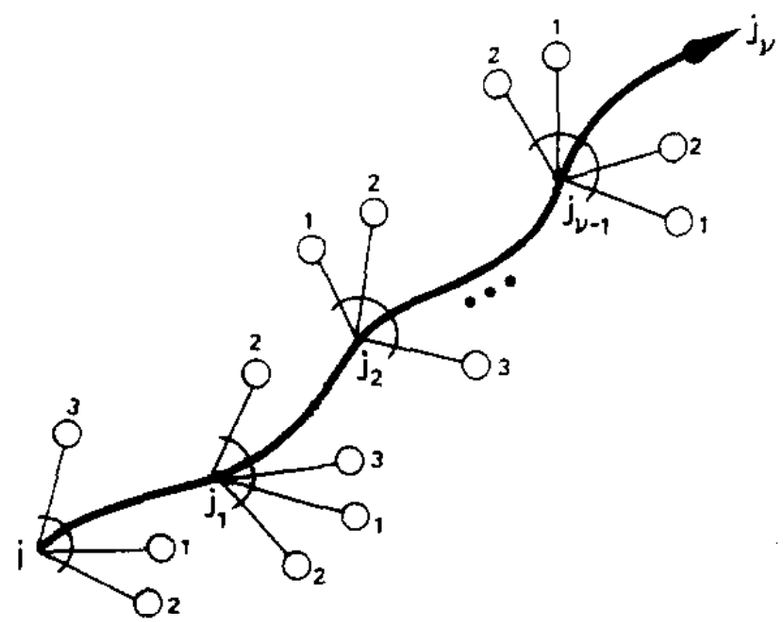
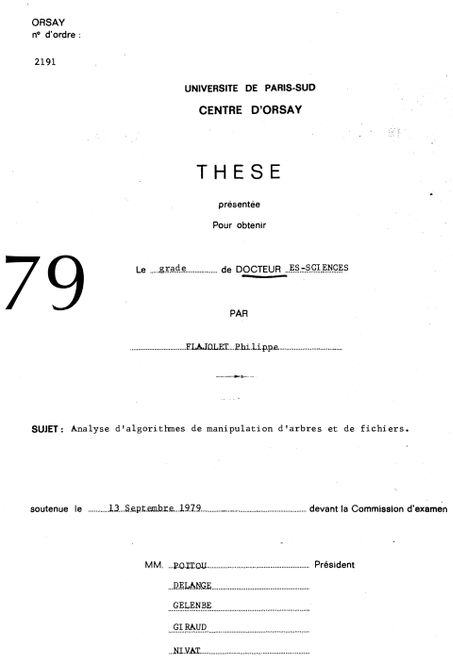
Combinatorial Newton Iteration with Michèle

Bruno Salvy
Inria & ENS de Lyon



I. Introduction

Analytic Combinatorics



ORSAY
n° d'ordre :

2191

UNIVERSITE DE PARIS-SUD
CENTRE D'ORSAY

THESE

présentée

Pour obtenir

Le grade de DOCTEUR ES-SCIENCES

PAR

FLAJOLET Philippe

SUJET : Analyse d'algorithmes de manipulation d'arbres et de fichiers.

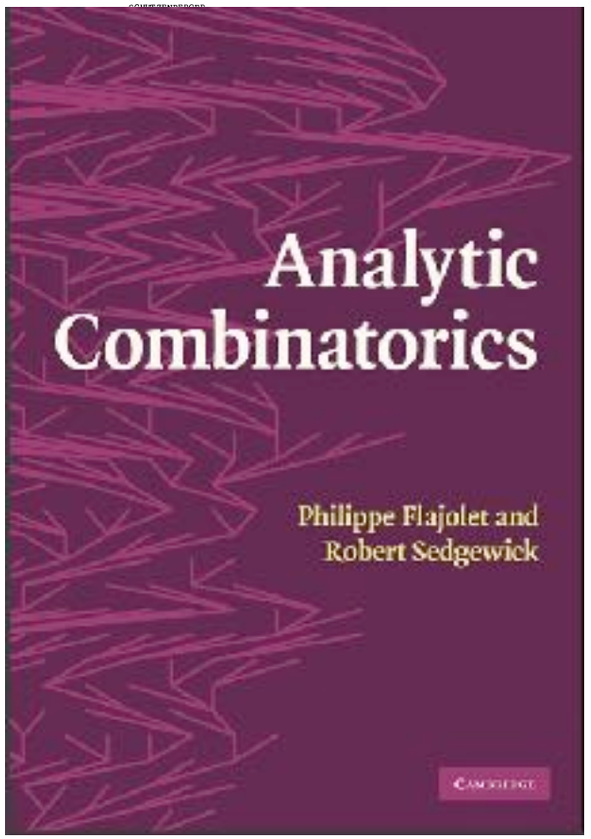
soutenue le 13 Septembre 1979 devant la Commission d'examen

MM. POITOU Président
DELANGE
GELENBE
GIRAUD
NIVAT

1979

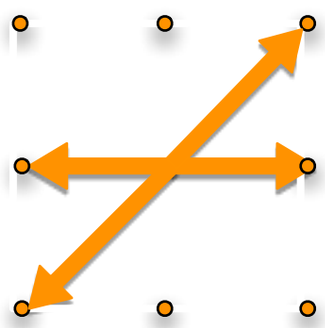
From a ***specification*** to

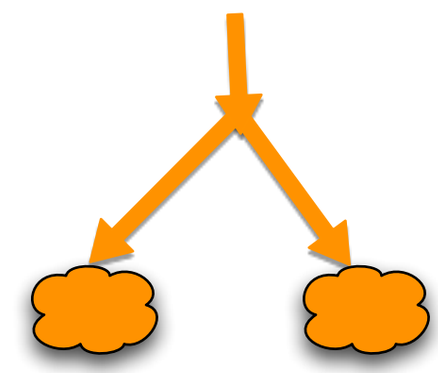
- enumeration;
- analysis;
- random generation.



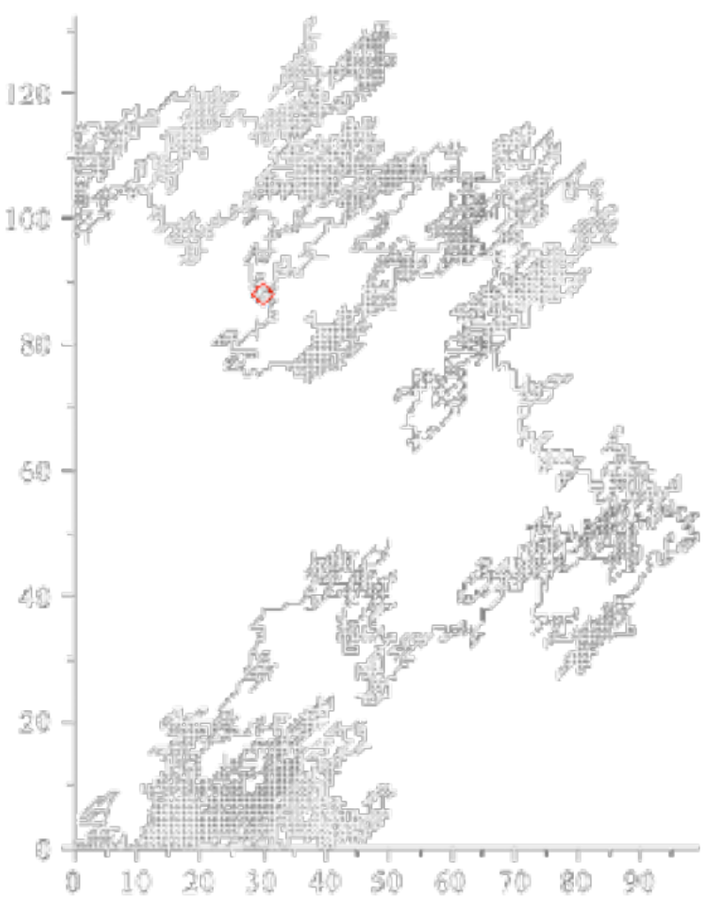
2009

Combinatorics, Randomness and Analysis





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

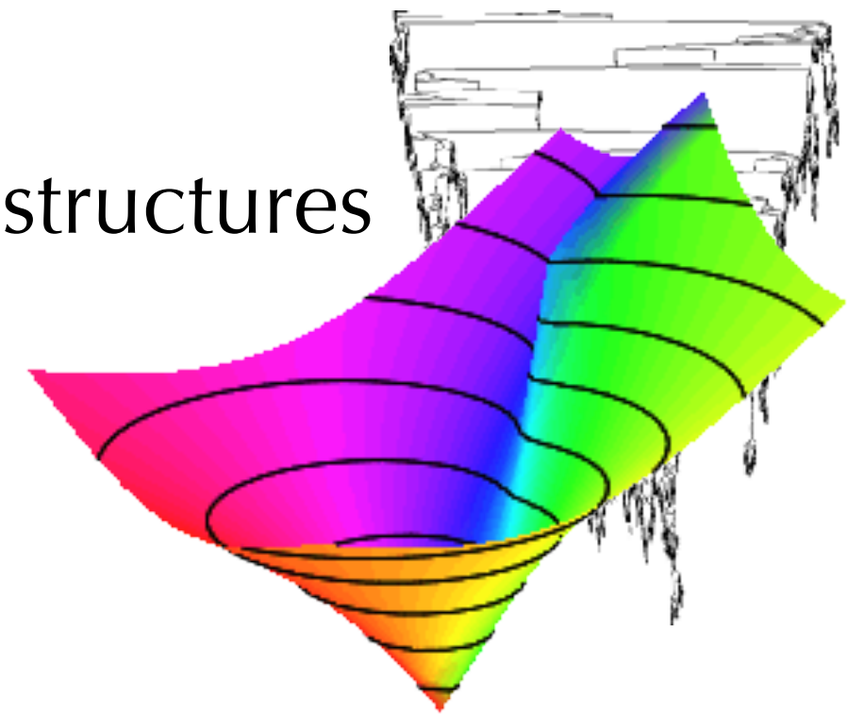


A quest for **universality** in random discrete structures**:**
➡ **probabilistic complexity** of structures and algorithms.

Quantitative results using **complex analysis.**

Overview (1/2)

Ex.: binary trees



- Equations over combinatorial structures
- Generating functions

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

- Complex analysis

$$f_n \sim \cdots, \; n \to \infty$$

$$\mathcal{B} = \mathcal{Z} \cup \mathcal{B} \times \mathcal{B}$$

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

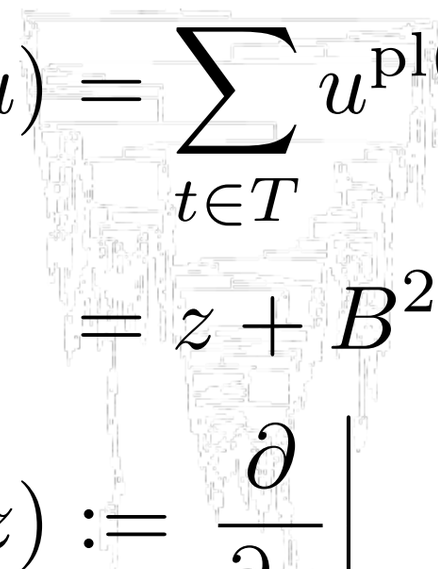
$$B_n \sim \frac{4^{n-1} n^{-3/2}}{\sqrt{\pi}}$$

Overview (2/2)

Ex.: path length in binary trees

- Equations over combinatorial structures + parameters

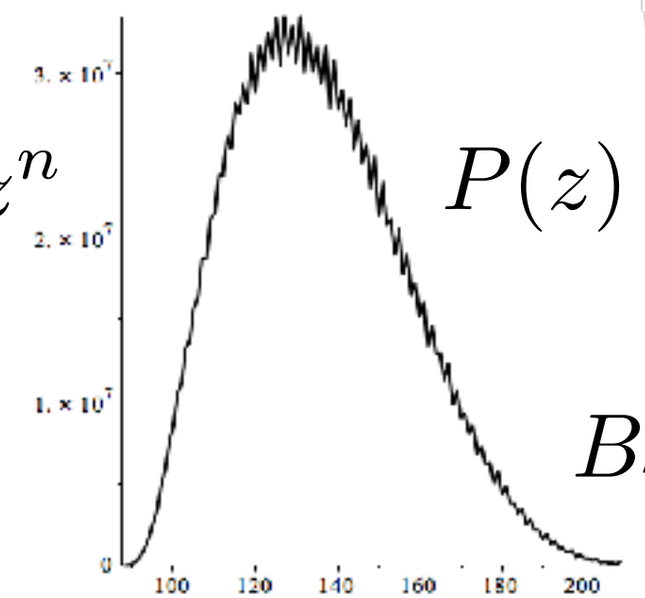
$$B(z,u) = \sum_{t \in T} u^{\mathrm{pl}(t)} z^{|t|}$$



$$= z + B^2(zu,u)$$

- Multivariate generating series

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



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

$$B_n \sim \frac{4^{n-1} n^{-3/2}}{\sqrt{\pi}}$$

- Complex analysis

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

$$P_n \sim 4^{n-1}$$

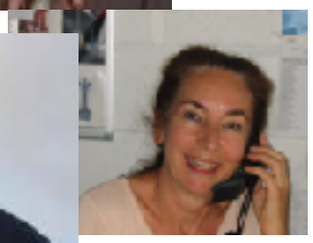
- Limiting distribution

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

Michèle's work in Analytic Combinatorics

84-89: complexity rewriting

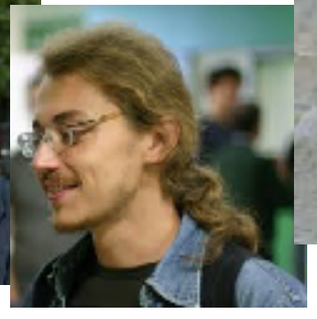
90-95: limiting distributions (➥ Michael's talk)

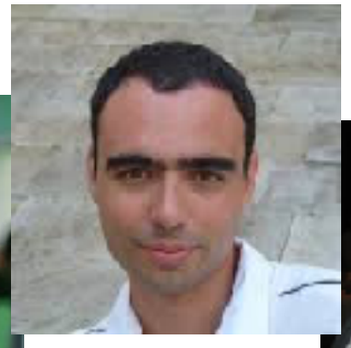


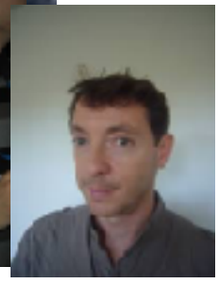
98-01: planar maps & Airy

05-12: Boltzmann sampling (⇜ incl. our joint work)

11-...: k-trees (➥ Hsien-Kuei's talk)



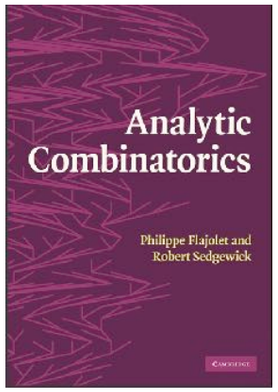


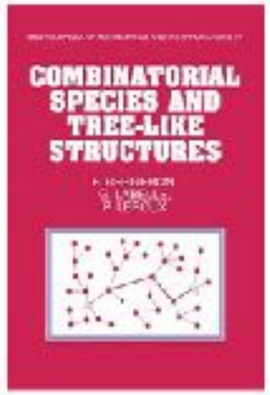


Our joint work

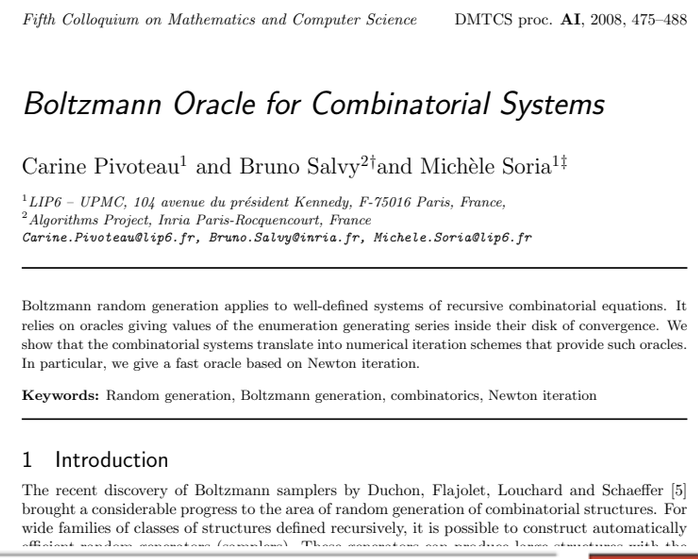
(with Carine & Michèle)

- A bridge between analytic combinatorics and species theory





- Algorithms for: checking consistency of combinatorial systems; fast enumeration; Boltzmann oracle.



Fifth Colloquium on Mathematics and Computer Science DMTCS proc. **AI**, 2008, 475–488

Boltzmann Oracle for Combinatorial Systems

Carine Pivoteau[1] and Bruno Salvy[2†] and Michèle Soria[1‡]

[1] *LIP6 – UPMC, 104 avenue du président Kennedy, F-75016 Paris, France,*
[2] *Algorithms Project, Inria Paris-Rocquencourt, France*
Carine.Pivoteau@lip6.fr, Bruno.Salvy@inria.fr, Michele.Soria@lip6.fr

Boltzmann random generation applies to well-defined systems of recursive combinatorial equations. It relies on oracles giving values of the enumeration generating series inside their disk of convergence. We show that the combinatorial systems translate into numerical iteration schemes that provide such oracles. In particular, we give a fast oracle based on Newton iteration.

Keywords: Random generation, Boltzmann generation, combinatorics, Newton iteration

1 Introduction

The recent discovery of Boltzmann samplers by Duchon, Flajolet, Louchard and Schaeffer [5] brought a considerable progress to the area of random generation of combinatorial structures. For wide families of classes of structures defined recursively, it is possible to construct automatically

Contents lists available at SciVerse ScienceDirect

Journal of Combinatorial Theory, Series A

www.elsevier.com/locate/jcta

ELSEVIER

Algorithms for combinatorial structures: Well-founded systems and Newton iterations ☆

Carine Pivoteau [a], Bruno Salvy [b], Michèle Soria [c]

[a] Université Paris-Est, LIGM (CNRS UMR 8049), Marne-la-Vallée, France
[b] Paris, France
[c] Université Pierre et Marie Curie LIP6 (CNRS UMR 7606), Paris, France

ARTICLE INFO

Article history:
Received 15 September 2011
Available online xxxx

This article is dedicated to the memory of Philippe Flajolet

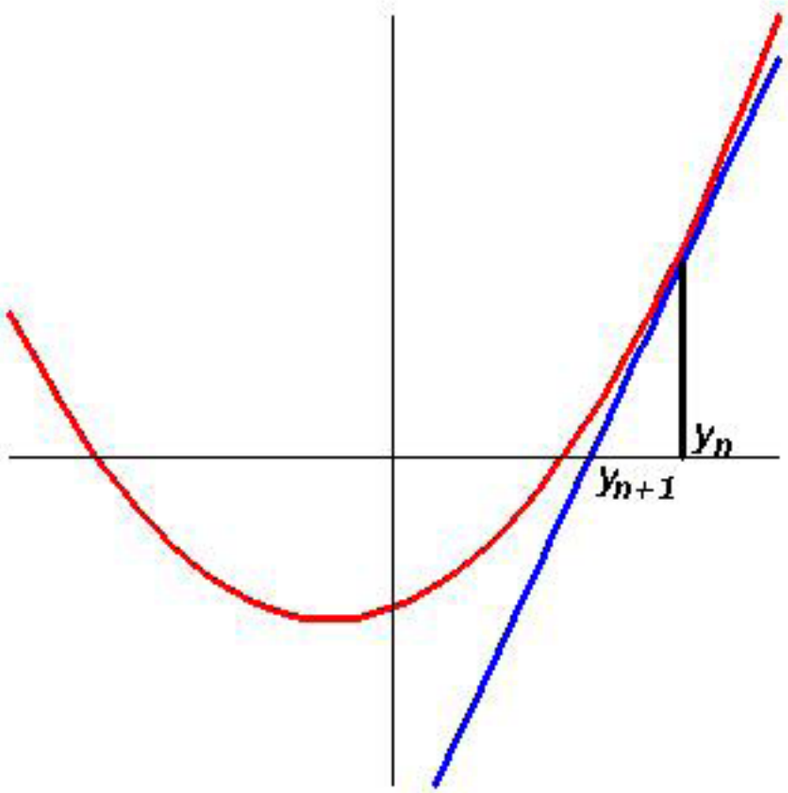
Keywords:
Species theory
Analytic combinatorics
Newton iteration
Generating functions
Complexity

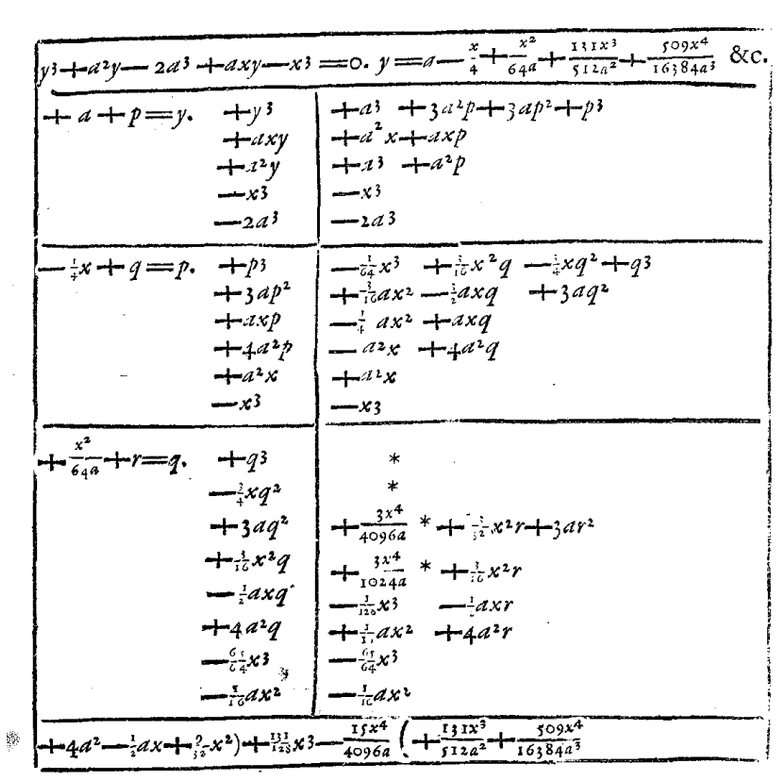
ABSTRACT

We consider systems of recursively defined combinatorial structures. We give algorithms checking that these systems are well-founded, computing generating series and providing numerical values. Our framework is an articulation of the constructible classes of Flajolet and Sedgewick with Joyal's species theory. We extend the implicit species theorem to structures of size zero. A quadratic iterative Newton method is shown to solve well-founded systems combinatorially. From there, truncations of the corresponding generating series are obtained in quasi-optimal complexity. This iteration transfers to a numerical scheme that converges unconditionally to the values of the generating series inside their disk of convergence. These results provide important subroutines in random generation. Finally, the approach is extended to combinatorial differential systems.

© 2012 Elsevier Inc. All rights reserved.

II. Newton Iteration





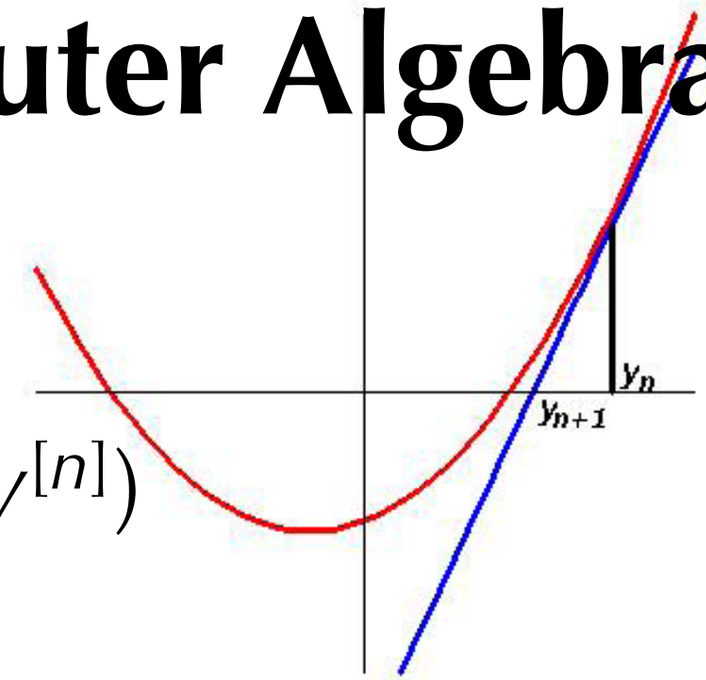
$y^3 + a^2y - 2a^3 + axy - x^3 = 0.$ $y = a - \frac{x}{4} + \frac{x^2}{64a} + \frac{131x^3}{512a^2} + \frac{509x^4}{16384a^3}$ &c.

$+a+p=y.$	$+y^3$	$+a^3 \; +3a^2p+3ap^2+p^3$
	$+axy$	$+a^2x+axp$
	$+a^2y$	$+a^3 \; +a^2p$
	$-x^3$	$-x^3$
	$-2a^3$	$-2a^3$
$-\frac{1}{4}x+q=p.$	$+p^3$	$-\frac{1}{64}x^3 \; +\frac{3}{16}x^2q \; -\frac{3}{4}xq^2+q^3$
	$+3ap^2$	$+\frac{3}{16}ax^2 \; -\frac{3}{2}axq \; +3aq^2$
	$+axp$	$-\frac{1}{4}ax^2 \; +axq$
	$+4a^2p$	$-a^2x \; +4a^2q$
	$+a^2x$	$+a^2x$
	$-x^3$	$-x^3$
$+\frac{x^2}{64a}+r=q.$	$+q^3$	*
	$-\frac{3}{4}xq^2$	*
	$+3aq^2$	$+\frac{3x^4}{4096a}$ * $+\frac{3}{32}x^2r+3ar^2$
	$+\frac{3}{16}x^2q$	$+\frac{3x^4}{1024a}$ * $+\frac{3}{16}x^2r$
	$-\frac{3}{2}axq$	$-\frac{3}{128}x^3 \; -\frac{3}{2}axr$
	$+4a^2q$	$+\frac{3}{4}ax^2 \; +4a^2r$
	$-\frac{61}{64}x^3$	$-\frac{61}{64}x^3$
	$-\frac{3}{16}ax^2$	$-\frac{3}{16}ax^2$

$+4a^2-\frac{1}{4}ax+\frac{9}{32}x^2)+\frac{133}{128}x^3-\frac{15x^4}{4096a}\left(+\frac{131x^3}{512a^2}+\frac{509x^4}{16384a^3}\right.$

1671

Newton Iteration in Computer Algebra



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

$y^{[n+1]}=y^{[n]}+u^{[n]}$, with $\phi'(y^{[n]})u^{[n]}=-\phi(y^{[n]})$

Quadratic convergence ⟷ Divide-and-Conquer

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

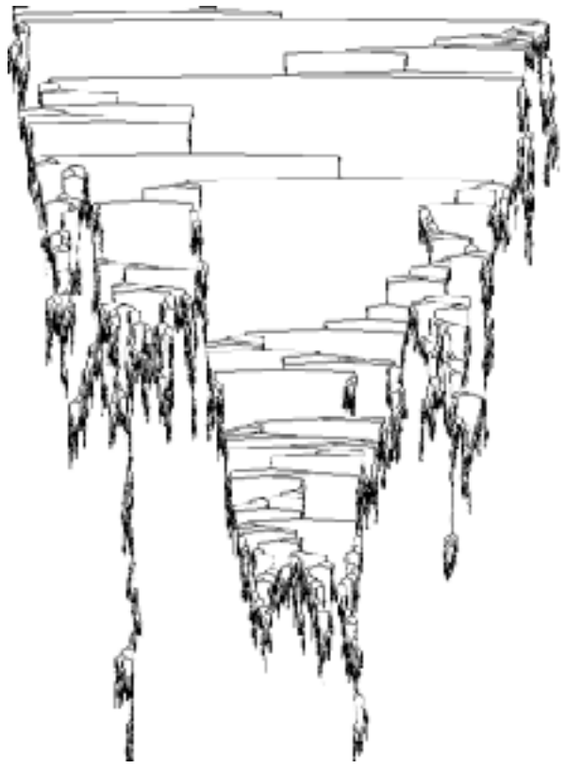
Cost(N) = ct × Cost(last step)

Useful with fast multiplication (e.g., FFT)

→ $O(N \log N)$ ops.

Catalan numbers (binary trees)

Generating function satisfies $y = 1 + zy^2$



Newton iteration

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

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

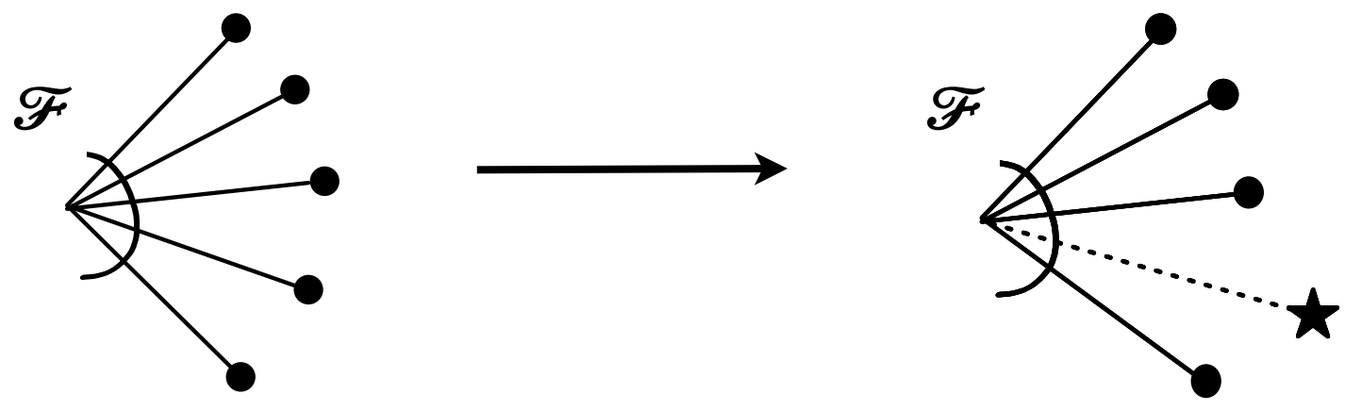
$$y^{[1]} = \boxed{1}$$

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

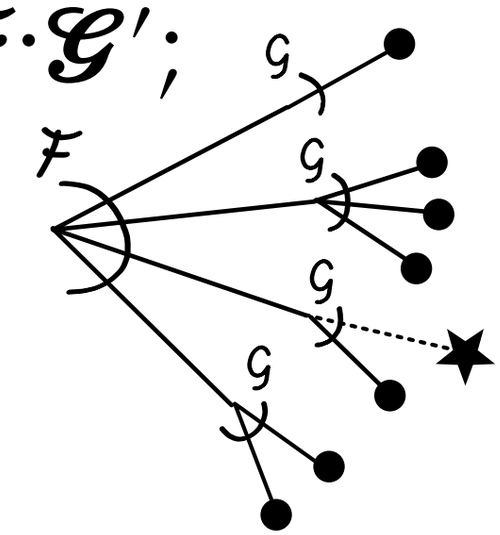
$$y^{[3]} = \boxed{1 + z + 2z^2 + 5z^3 + 14z^4 + 42z^5 + 132z^6} + 428z^7 + \ldots$$

Derivative of Combinatorial Structures

(origin: species theory)



- $0'=1'=0$; $Z'=1$; $\textsc{Set}'=\textsc{Set}$; $\textsc{Cyc}'=\textsc{Seq}$; $\textsc{Seq}'=\textsc{Seq}\cdot\textsc{Seq}$;
- $(\mathscr{F}+\mathscr{G})'=\mathscr{F}'+\mathscr{G}'$; $(\mathscr{F}\cdot\mathscr{G})'=\mathscr{F}'\cdot\mathscr{G}+\mathscr{F}\cdot\mathscr{G}'$;
- $\mathscr{F}(\mathscr{G})'=\mathscr{F}'(\mathscr{G})\cdot\mathscr{G}'$.



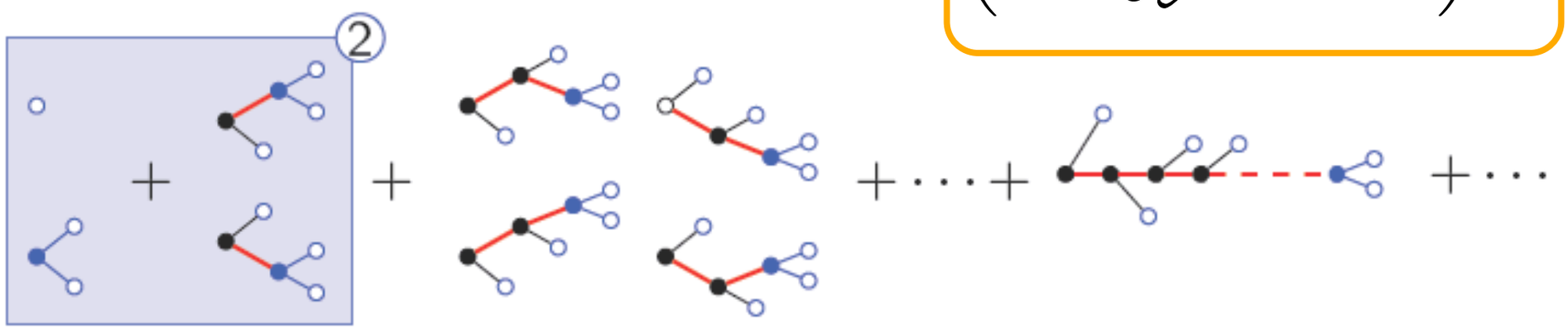
Newton Iteration for binary trees

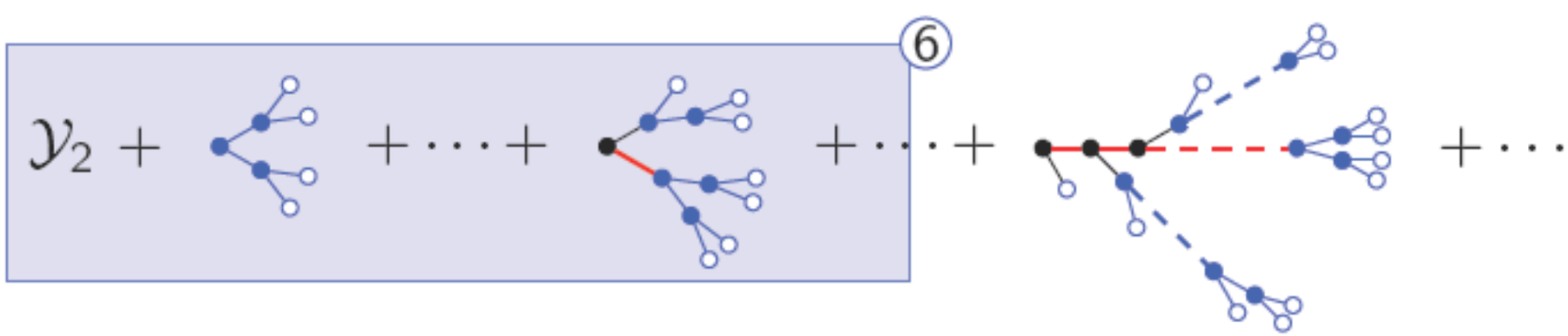
$$\mathcal{Y} = 1 + \mathcal{Z} \cdot \mathcal{Y} \cdot \mathcal{Y} =: \mathcal{H}(\mathcal{Z}, \mathcal{Y})$$

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

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

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

$\mathcal{Y}_2 =$ 

$\mathcal{Y}_3 =$ 

Generation by increasing Strahler numbers

[Décoste, Labelle, Leroux 1982]

Application: Unlabelled Rooted Trees

Combinatorial equation: $\mathcal{T} = \mathcal{Z} \cdot \textsc{Set}(\mathcal{T}) =: \mathcal{H}(\mathcal{Z}, \mathcal{T})$

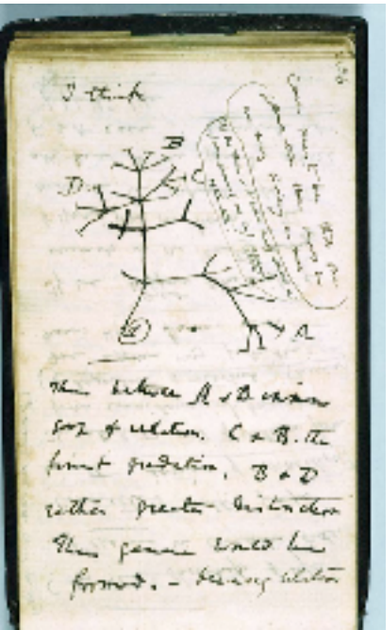
Combinatorial Newton iteration:

$$\mathcal{T}^{[n+1]} = \mathcal{T}^{[n]} + \textsc{Seq}(\mathcal{H}(\mathcal{T}^{[n]})) \cdot (\mathcal{H}(\mathcal{T}^{[n]}) \setminus \mathcal{T}^{[n]})$$

OGF equation: $\tilde{T}(z) = H(z, \tilde{T}(z))$

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

Newton for OGF



$$\tilde{T}^{[n+1]} = \tilde{T}^{[n]} + \frac{H(z, \tilde{T}^{[n]}) - \tilde{T}^{[n]}}{1 - H(z, \tilde{T}^{[n]})}$$

$$\boxed{0},$$

$$\boxed{z + z^2} + z^3 + z^4 + \cdots,$$

$$\boxed{z + z^2 + 2z^3 + 4z^4 + 9z^5} + 20z^6 + \cdots$$

Our Result

Thm. [Pivoteau-S-Soria 2012] First N coefficients of GFs of all *constructible* species in

1. arithmetic complexity

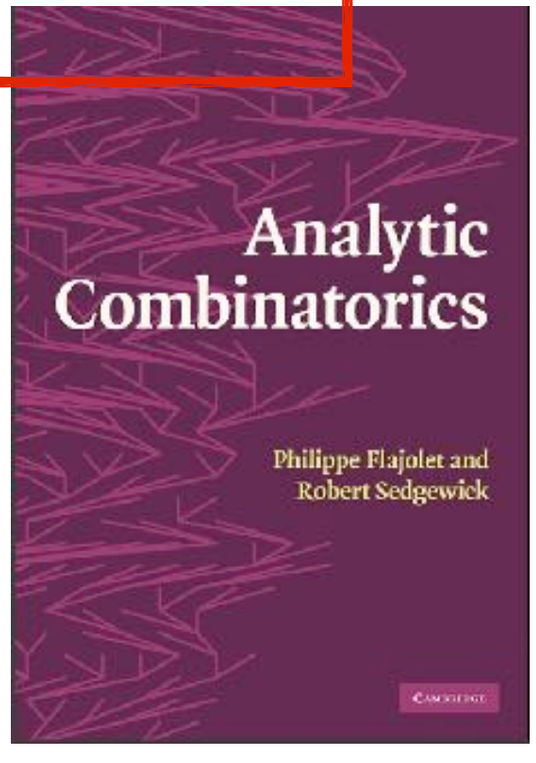
$O(N \log N)$ (both ogf & egf);

2. bit complexity

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

```
> with(NewtonGF):
> Trees:={T=Prod(Z,Set(T))}:
> GFSeries(Trees,unlabelled,z,21)[2];
```

$$T = z + z^2 + 2z^3 + 4z^4 + 9z^5 + 20z^6 + 48z^7 + 115z^8 + 286z^9 + 719z^{10} + 1842z^{11} + 4766z^{12} + 12486z^{13} + 32973z^{14} + 87811z^{15} + 235381z^{16} + 634847z^{17} + 1721159z^{18} + 4688676z^{19} + 12826228z^{20} + O(z^{21})$$



III. Boltzmann Sampling and its Oracle

Algorithm

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

x is a parameter of the algorithm

- Singleton: easy.
- Cartesian product: $\mathscr{H}=\mathscr{F}\times\mathscr{G}$

```
Generate f∈𝓕;
Generate g∈𝓖;
return (f,g);
```

- Disjoint union $\mathscr{H}=\mathscr{F}+\mathscr{G}$

```
b=Bernoulli(F(x)/H(x));
if b=1 then Generate f∈𝓕
else Generate g∈𝓖;
```

Example: binary tree with F(.49)/H(.49)≃.5995

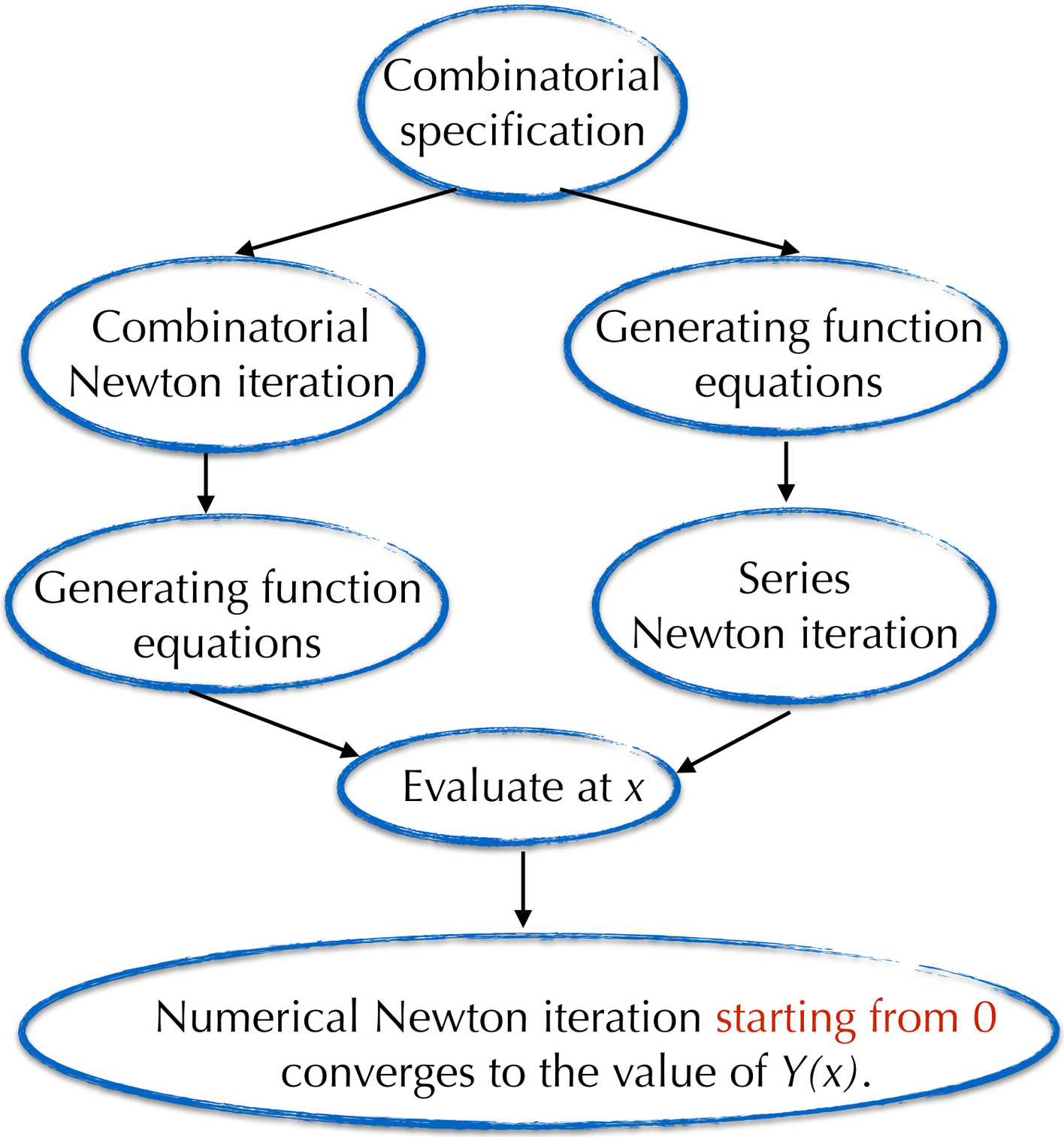
$$+ \mathcal{Z} \times \mathcal{B} \times \mathcal{B}$$

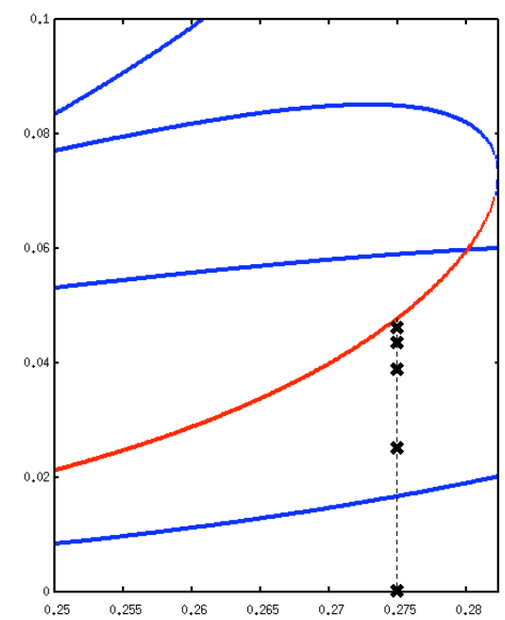
Coin: 0,1,0,1,0,1,1,0,1,1,1,...

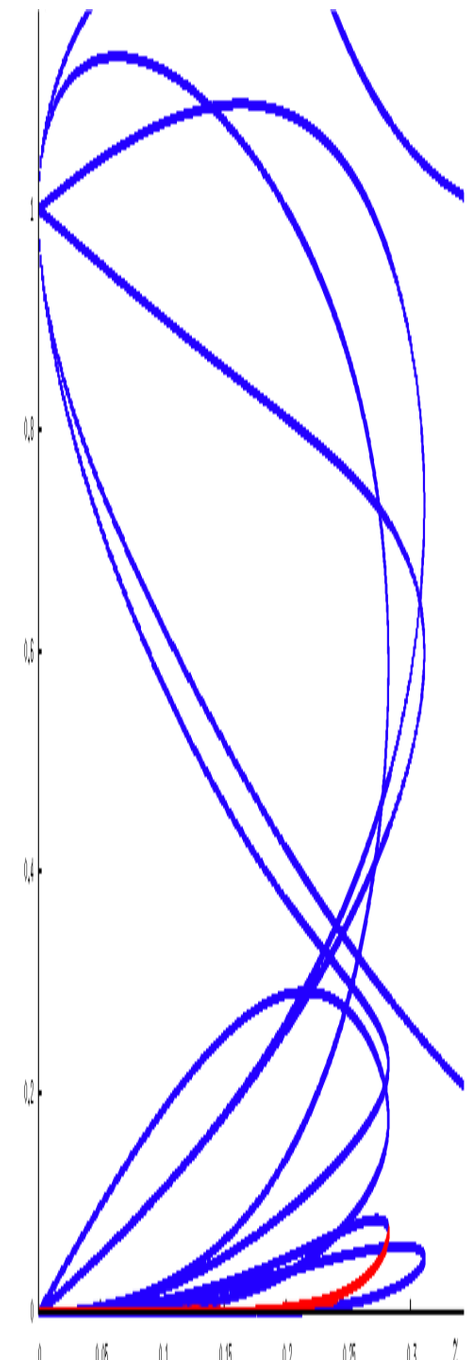
Complexity linear in |t|, given T(x)

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

Oracle: Labelled Case







Demo
NewtonGF

Unlabelled Oracle on an Example

Combinatorial equation: $\mathcal{T} = \mathcal{Z} \cdot \textsc{Set}(\mathcal{T}) =: \mathcal{H}(\mathcal{Z}, \mathcal{T})$

Combinatorial Newton iteration:

$$\mathcal{T}^{[n+1]} = \mathcal{T}^{[n]} + \textsc{Seq}(\mathcal{H}(\mathcal{T}^{[n]})) \cdot (\mathcal{H}(\mathcal{T}^{[n]}) \setminus \mathcal{T}^{[n]})$$

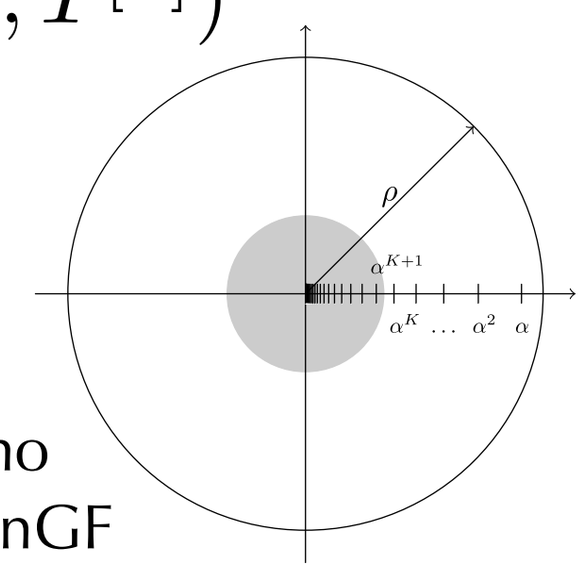
OGF equation: $\tilde{T}(z) = H(z, \tilde{T}(z))$

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

Newton for OGF

$$\tilde{T}^{[n+1]} = \tilde{T}^{[n]} + \frac{H(z, \tilde{T}^{[n]}) - \tilde{T}^{[n]}}{1 - H(z, \tilde{T}^{[n]})}$$

n	$\tilde{T}^{[n]}(0.3)$	$\tilde{T}^{[n]}(0.3^2)$	$\tilde{T}^{[n]}(0.3^3)$
0	0	0	0
1	.43021322639	0.99370806338e-1	0.27759817516e-1
2	.54875612912	0.99887132154e-1	0.27770629187e-1
3	.55709557053	0.99887147197e-1	0.27770629189e-1
4	.55713907945	0.99887147198e-1	0.27770629189e-1
5	.55713908064	0.99887147198e-1	0.27770629189e-1



Demo
NewtonGF

NewtonGF: from algos to code

(with Carine Pivoteau)

- Nov. 2005: first Maple worksheet
- Sep. 2008: first basic version distributed (different name)
- Fev. 2012: better numerical precision
- Nov. 2013: cardinality constraints
- Oct. 2014: v1.0 Labelled case complete
- Janv. 2015: v1.1. Series in the unlabelled case
- Jul. 2015: v1.2. Numerics in the unlabelled case; faster series expansions (Pablo).
- Jul. 2016: v1.3. Faster Newton iteration (exs MBM).
- Jan. 2017: v2.0. Radius.
- Fev. 2017: v2.1. Bug fix.

More to come

http://perso.ens-lyon.fr/bruno.salvy/software/the-newtongf-package/

The End

Bonne retraite, Michèle !