De l'algorithmique à l'arithmétique via le calcul formel

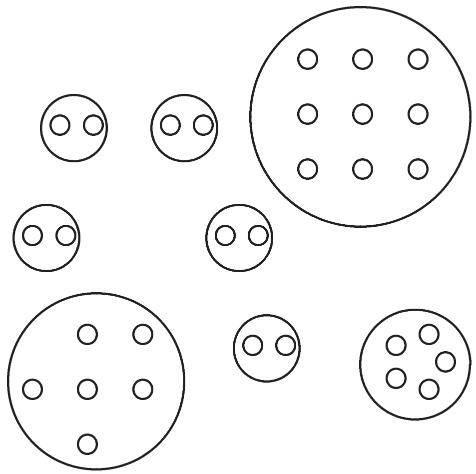
Paul Zimmermann

Plan

- Algorithmique : génération aléatoire
- Calcul formel : racines réelles d'un polynôme
- Arithmétique : « produit médian »
- Expérimentation : la bibliothèque MPFR

Génération aléatoire

Étant donnée une classe $\mathcal{A}$, et un entier n, comment générer aléatoirement et uniformément un objet de taille n ?
(On suppose $\mathcal{A}_n = \{a \in \mathcal{A}, |a| = n\}$ fini.)



L'une des 5604 partitions de taille 30 $(9 + 6 + 5 + 2 + 2 + 2 + 2 + 2)$.

Travaux antérieurs

1978, Nijenhuis, Wilf : algorithmes efficaces pour les compositions et partitions. Ébauche de la « méthode récursive ».

1980, Arnold, Sleep : algorithme linéaire pour mots de parenthèses :

$$P_{r,k} = \frac{1}{2}\frac{r(k+r+2)}{k(r+1)}.$$

1983, Hickey, Cohen : deux algorithmes généraux pour les langages *context-free*, en temps $\mathcal{O}(n^2 \log^2 n)$ ou espace $\mathcal{O}(n^{r+1})$.

1983, Greene : première utilisation systématique d'une description récursive pour la génération aléatoire.

1985, Rémy : algorithme linéaire pour les arbres binaires, via les arbres à feuilles numérotées.

1994, Mairson : $\mathcal{O}(n^2)$ en temps/espace pour les langages *context-free*.

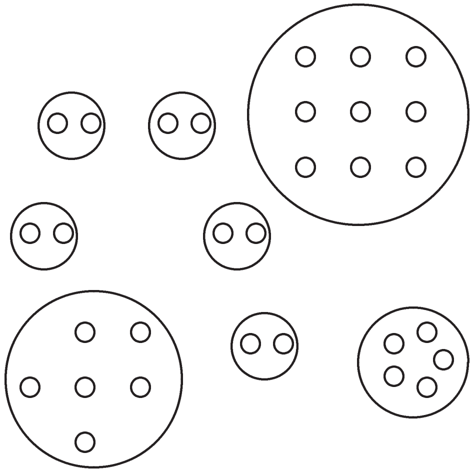
La méthode récursive (I)

Ph. Flajolet, P. Z., B. van Cutsem, *A Calculus for the Random Generation of Labelled Combinatorial Structures*, Theoretical Computer Science, 1994.

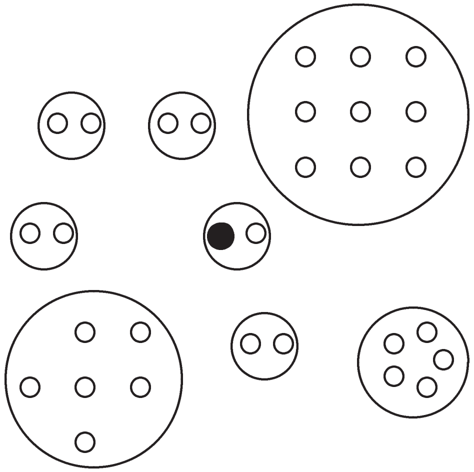
Structure *décomposable* : construite à partir de **1** et **Z** à l'aide des constructeurs `Union`, `Prod`, `Seq`, `Set`, `Cycle`.

$$P = \mathrm{Set}(E), \quad E = \mathrm{Union}(\mathbf{Z}, \mathrm{Prod}(\mathbf{Z}, E))$$

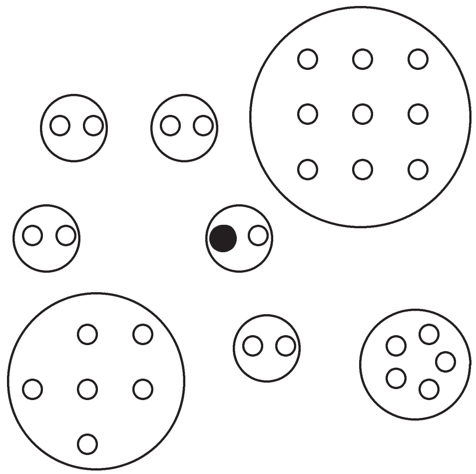
Opérateur de *marquage* : ΘA est l'ensemble des objets de $\mathcal{A}$ où l'un des atomes est distingué. $\Theta A_n = nA_n$.



Opérateur de *marquage* : ΘA est l'ensemble des objets de $\mathcal{A}$ où l'un des atomes est distingué. $\Theta A_n = nA_n$.



Opérateur de *marquage* : ΘA est l'ensemble des objets de $\mathcal{A}$ où l'un des atomes est distingué. $\Theta A_n = nA_n$.



Distribution uniforme sur $\Theta A \iff$ distribution uniforme sur A

Forme *standard* :

$\Theta P = Q, Q = R \cdot S, R = \Theta E, S = \Delta(P), E = \mathrm{Union}(\mathbf{Z}, F), F = \mathbf{Z} \cdot E.$

Précalcul : pour tout non-terminal T, et $k \leq n$, calculer $T_n = |\mathcal{T}_n|$.

La méthode récursive (II)

1. $A \to \mathbf{1}$ $\Longrightarrow$ `1`
2. $A \to \mathbf{Z}$ $\Longrightarrow$ `Z`
3. $A \to \mathrm{Union}(B, C)$ $\Longrightarrow$

```
if rand(A(n)) < B(n) then gB(n)
else gC(n) fi
```

4. $A \to \mathrm{Prod}(B, C)$ $\Longrightarrow$

```
u := rand(A(n));
for k := 0 to n while u >= 0 do
    u := u - B(k) * C(n-k);
Prod(gB(k), gC(n-k))
```

5. $A \to \Theta B$ $\Longrightarrow$ `gB(n)`
6. $\Theta A \to B$ $\Longrightarrow$ `gB(n)`

La méthode récursive (III)

Parcours séquentiel : complexité arithmétique $\mathcal{O}(n^2)$.

Parcours *boustrophédon* :

$4'$: `for k := 0 to n` $\Longrightarrow$ `for k in {0, n, 1, n-1, ...}`

$$f(n) \leq \max_{k \leq n}[\min(k, n-k) + f(k) + f(n-k)]$$

Complexité arithmétique $\mathcal{O}(n \log n)$ dans le pire cas !

Le logiciel COMBSTRUCT

Prototype GAÏA en 1994, intégré à MAPLE par Eithne Murray.

```
> with(combstruct):
> IntegerPartition := [P, {P=Set(E), E=Sequence(Z,card>=1)}]:
> seq(count(IntegerPartition, size=n), n=1..16);
1, 2, 3, 5, 7, 11, 15, 22, 30, 42, 56, 77, 101, 135, 176, 231
> draw(IntegerPartition, size=30);
Set(Sequence(Z, Z, Z, Z, Z, Z), Sequence(Z, Z), Sequence(Z, Z),

  Sequence(Z, Z), Sequence(Z, Z), Sequence(Z, Z),

  Sequence(Z, Z, Z, Z, Z), Sequence(Z, Z, Z, Z, Z, Z, Z, Z, Z))
```

Hybridation

A. Denise, P. Z., *Uniform Random Generation of Decomposable Structures Using Floating-Point Arithmetic*, Theoretical Computer Science, 1999

Complexité *arithmétique* $\mathcal{O}(n \log n)$, *binaire* $\mathcal{O}(n \log n M(n \log n))$:

```
> binTree := [B, {B = Union(Z, Prod(B, B))}]:
> count(binTree, size=100);
   227508830794229349661819540395688853956041682601541047340
```

Idée : calculer les T_k en virgule flottante, avec une précision p. Pour chaque valeur calculée u, on borne l'erreur $\epsilon(u)$.

Gestion de `if u < v` : on peut décider lorsque $|u - v| > \epsilon(u) + \epsilon(v)$.

Le logiciel CS

Réalisé avec Isabelle Dutour et Alain Denise, en MuPAD.

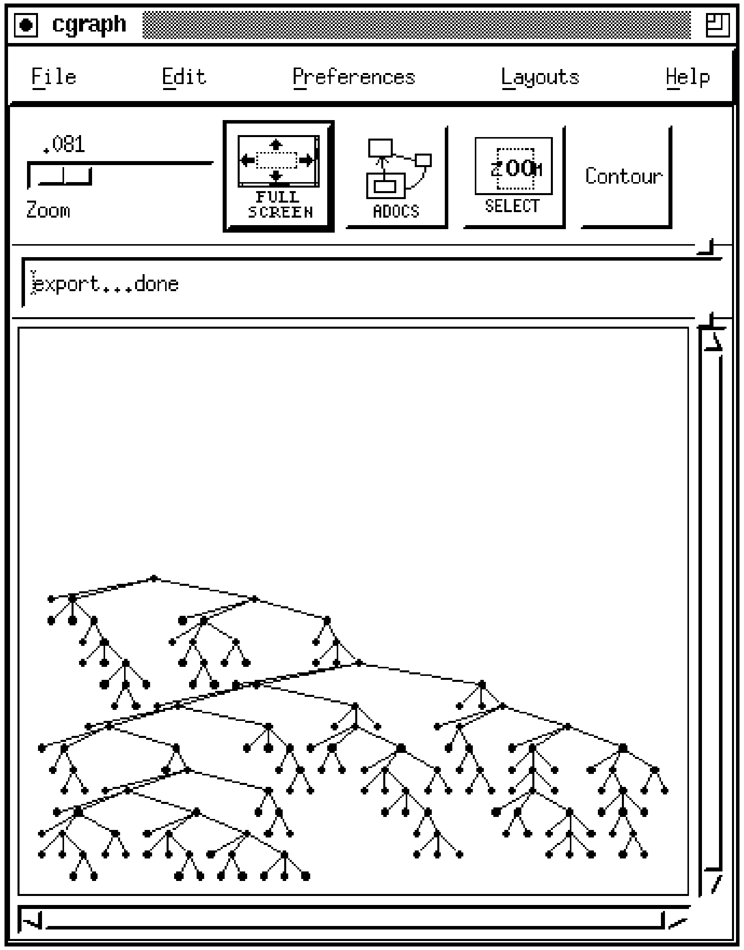
```
>> Schroeder := [S, {S=Union(Z, Prod(Z, Sequence(S, card>=2)))}]:
>> cs::count(Schroeder, size=100);
               6307128672549929005187958707700917978106 1461

>> cs::draw(Schroeder, size=17);
Prod(Z, Sequence(Z, Z, Prod(Z, Sequence(Z, Prod(Z, Sequence(Z, Z)),

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

>> cs::compile(Schroeder[2], target=C, file="schroeder.c", main=S);
```

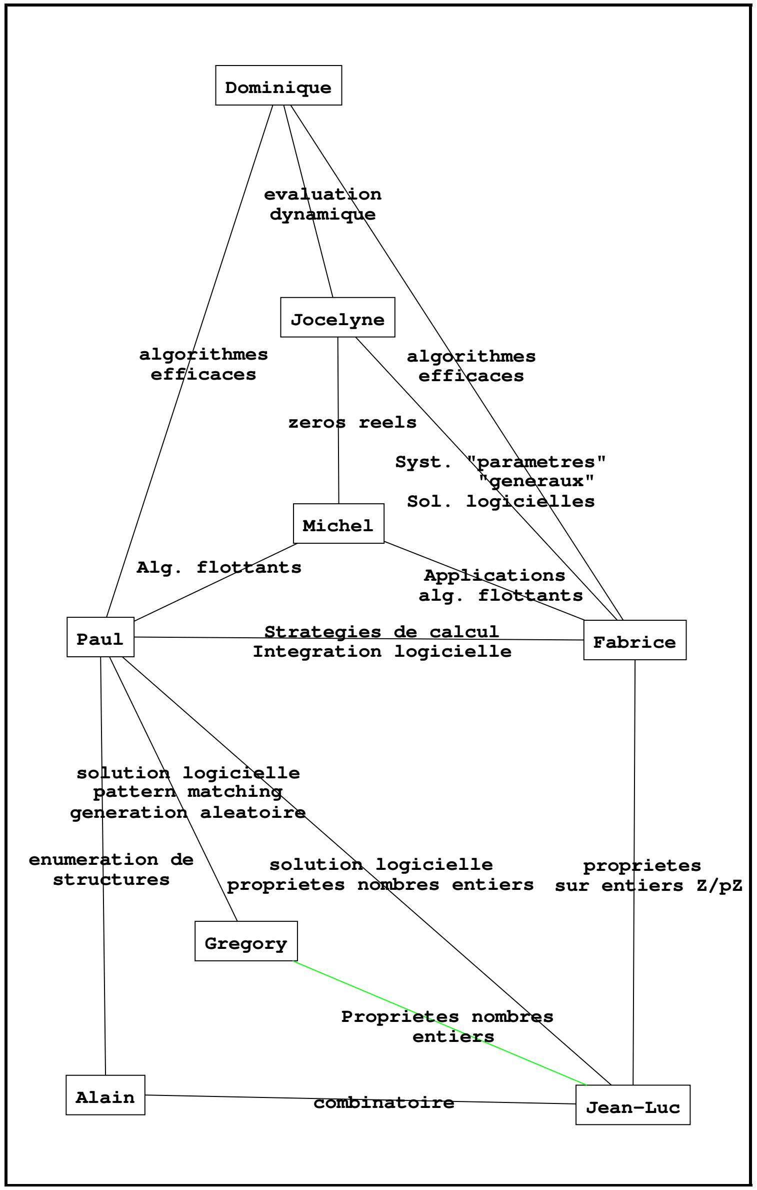
Tracé graphique

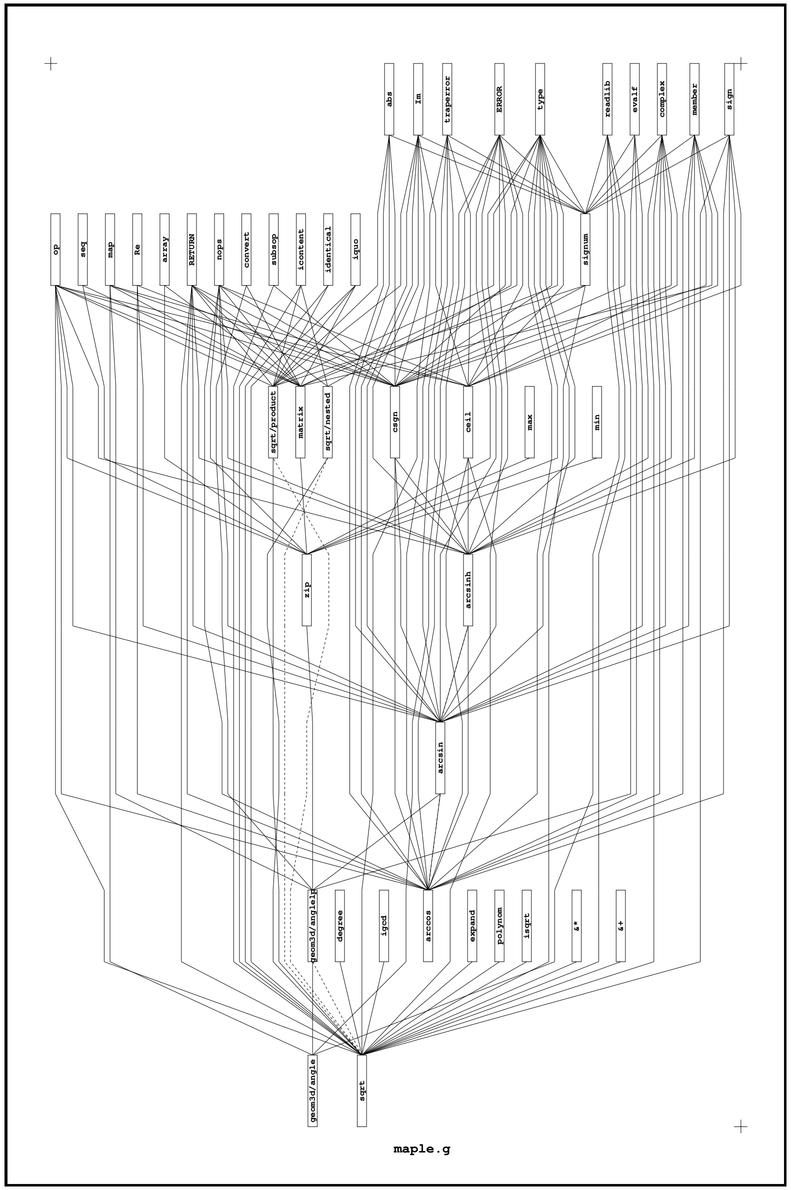
François Bertault, *Génération et tracé de structures décomposables*, Université Henri Poincaré Nancy 1, septembre 1997.

Structure décomposable ⇒ génération récursive ⇒ tracé récursif

Logiciels ADOCS, CGRAPH et PADNON.




Dominique
evaluation dynamique
Jocelyne
algorithmes efficaces
algorithmes efficaces
zeros reels
Syst. "parametres" "generaux" Sol. logicielles
Michel
Alg. flottants
Applications alg. flottants
Paul
Strategies de calcul Integration logicielle
Fabrice
solution logicielle pattern matching generation aleatoire
enumeration de structures
solution logicielle proprietes nombres entiers
proprietes sur entiers Z/pZ
Gregory
Proprietes nombres entiers
Alain
combinatoire
Jean-Luc


abs
Im
traperror
ERROR
type
readlib
evalf
complex
member
sign
op
seq
map
Re
array
RETURN
nops
convert
subsop
icontent
identical
iquo
signum
sqrt/product
matrix
sqrt/nested
csgn
ceil
max
min
zip
arcsinh
arcsin
degree
igcd
arccos
expand
polynom
isqrt
&*
&+
geom3d/angle
sqrt
maple.g

Perspectives

- implantation du précalcul en $\mathcal{O}(M(n)\log n)$ avec « calcul détendu » de J. van der Hoeven
- précalcul en $\mathcal{O}(M(n)\log n)$ avec calcul flottant ?
- *uniform approximate-size generation* (Duchon, Flajolet, Louchard, Schaeffer, 2001) : objet de taille m, $|m-n| \leq \epsilon n$ en $\mathcal{O}(1)$

Racines réelles d'un polynôme

Le problème : étant donné $P \in \mathbb{Z}[x]$, isoler les racines réelles de P.

- suites de Sturm : $P \to \frac{\partial P}{\partial x}$

```
> p := -85*x^5-55*x^4-37*x^3-35*x^2+97*x+50:
> readlib(sturm)(sturmseq(p, x), x, -infinity, infinity);
                                  3
```

- règle de Descartes : $P \to P(x+1)$

```
> readlib(realroot)(p);
              [[0, 2], [-1, 0], [-2, -1]]
```

Travaux antérieurs

Vincent, 1836 : « réciproque » de la règle de Descartes

Uspensky, 1948 : premier algorithme basé sur la règle de Descartes

Collins, Akritas, 1976 : premier algorithme effectif

Krandick, 1996 : variante en largeur d'abord

Johnson, Krandick, 1997 : algorithme hybride

Règle de Descartes

$P = a_n x^n + \cdots + a_1 x + a_0$,

$\sigma(P)$: nombre de racines positives (avec multiplicités) de P,

$\mathrm{Var}(P)$: nombre de changements de signe dans $a_n, \ldots, a_1, a_0$:

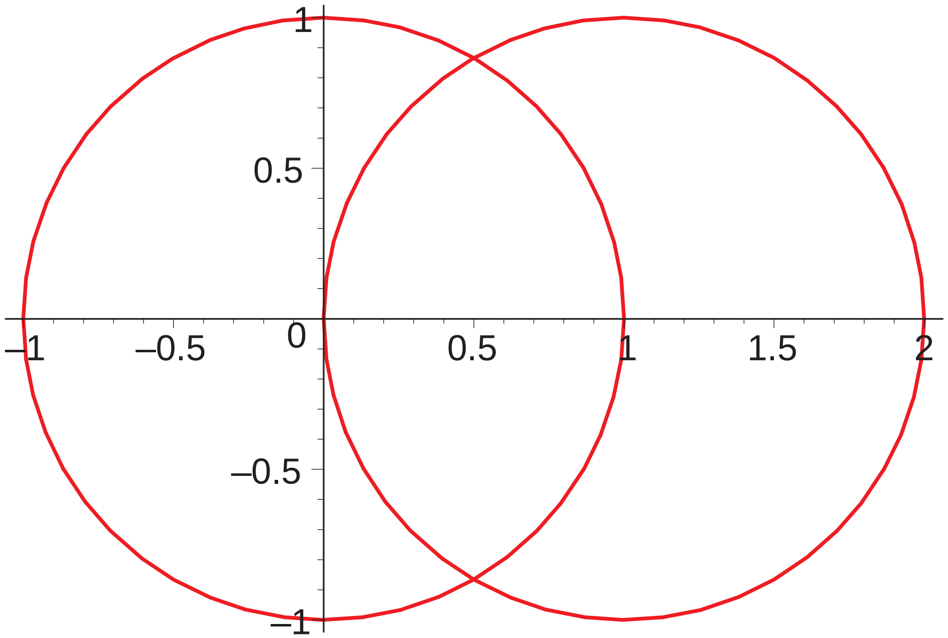
$$\sigma(P) \leq \mathrm{Var}(P).$$

Corollaire :

$$\sigma_{]0,1[}(P) \leq \mathrm{Var}(P(\frac{1}{1+x})).$$

Remarque : $\sigma(\cdot)$ et $\mathrm{Var}(\cdot)$ ayant même parité, il y a égalité lorsque $\mathrm{Var}(\cdot)$ égale 0 ou 1.

Collins/Johnson (1989) : si P a exactement une racine $0 < \alpha < 1$ et aucune autre dans les disques de rayon 1 centrés en $(0, 0)$ et $(1, 0)$, alors $\mathrm{Var} P(\frac{1}{1+x}) = 1$.



Notre contribution

F. Rouillier, P. Z., *Efficient isolation of a polynomial real roots*, Rapport de recherche INRIA 4113, 2001.

- version unifiée des variantes de Collins/Akritas et Krandick
- version « optimale » en mémoire constante

Algorithme d'Uspensky générique

Entrée : $P \in \mathbb{Z}[x]$ sans facteur carré, $<$ un ordre sur $\mathbb{Z}^2$

Sortie : liste d'intervalles d'isolation des racines réelles de P

$I \leftarrow \{\}, \quad T \leftarrow \texttt{initializeTree}(P)$

while $T \neq \{\}$ **do**

  $(k, c, Q) \leftarrow \texttt{getNewNode}(T, <)$

  $s \leftarrow \texttt{DescartesBound}(Q)$

  if $s = 1$ **then** $I \leftarrow I \cup \{(k, c)\}$

  elif $s > 1$ **then** $T \leftarrow T \cup \texttt{addSuccessors}((k, c), Q)$

  end if

end while

Return I

Collins/Akritas : ordre $<_{\mathrm{back}}$ (en profondeur d'abord)

`initializeTree` : $\{(0, 0, P)\}$

`getNewNode` : trivial

`addSuccessors` :

calculer $L = 2^n Q(x/2)$, $R = L(x+1)$,

ajouter $(k+1, 2c, L)$ et $(k+1, 2c+1, R)$.

Krandick : idem Collins/Akritas avec $<_{lev}$ (en largeur d'abord)

Rouillier/Z. : ordre quelconque $<$

`initializeTree` : $\{(0, 0)\}$

`getNewNode` : calculer $Q = P(\frac{x+c}{2^k})$

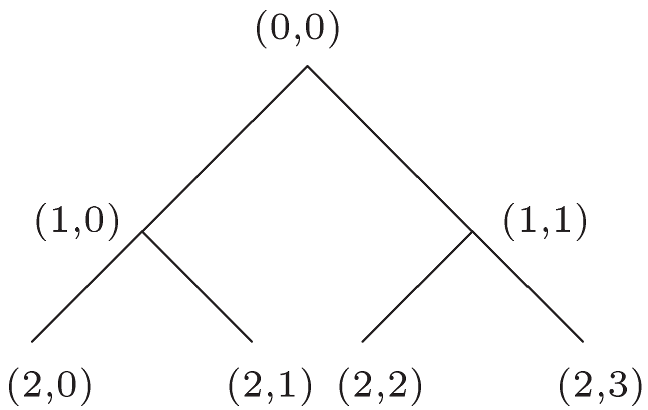
`addSuccessors` :

ajouter $(k+1, 2c)$ et $(k+1, 2c+1)$.

Idée : Au lieu de recalculer $P(\frac{x+c}{2^k})$ depuis le début, on repart du dernier nœud (k', c') :

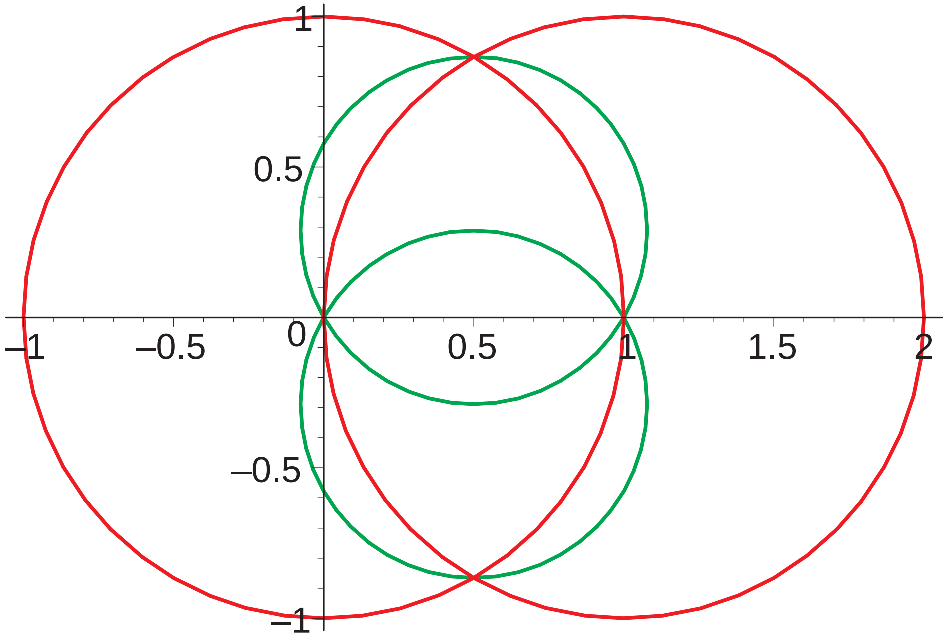
`getNewNodeRel` : $Q \leftarrow H_{2^{k'-k}} T_{2^{k'-k}c-c'}(Q')$

Dans l'algorithme **Rouillier/Z.** avec l'ordre $<_{\text{back}}$, toutes les translations sont T_0 ou T_1.



Implantation en RS : polynômes de degré 1000 en quelques minutes

Mehlhorn, 2001 : résultat de Collins/Johnson restreint aux disques de rayon $\frac{1}{\sqrt{3}}$ centrés en $(\frac{1}{2}, \frac{\pm 1}{2\sqrt{3}})$.



Perspectives

- version hybride en cours (Rouillier) : quelques centaines de bits suffisent, gain jusqu'à un facteur 10
- extension à $\mathbb{Q}(\alpha)[x]$ pour α algébrique réel (Cotterill)
- traduction dans la base des polynômes de Bernstein $B_k^n(x) = \binom{n}{k} x^k (1-x)^{n-k}$, où $B_k^n(x+1) = B_{n-k}^n(-x)$

Le produit médian

Motivation initiale : Itération de Newton pour $1/A$.

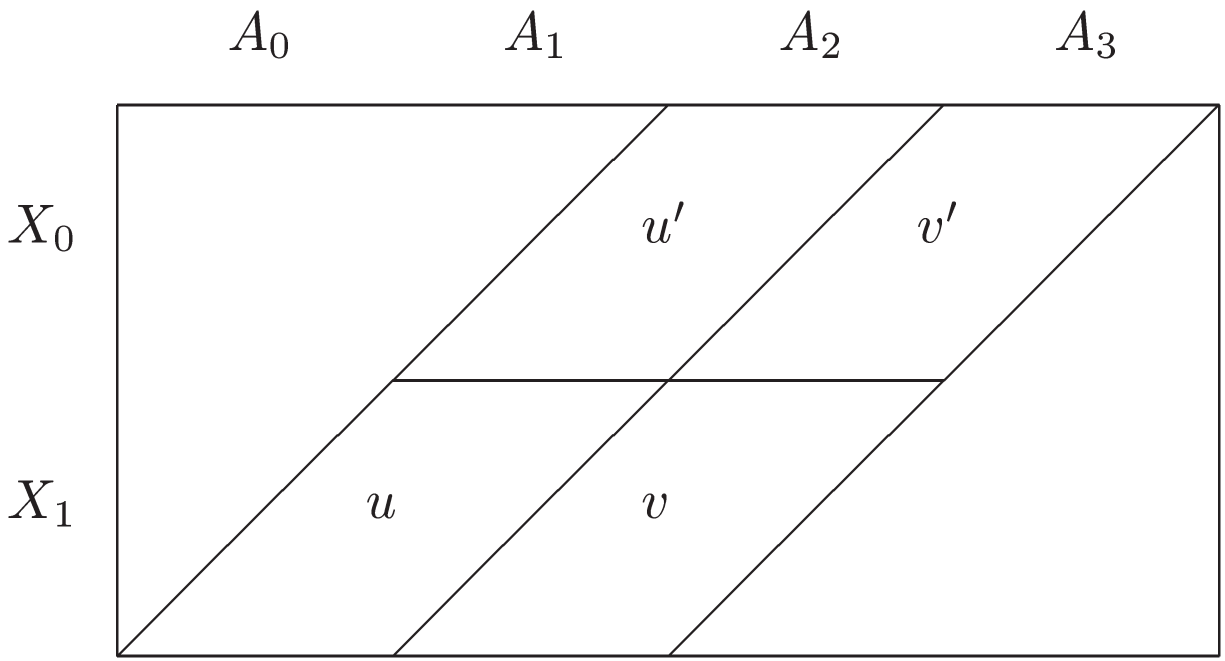
$$x_{k+1} = x_k + x_k(1 - Ax_k)$$

Étant donné un polynôme A de degré $2n-2$, et x de degré $n-1$, comment calculer efficacement les n coefficients médians de $A \cdot x$?

```
> A := a0 + a1*t + a2*t^2:
> x := x0 + x1*t:
> collect(A*x, t);
                3                           2
      a2 x1 t  + (a1 x1 + a2 x0) t  + (a0 x1 + a1 x0) t + a0 x0
```

Méthode naïve : n^2 ou $2M(n)$.

Applications : calcul de polynôme minimal (V. Shoup, ISSAC'99)



$$\mathrm{MP}([X_0, X_1], [A_0, A_1, A_2, A_3]) = [u + u', v + v'] =$$
$$[\mathrm{MP}(X_1, [A_0, A_1]) + \mathrm{MP}(X_0, [A_1, A_2]),$$
$$\mathrm{MP}(X_1, [A_1, A_2]) + \mathrm{MP}(X_0, [A_2, A_3])]$$

Astuce de base

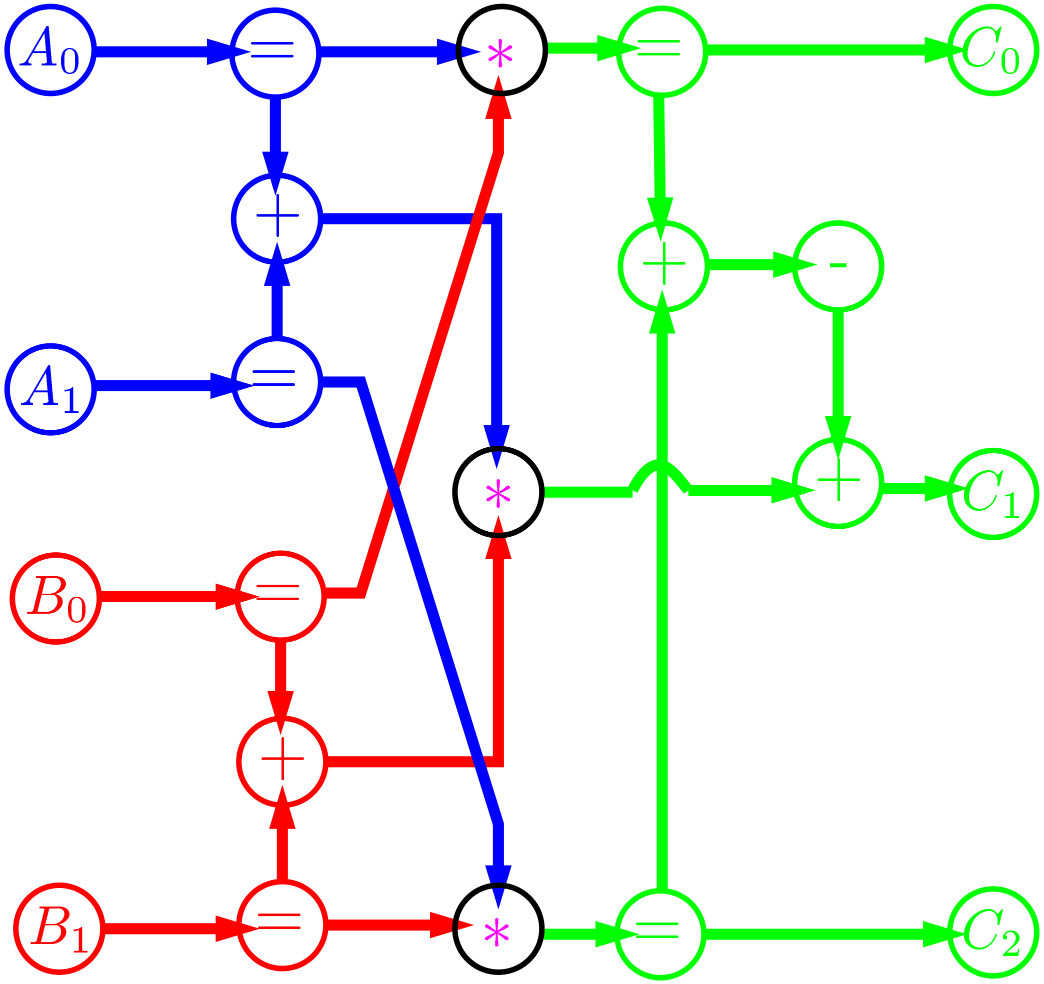
```
> alpha := (a0 + a1) * x1:
> beta  := (a1 + a2) * x0:
> gamma := a1 * (x1 - x0):
> collect((beta + gamma) * t^2 + (alpha - gamma) * t, t,
          expand);
                               2
          (a1 x1 + a2 x0) t  + (a0 x1 + a1 x0) t
```

Notre contribution

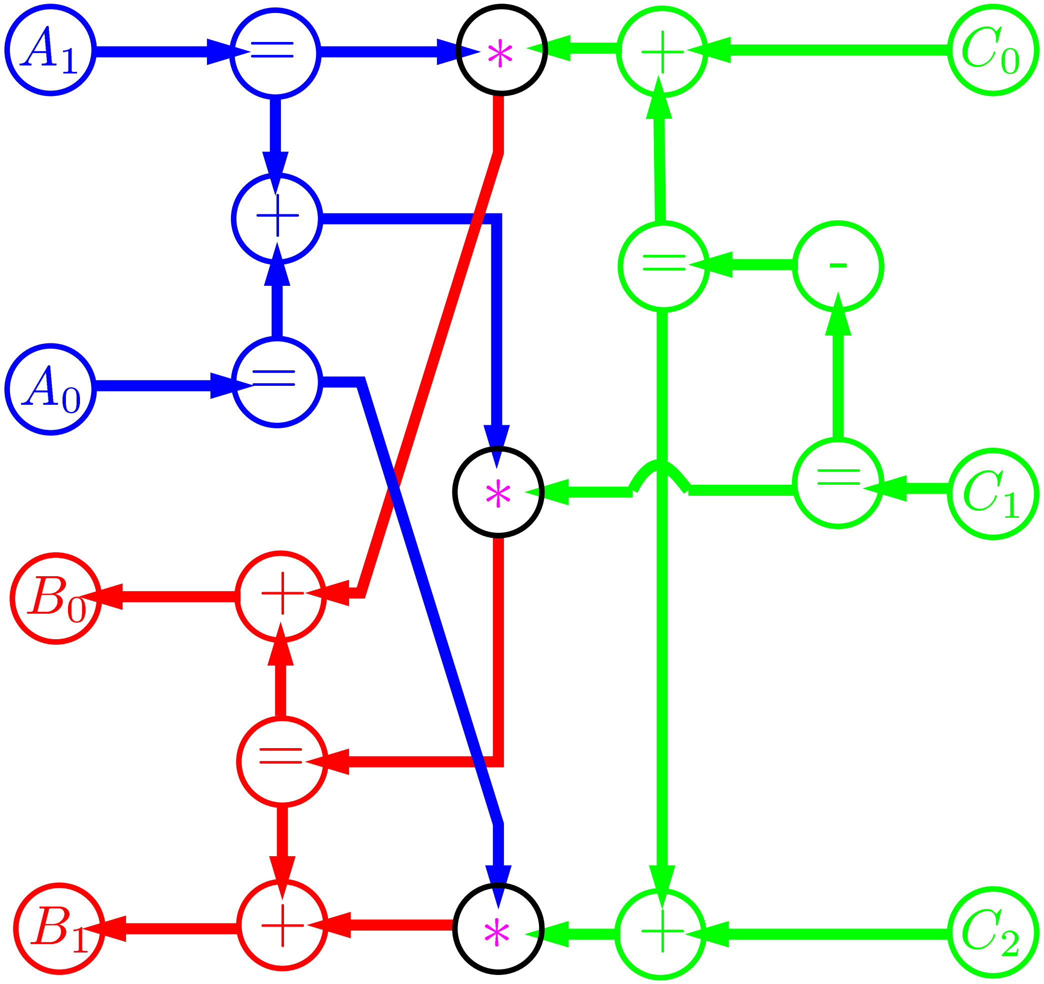
G. Hanrot, M. Quercia, P. Z., *The Middle Product Algorithm (or how to speed up the division and square root of power series)*, en cours de révision pour AAECC.

- $MP(n) \sim M(n)$
- pour Karatsuba, $I(n) \sim D^*(n) \sim K(n)$, $S^*(n) \sim \frac{3}{4}K(n)$, $D(n) \sim \frac{5}{3}K(n)$, $S(n) \sim \frac{5}{4}K(n)$
- pour FFT, $I(n) \sim 2M(n)$, $D^*(n) \sim \frac{5}{2}M(n)$, $S^*(n) \sim 3M(n)$, $D(n) \sim \frac{7}{2}M(n)$, $S(n) \sim 4M(n)$

Karatsuba : $P_0 = A_0 \cdot B_0$, $P_1 = (A_0 + A_1) \cdot (B_0 + B_1)$, $P_2 = A_1 \cdot B_1$, $C_0 = P_0$, $C_1 = P_1 - (P_0 + P_2)$, $C_2 = P_2$.



MP-Karatsuba : $P_0 = A_1 \cdot (C_0 - C_1)$, $P_1 = (A_0 + A_1) \cdot C_1$, $P_2 = A_0 \cdot (C_2 - C_1)$, $B_0 = P_0 + P_1$, $B_1 = P_1 + P_2$.



Perspectives

- application immédiate aux polynômes et séries formelles : gain d'un facteur 2 constaté pour la fonction `Quo` de Magma
- module Maple `MP`
- $D^*(n) \sim M(n)$?
- $D(n) \sim M(n)$?

La bibliothèque MPFR

(D. Daney, G. Hanrot, V. Lefèvre, F. Rouillier)

```
> e := 892*ln(2) - 163*Pi - 184*gamma:
> evalf(e, 1);
                                   -100.

> evalf(e, 2);
                                   10.

> evalf(e, 3);
                                   0.

> evalf(e, 4);
                                   -.1

> evalf(e, 20);
                                               -6
                              .18245287177 10
```

```
    |\^/|     Maple 7 (IBM INTEL LINUX)
._|\|   |/|_. Copyright (c) 2001 by Waterloo Maple Inc.
 \  MAPLE  /  All rights reserved. Maple is a registered trademark of
 <____ ____>  Waterloo Maple Inc.
      |       Type ? for help.
> ?Rounding
 Rounding Mode                Meaning

   nearest      round to the nearest possible value,
                with ties resolved by rounding to the nearest even
      0         round towards zero
  infinity      round towards positive infinity
 -infinity      round towards negative infinity
> Digits:=3:
> a:=1.0: b:=9e-5: Rounding:=0: a-b;
                                              1.0
```

Objectif : calcul flottant efficace en précision arbitraire (comme dans le programme MP de Richard Brent) avec arrondi « exact ».

Basée sur GMP : portabilité et efficacité.

Arrondi exact : à la main pour les opérations de base, calcul précis d'erreur de troncature et d'arrondi pour les fonctions mathématiques.

Effets de bord : travaux de fond ayant mené au « produit médian », à la racine carrée récursive en $\frac{3}{2}K(n)$, aux implantations efficaces de division et racine carrée récursives, prouvées par Didier Bondyfalat et Nicolas Magaud (INRIA Sophia).

MPFI (Revol, Rouillier) : arithmétique d'intervalle avec MPFR

Calcul de $892 \log 2 - 163\pi - 184\gamma$ avec MPFI :

```
$ ./mpfi_ex 1
[-3.e3, 2.e3]
$ ./mpfi_ex 2
[-8.e2, 4.e2]
$ ./mpfi_ex 3
[-5.e2, 3.e2]
$ ./mpfi_ex 4
[-2.e2, 7.e1]
$ ./mpfi_ex 60
[1.82452870478577722e-7, 1.82452873365157587e-7]
```

Perspectives

- outils `gnumeric`, `ndiff`
- nombreuses applications en calcul hybride
- prototypage d'algorithmes d'arrondi exact en précision fixe
- projet DLMF (*Digital Library of Mathematical Functions*)
- interface avec un système de calcul formel ?

Perspectives générales

- sémantique des calculs flottants (standard ?)
- algorithmes hybrides
- mathématiques « expérimentales »