathrm{pl}(b)} z^{|b|}$$

Parameters

- Equations over combinatorial structures + parameters
- Multivariate generating series

$$F(z,u) = \sum_{n,k} f_{n,k} u^k z^n$$

Ex.: path length in binary trees



$$B(z,u) = \sum_{b \in \mathcal{B}} u^{\mathrm{pl}(b)} z^{|b|}$$

$$P(z) = \left. \frac{\partial B(z,u)}{\partial u} \right|_{u=1}$$

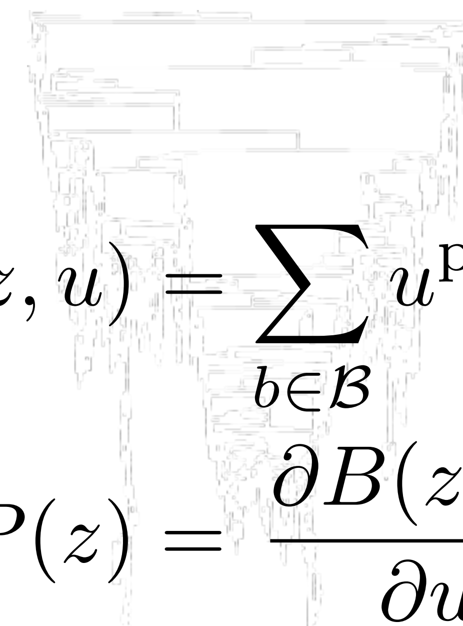
Parameters

- Equations over combinatorial structures + parameters
- Multivariate generating series

$$F(z,u) = \sum_{n,k} f_{n,k} u^k z^n$$

- Complex analysis
  $f_{n,k} \sim \cdots, n \to \infty$

Ex.: path length in binary trees



$$B(z,u) = \sum_{b \in \mathcal{B}} u^{\mathrm{pl}(b)} z^{|b|}$$

$$P(z) = \left. \frac{\partial B(z,u)}{\partial u} \right|_{u=1}$$

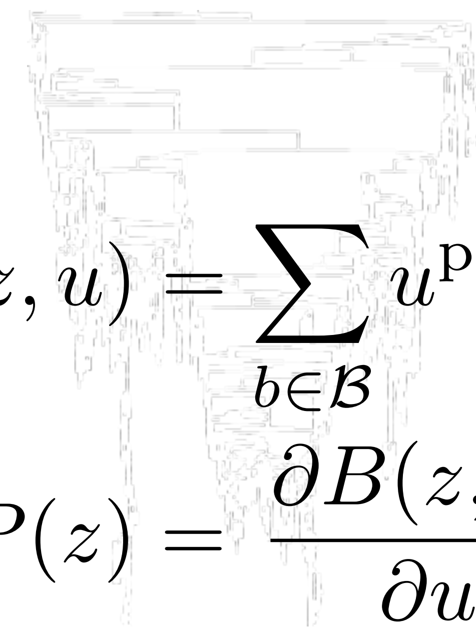
Parameters

- Equations over combinatorial structures + parameters
- Multivariate generating series

$$F(z,u) = \sum_{n,k} f_{n,k} u^k z^n$$

- Complex analysis
  $f_{n,k} \sim \cdots, n \to \infty$

Ex.: path length in binary trees



$$B(z,u) = \sum_{b \in \mathcal{B}} u^{\mathrm{pl}(b)} z^{|b|}$$

$$P(z) = \left.\frac{\partial B(z,u)}{\partial u}\right|_{u=1}$$

$$\frac{P_n}{nB_n} \sim \sqrt{\pi n}$$

Parameters

- Equations over combinatorial structures + parameters

- Multivariate generating series

$$F(z,u) = \sum_{n,k} f_{n,k} u^k z^n$$

- Complex analysis
  $f_{n,k} \sim \cdots, n \to \infty$

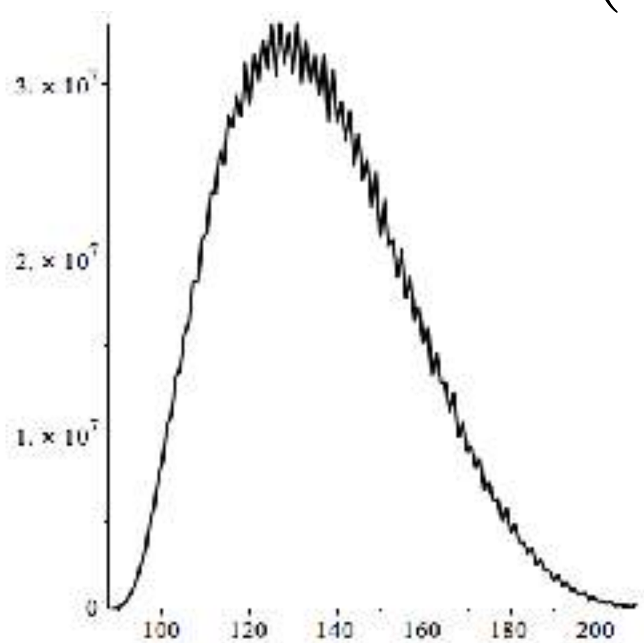
- Limiting distribution

Ex.: path length in binary trees

$$B(z,u) = \sum_{b \in \mathcal{B}} u^{\mathrm{pl}(b)} z^{|b|}$$

$$P(z) = \left. \frac{\partial B(z,u)}{\partial u} \right|_{u=1}$$

$$\frac{P_n}{nB_n} \sim \sqrt{\pi n}$$



Parameters

- Equations over combinatorial structures + parameters
- Multivariate generating series

$$F(z,u) = \sum_{n,k} f_{n,k} u^k z^n$$

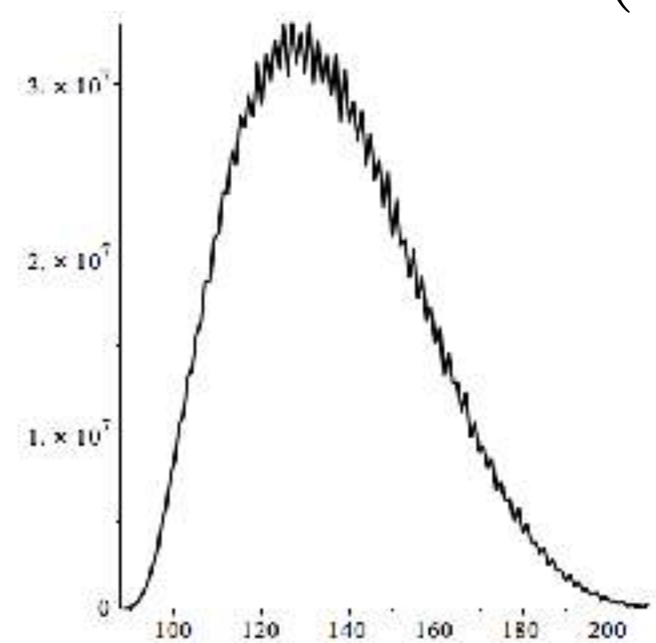
- Complex analysis
  $f_{n,k} \sim \cdots, n \to \infty$
- Limiting distribution

Ex.: path length in binary trees

$$B(z,u) = \sum_{b \in \mathcal{B}} u^{\mathrm{pl}(b)} z^{|b|}$$

$$P(z) = \left.\frac{\partial B(z,u)}{\partial u}\right|_{u=1}$$

$$\frac{P_n}{nB_n} \sim \sqrt{\pi n}$$



Demo.

If you can specify it, you can analyze it!

Permutations	Languages
Mappings	Integers
Words	Compositions
Strings	Partitions
Urns	...
Trees	

+ *parameters*

Universality phenomena:

Ex.: # trees of various types $\longrightarrow$ $K\rho^n n^{-3/2}$ $\longrightarrow$ pathlength in $n^{1/2}$

Summary & Conclusion

Summary & Conclusion

What we have:

Combinatorial specification →

	Automated?
counting;	✓
random generation;	✓
asymptotic analysis.	✗ (not fully)

Summary & Conclusion

What we have:

		Automated?
Combinatorial specification →	counting;	✓
→	random generation;	✓
→	asymptotic analysis.	✗ (not fully)

Ultimate (dream) goal:

Combinatorial specification + Algorithm ⟶ Automatic complexity analysis

Summary & Conclusion

What we have:

Automated?

Combinatorial specification →

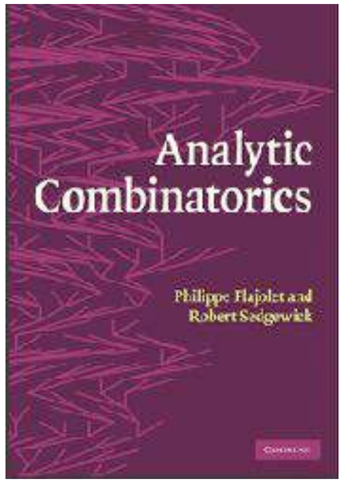
- counting; ✓
- random generation; ✓
- asymptotic analysis. ✗ (not fully)

Ultimate (dream) goal:

Combinatorial specification + Algorithm ⟶ Automatic complexity analysis

Where to learn more:

On-line course by R. Sedgewick at
http://ac.cs.princeton.edu/online/



Summary & Conclusion

What we have:

Automated?

Combinatorial specification →
- counting; ✓
- random generation; ✓
- asymptotic analysis. ✗ (not fully)

The End

Ultimate (dream) goal

Combinatorial specification + Algorithm → Automatic complexity analysis

Where to learn more:

On-line course by R. Sedgewick at
http://ac.cs.princeton.edu/online/