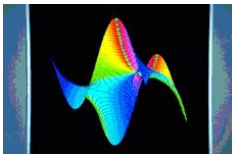


# Composition of Power Series, Change of Basis and Orthogonal Polynomials

Bruno Salvy

Bruno.Salvy@inria.fr

Algorithms Project, Inria



June 2nd, 2008

Joint work with Alin Bostan and Éric Schost

arXiv:0804.2337 (ISSAC'08) and arXiv:0804.2373.

# I Introduction

# Univariate Composition

## Problem (Composition of Power Series)

**Input:** two power series  $f$  and  $g$  at precision  $N$ , with  $g(0) = 0$ .

**Output:**  $f(g)$  at precision  $N$ .

**Measure of complexity:** number of arithmetic operations.

Hypothesis for the talk: characteristic 0.

**General algorithms:**

- Naïve:  $\mathcal{O}(N^3)$ ;
- Naïve with fast multiplication:  $\mathcal{O}(NM(N))$ ;
- Brent & Kung (1978):  $\mathcal{O}(\sqrt{N \log N} M(N))$ .

## Fast Multiplication of Polynomials of degree $N$

- Naïve:  $M(N) = \mathcal{O}(N^2)$ ;
- Karatsuba (1963):  $M(N) = \mathcal{O}(N^{1.59})$ ;
- FFT:  $M(N) = \mathcal{O}(N \log N)$  (Schönhage & Strassen 71).

Nothing better known in general → **exploit structure**

# Univariate Composition

General algorithms:  $\mathcal{O}(\sqrt{N} \log N M(N))$ ;

Special  $f$ 's by Newton iteration:  $\mathcal{O}(M(N))$  for inverse (Sieveking 72, Kung 74), log, exp, power (Brent 75);

Special  $g$ 's by Divide-and-Conquer:  $\mathcal{O}(M(N) \log N)$ ;

- polynomials (Brent & Kung 78);
- algebraic series (van der Hoeven 02);

$f$  a polynomial, special  $g$ 's:

- $x + a$ :  $\mathcal{O}(M(N))$  (Aho, Steiglitz, Ullman 75);
- $(ax + b)/(cx + d)$ :  $\mathcal{O}(M(N))$  (Pan 98);
- **NEW**: several other functions in  $\mathcal{O}(M(N))$ ;
- **NEW**: **exp and log** in  $\mathcal{O}(M(N) \log N)$ .

# Bivariate Evaluation

## Problem (Bivariate Evaluation)

$$F(x, t) = \sum_{i,j} F_{i,j} x^i t^j = \sum_j G_j(x) t^j.$$

Input:  $a_0, \dots, a_N$ ;

Output:  $\sum_j a_j G_j(x) \bmod x^{N+1}$ .

### Examples:

- $G_j = g^j$ : univariate composition;
- $G_j$  polynomial of degree  $j$ : change of basis from  $(G_j)$  to  $(x^j)$ .

## Our Result

Good complexity for this map **and its inverse** for “nice”  $F$ .

# Special Cases with Fast Change of Basis

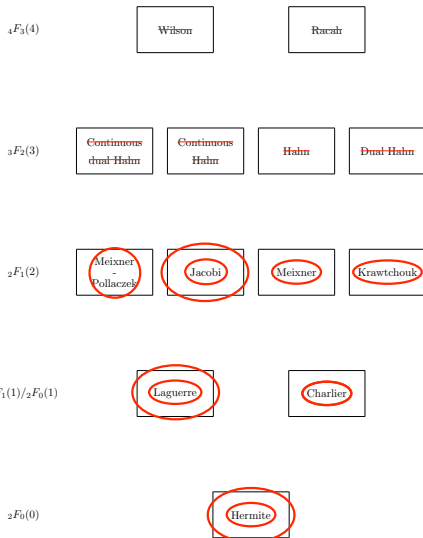
## Very fast ( $\mathcal{O}(M(N))$ )

- Orthogonal: **Jacobi**, Legendre, Chebyshev  $U$ ,  $T$ , Gegenbauer, Laguerre, Hermite.
- Other: **Fibonacci**, Bernoulli, Euler, Mott, **Spread**, Bessel, . . .

## Fast ( $\mathcal{O}(M(N) \log N)$ )

- Orthogonal: Meixner, Meixner-Pollaczek, Krawtchouk, Charlier;
- Other: Falling factorial, Bell, Actuarial, Narumi, Peters, . . .

**Not all** of Sheffer type.



Other algorithm giving all orthogonal polys in  $\mathcal{O}(M(N) \log N)$

## II Very Fast Composition ( $\mathcal{O}(M(N))$ operations)

# Shift is Cheap (Aho, Steiglitz, Ullman 1975)

## Problem (Polynomial Shift)

**Input:**  $P(x)$  polynomial of degree  $N$ ,  $a$  a point.

**Output:**  $P(x + a)$ .

Horner:  $\mathcal{O}(N^2)$  operations

Taylor:

$$P^{(N-i)}(a) = \sum_{j=0}^i P^{(N-i+j)}(0) \frac{a^j}{j!}.$$

Consequence: generating series

$$\sum_{i=0}^N \underbrace{P^{(N-i)}(a)}_{\text{coeffs of } P(x+a)} x^i + \dots = \sum_{k=0}^n \underbrace{P^{(k)}(0)}_{\text{coeffs of } P(x)} x^{N-k} \times \sum_{j \geq 0} \frac{a^j x^j}{j!}.$$

$M(N) + \mathcal{O}(N)$  operations.





## Warm-up: Euler Transform for 3 Multiplications

## Problem (Euler Transform)

**Input:**  $P(x)$  polynomial of degree  $N$  (or truncated series);

**Output:** First  $N$  coefficients of  $\frac{1}{1-x}P\left(\frac{x}{1-x}\right)$ .

**Algorithm:** »

$$S_1(x) := P(x-1); \quad M(N) + \mathcal{O}(N) \text{ ops;}$$

$$S_2(x) := x^N S_1(1/x); \quad = x^N P\left(\frac{1-x}{x}\right) \quad 0 \text{ ops;}$$

$$S_3(x) := S_2(x+1); \quad = (1+x)^N P\left(\frac{-x}{1+x}\right) \quad M(N) + \mathcal{O}(N) \text{ ops;}$$

$$S_4(x) := S_3(-x); \quad = (1-x)^N P\left(\frac{x}{1-x}\right) \quad \mathcal{O}(N) \text{ ops;}$$

$$\text{return } (1-x)^{-N-1} S_4(x). \quad M(N) + \mathcal{O}(N) \text{ ops.}$$

## More Systematically

## Collection of operators on series...

$$A_a : g \mapsto a + g$$

$$M_\lambda : g \mapsto \lambda g$$

$$\text{Inv} : g \mapsto 1/g$$

$$P_k : g \mapsto g^k$$

$$R_k : g \mapsto g^{1/k}$$

## ...and associativity rules

$$f(A_a(g)) = f|_{x \mapsto x+a} \circ g$$

$$f(M_\lambda(g)) = f|_{x \mapsto \lambda x} \circ g$$

$$f(\text{Inv}(g)) = g^{\deg f} \text{Rec}(f) \circ g$$

$$f(P_k(g)) = f|_{x \mapsto x^k} \circ g$$

$$f(R_k(g)) = f|_{x \mapsto x^{1/k}} \circ g$$

$$x \mapsto \frac{x}{1-x} \Rightarrow \frac{1}{1-x} - 1 \text{ decomposes as } A_{-1} \circ \text{Inv} \circ A_1 \circ M_{-1}.$$

## More Systematically

## Collection of operators on series...

$$A_a : g \mapsto a + g$$

$$M_\lambda : g \mapsto \lambda g$$

$$\text{Inv} : g \mapsto 1/g$$

$$P_k : g \mapsto g^k$$

$$R_k : g \mapsto g^{1/k}.$$

## ...and associativity rules

$$f(A_a(g)) = f|_{x \mapsto x+a} \circ g$$

$$f(M_\lambda(g)) = f|_{x \mapsto \lambda x} \circ g$$

$$f(\text{Inv}(g)) = g^{\deg f} \text{Rec}(f) \circ g$$

$$f(P_k(g)) = f|_{x \mapsto x^k} \circ g$$

$$f(R_k(g)) = f|_{x \mapsto x^{1/k}} \circ g$$

$$x \mapsto \frac{x}{1-x} = \frac{1}{1-x} - 1 \text{ decomposes as } A_{-1} \circ \text{Inv} \circ A_1 \circ M_{-1}.$$

$$\text{Algorithm: } f\left(\frac{x}{1-x}\right) = \text{Rec}\left(f|_{x \mapsto x-1}\right) \Big|_{x \mapsto x+1} \Big|_{x \mapsto -x} \times (A_1 \circ M_{-1}(x))^N.$$

Check

# More Systematically

## Collection of operators on series...

$$A_a : g \mapsto a + g$$

$$M_\lambda : g \mapsto \lambda g$$

$$\text{Inv} : g \mapsto 1/g$$

$$P_k : g \mapsto g^k$$

$$R_k : g \mapsto g^{1/k}.$$

## ...and associativity rules

$$f(A_a(g)) = f|_{x \mapsto x+a} \circ g$$

$$f(M_\lambda(g)) = f|_{x \mapsto \lambda x} \circ g$$

$$f(\text{Inv}(g)) = g^{\deg f} \text{Rec}(f) \circ g$$

$$f(P_k(g)) = f|_{x \mapsto x^k} \circ g$$

$$f(R_k(g)) = f|_{x \mapsto x^{1/k}} \circ g$$

## Theorem

For any well-defined  $g$  composition of  $\{A_a, M_\lambda, \text{Inv}, P_k, R_k\}$  and any polynomial  $f$ ,  $f \circ g \bmod x^N$  computed in  $\mathcal{O}(M(N))$  ops.

**Proof.** Fast right and left compositions.

What can we  
construct  
with  
 $\{A_a, M_\lambda, \text{Inv}, P_k, R_k\}$ ?



# Polynomials & Rational Functions

## Proposition (Degree 2)

All polynomials and rational functions of degree 2, and their (compositional) inverses.

- ①  $ax + b$ : easy;
- ②  $ax^2 + bx + c = (b - a^2/4) + (x + a/2)^2$  ;
- ③  $(ax + b)/(cx + d) = a/c + (b - ad/c)/(cx + d)$  (Pan, 98);
- ④  $x/(x^2 - d) = \frac{1}{2\sqrt{d}} \sqrt{\left(\frac{2d}{d-x^2} - 1\right)^2 - 1}$ ;
- ⑤  $(x^2 - d)/x = \textcircled{4}^{-1}$ ;
- ⑥  $(x^2 + ax - d)/x = a + \textcircled{5}$ ;
- ⑦  $(x^2 + ax + b)/(x + c) = \textcircled{6} \Big|_{x \mapsto x+c, d \mapsto ac-b}$ ;
- ⑧  $(x + c)/(x^2 + ax + b) = \textcircled{7}^{-1}$ ;
- ⑨  $(x^2 + dx + e)/(x^2 + ax + b) = 1 + (d - a) \cdot \textcircled{8} \Big|_{c \mapsto (e-b)/(d-a)}$ ;
- ⑩  $(x^2 + ax + e)/(x^2 + ax + b) = 1 + (e - b) \cdot \textcircled{2} \Big|_{a \mapsto 1, b \mapsto a, c \mapsto b}$ .

# Sequence Transformations

## Problem (Binomial Convolution)

**Input:** sequence  $(a_n)$ ,  $n = 0, \dots, N$ ;

**Output:** sequence  $(b_n)$ ,  $n = 0, \dots, N$ , where

$$b_n = \sum_{k \geq 0} \binom{n + pk}{m + qk} a_k.$$

Generating series

$$\sum_n b_n z^n = \frac{z^m}{(1-z)^{m+1}} A\left(\frac{z^{p-q}}{(1-z)^p}\right).$$

Algorithm in  $\mathcal{O}(M(N))$  for

- $p = 1, q = 0$  (Euler);
- $p = 2$  and  $q = 0$  or  $q = 1$ .

## III Fast Composition ( $\mathcal{O}(M(n)\log n)$ operations)



# Result

## Theorem

For any well-defined  $g$  composition of  $\{A_a, M_\lambda, \text{Inv}, P_k, R_k, \text{polynomial}, \text{algebraic function}, \text{exp}, \text{log}\}$  and any polynomial  $f$ ,  $f \circ g \bmod x^N$  computed in  $\mathcal{O}(M(N) \log N)$  ops.

## Proof.

- Fast left compositions  
(Newton, Brent & Kung);
- Fast right compositions:
  - polynomials: Brent & Kung (78);
  - algebraic functions: van der Hoeven (02);
  - exp, log: next screens.

# Exponential

## Problem (Right Composition with Exp)

**Input:**  $(a_0, \dots, a_N)$

**Output:** coefficients of  $\sum_j a_j e^{jx} \bmod x^{N+1}$ .

Laplace:  $\mathcal{L}(x^i) := i!x^i \Rightarrow \sum_{j=0}^N a_j e^{jx} = \mathcal{L}^{-1} \underbrace{\sum_{j=0}^N \frac{a_j}{1 - jx}}_{:= S_0^N}$ .



## Transposed Multipoint Evaluation (Bostan, Lecerf, Schost 03)

- 1 Divide and conquer: numerator and denominator of  $S_0^N$  from those of  $S_0^{N/2}$  and  $S_{N/2+1}^N$ ;  $\frac{3}{2} M(N) \log N + \mathcal{O}(M(N))$  ops
- 2 Compute the series expansion of the quotient  $\mathcal{O}(M(N))$  ops
- 3 Multiply the  $i$ th coeff by  $1/i!$ ,  $i = 0, \dots, N$   $\mathcal{O}(N)$  ops

# Matrices Acting on the Vector of Coefficients

Free ( $\mathcal{O}(n)$ )

- Rec: permutation  $i \mapsto N - i$ ;

Super fast ( $\mathcal{O}(N)$ ):

- $\text{Diag}(u_i)$ , e.g.,  $M_\lambda(u_i = \lambda^i)$ ;  $\mathcal{L}(u_i = i!)$ ;  $\mathcal{L}^{-1}(u_i = 1/i!)$ ;

Very fast ( $\mathcal{O}(M(N))$ ):

- Multiplication by a fixed  $P$ :  $\text{Mul}(P)$ ;
- Shift  $A_a$ :  $\mathcal{L}^{-1} \cdot \text{Rec} \cdot \text{Mul}(e^{ax}) \cdot \text{Rec} \cdot \mathcal{L}$ ;

Check

Fast ( $\mathcal{O}(M(N) \log N)$ ):

- Multipoint evaluation:  $V(a_0, \dots, a_N)$  (Vandermonde);
- Interpolation:  $V(a_0, \dots, a_N)^{-1}$ ;
- Exp:  $\mathcal{L}^{-1} \cdot V(0, \dots, N)^t$ ;
- $\log(1+x)$ :  $A_1 \cdot (V(0, \dots, N)^{-1})^t \cdot \mathcal{L}$ . Invert  $x \mapsto e^x - 1!$

Check

Slow ( $\mathcal{O}(\sqrt{N} \log N M(N))$ )

- Composition with  $g$ :  $(g_{i,k})$ , with  $g^k = \sum g_{i,k} x^i \bmod x^N$ .

# $\log(1+x)$ and Transposed Interpolation

$$P(x) \xrightarrow{A_{-1}} \sum_{j=0}^N a_j x^j \xrightarrow{\mathcal{L}} \sum_{j=0}^N \frac{b_j}{1-jx} \xrightarrow{\text{MEval}^t} \sum_{j=0}^N c_j x^j \xrightarrow{\mathcal{L}^{-1}} P(e^x - 1) + O(x^{N+1}).$$

## Problem: Transposed Interpolation

**Input:**  $C(x)$  polynomial of degree  $N$

**Output:**  $(b_0, \dots, b_N)$  such that  $C(x) = \sum_{j=0}^N \frac{b_j}{1-jx} + O(x^{N+1})$ .

# log(1 + x) and Transposed Interpolation

## Problem: Transposed Interpolation

**Input:**  $C(x)$  polynomial of degree  $N$

**Output:**  $(b_0, \dots, b_N)$  such that  $C(x) = \sum_{j=0}^N \frac{b_j}{1-jx} + O(x^{N+1})$ .

Let  $P(x) := \prod_{j=0}^N (X - j)$ , then

$$\frac{1}{x} C\left(\frac{1}{x}\right) = \sum_{j=0}^N \frac{b_j}{x-j} = \frac{Q(x)}{P(x)} + O(x^{-N-2}) \Rightarrow b_j = \frac{Q(j)}{P'(j)}.$$

## Algorithm (Bostan, Lecerf, Schost 03) with linear rec for $P'$

- 1 Expand  $P$  by divide-and-conquer;  $\frac{1}{2} M(N) \log N + O(N)$
- 2 Compute  $C \times \text{Rec}(P) = \text{Rec}(Q) + O(x^{N+1})$ ;  $M(N) + O(N)$
- 3 Evaluate  $Q$  and  $P'$  at  $0, \dots, N$ ;  $M(N) \log N + O(N)$
- 4 Return  $(Q(0)/P'(0), \dots, Q(N)/P'(N))$ .  $O(N)$ .

## IV Change of Basis

# Composition Sequences

## Definition (Composition Sequence)

A map  $F = o_1 \circ \cdots \circ o_m$  with  $o_i$ 's in  $\{A_a, M_\lambda, P_k, R_k, \text{Inv}, \text{exp}, \text{log}\}$ .  
It *defines*  $F(x)$  if all intermediates  $o_i \circ \cdots \circ o_m(x)$  are power series.

## Summary so far

$P$  a polynomial,  $F$  a composition sequence defining  $F(x)$ , then

- $F(x) \bmod x^N$ ;
- $P(F(x)) \bmod x^N$

can be computed in a number of operations

$$T_F(N) = \begin{cases} \mathcal{O}(M(N)) & \text{if no } o_i \text{ is exp or log,} \\ \mathcal{O}(M(N) \log N) & \text{otherwise.} \end{cases}$$

# Main Theorem

Bivariate Evaluation with  $F(x, t) = \sum F_{ij}x^i t^j = \sum G_j(x)t^j$

Input:  $(a_0, \dots, a_N)$ ;

Output:  $\sum_{i=0}^N b_i x^i = \sum a_j G_j(x) \bmod x^{N+1}$ .

## Main Theorem of (Bostan, S., Schost, 08)

Generating function  $F(x, t) = u(x) f(g(x) \times h(t)) v(t)$  with

- $g$  and  $h$  given by composition sequences;
- $u, v, f$  given by their coefficients  $\bmod x^{N+1}$ ;
- well-defined:  $g(0)h(0) = 0$ ;
- $g'(0)h'(0)u(0)v(0) \neq 0$ , all coefficients of  $f \neq 0$ ,

then  $(a_i) \mapsto (b_i)$  in  $\mathcal{O}(T_g(N) + T_h(N) + M(N))$  ops

then  $(b_i) \mapsto (a_i)$  in  $\mathcal{O}(T_g(N) + T_h(N) + M(N))$  ops.



Examples in  $\mathcal{O}(M(N))$ Sheffer Sequences:  $F = \exp(xh(t))v(t)$ 

Name	Notation	$h$	$v$
Laguerre	$L_n^\alpha$	$t(1-t)^{-1}$	$(1-t)^{-1-\alpha}$
Hermite	$H_n$	$2t$	$\exp(-t^2)$
Euler	$E_n^\alpha$	$t$	$2^\alpha(1+e^t)^{-\alpha}$
Bernoulli	$B_n^\alpha$	$t$	$t^\alpha(1+e^t)^{-\alpha}$
Mott	$M_n$	$(\sqrt{1-t^2}-1)/t$	1
Bessel	$p_n$	$1-\sqrt{1-2t}$	1

Examples in  $\mathcal{O}(M(N))$ 

Non-Sheffer:

- Fibonacci

$$\sum_{n \geq 0} F_n(x) t^n = \frac{1}{1-t^2} \frac{1}{1-\frac{xt}{1-t^2}};$$

- Chebyshev

$$-\frac{1}{2} + \sum_{n \geq 0} T_n(x) t^n = \frac{1-t^2}{2(1+t^2)} \frac{1}{1-\frac{2xt}{1+t^2}};$$

- Jacobi with  $(\alpha + \beta + 1) \neq 0$

$$\sum_{n \geq 0} \frac{(\alpha + \beta + 1)_n}{(\beta + 1)_n} P_n^{(\alpha, \beta)}(x) t^n =$$

$$(1+t)^{-\alpha-\beta-1} {}_2F_1 \left( \begin{matrix} \frac{\alpha+\beta+1}{2}, \frac{\alpha+\beta+1}{2} \\ \beta+1 \end{matrix} \middle| \frac{2t(1+x)}{(1+t)^2} \right).$$

Other Sheffer Sequences, but in  $\mathcal{O}(M(N) \log N)$ 

Name	Notation	$h$	$v$
Falling factorial	$(x)_n$	$\log(1+t)$	1
Bell	$\phi_n$	$\exp(t) - 1$	1
Bernoulli, 2nd kind	$b_n$	$\log(1+t)$	$t / \log(1+t)$
Poisson-Charlier	$c_n(x; a)$	$\log(1+t/a)$	$\exp(-t)$
Actuarial	$a_n^{(\beta)}$	$1 - \exp(t)$	$\exp(\beta t)$
Narumi	$N_n^{(a)}$	$\log(1+t)$	$t^a \log(1+t)^{-a}$
Peters	$P_n^{(\lambda, \mu)}$	$\log(1+t)$	$(1 + (1+t)^\lambda)^{-\mu}$
Meixner-Pollaczek	$P_n^{(\lambda)}(x; \phi)$	$i \log\left(\frac{1-te^{i\phi}}{1-te^{-i\phi}}\right)$	$(1 + t^2 - 2t \cos \phi)^{-\lambda}$
Meixner	$m_n(x; \beta, c)$	$\log\left(\frac{1-t/c}{1-t}\right)$	$(1-t)^{-\beta}$
Krawtchouk	$K_n(x; p, N)$	$\log\left(\frac{p-(1-p)t}{p(1+t)}\right)$	$(1+t)^N$

Falling factorial known (Gerhard 00);

Last 3 known in direct sense ( $P_i$ )  $\mapsto (x^i)$

(Driscoll, Healy, Rockmore 97; Pott, Steidl, Tasche 98).

# Algorithm

Bivariate Eval,  $u(x)f(g(x)h(t))v(t) = \sum F_{ij}x^i t^j = \sum G_j(x)t^j$

**Input:**  $(a_0, \dots, a_N)$ ; composition seq. for  $g$  and  $h$ ; series for  $f, u, v$

**Output:**  $\sum_{i=0}^N b_i x^i = \sum a_j G_j(x) \bmod x^{N+1}$ .

$$[x^i t^j]f(g(x)h(t)) = F_{ij} = \sum_k f_k g_{ik} h_{jk}, \quad \text{where } \begin{cases} g^k &= \sum_i g_{ik} x^i, \\ h^k &= \sum_j h_{jk} x^j. \end{cases}$$

In matrix form,

$$(b) = (F_{ij})(a) = \mathbf{Mul}(u)(g_{ik}) \mathbf{Diag}(f)(h_{jk})^t \mathbf{Mul}(v)^t(a).$$

We know how to multiply by  $(g_{ik})$  (fast right composition) if we have fast multiplication by  $(h_{jk})^t$ ,  $\mathbf{Mul}(v)^t$  we're done.

Granted by Tellegen's principle.

Effective by code transposition (Bostan, Lecerf, Schost 03).

# Basic Transpositions

Free ( $\mathcal{O}(N)$ )

- Rec: permutation  $i \mapsto N - i$ ;

Transpose

Rec

Super fast ( $\mathcal{O}(N)$ ):

- Diag( $u$ ), e.g.,  $M_\lambda, \mathcal{L}, \mathcal{L}^{-1}$ ;

Diag( $u$ )

Very fast ( $\mathcal{O}(M(N))$ ):

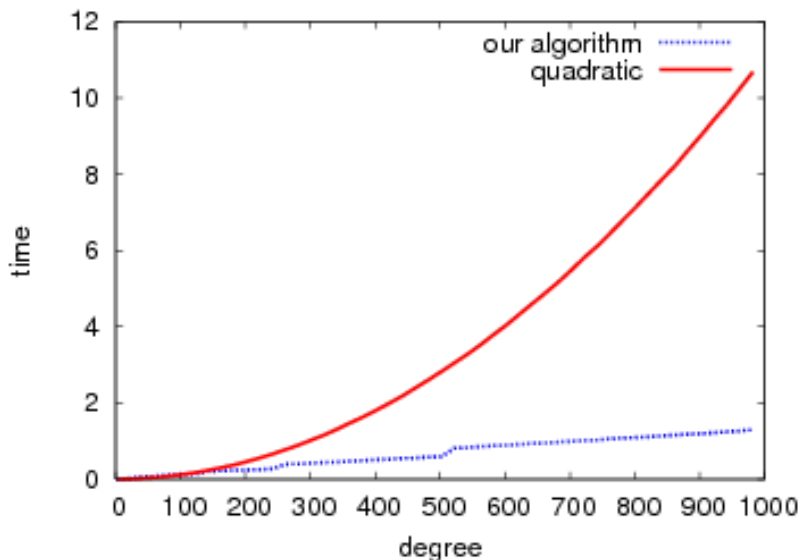
- Mul( $P$ );  $\text{Mul}(P)^t : A \mapsto A \times \text{Rec}(P) \text{ div } x^N$
- $A_a = \mathcal{L}^{-1} \text{RecMul}(e^{ax}) \text{Rec}\mathcal{L}$ ;  $\mathcal{L} \text{RecMul}(e^{ax})^t \text{Rec}\mathcal{L}^{-1}$

Fast ( $\mathcal{O}(M(N) \log N)$ ):

- Multipoint evaluation:  $V(a_0, \dots, a_N)$  (Vandermonde);
- Interpolation:  $V(a_0, \dots, a_N)^{-1}$ ;
- Exp:  $\mathcal{L}^{-1} \cdot V(0, \dots, N)^t$ ; Multipoint evaluation  $\cdot \mathcal{L}^{-1}$
- $\log(1+x)$ :  $A_1 \cdot (V(0, \dots, N)^{-1})^t \cdot \mathcal{L}$ .  $\mathcal{L} \cdot \text{Interpolation} \cdot A_1^t$

End of Proof

# Example: Conversion to Jacobi



## V General Orthogonal Polynomials

# Basic Facts on Orthogonal Polynomials

- ① A non-decreasing function  $\mu$  on  $(a, b)$ .
  - ② A scalar product  $\langle f, g \rangle = \ell(fg) = \int_a^b fg \, d\mu$  s.t.  $\forall n, \ell(x^n) < \infty$ .
  - ③ Gram-Schmidt for  $(x^n)$  produces  $(P_n)$  s.t.  $\deg P_n = n$  and if  $\deg q < n$ , then  $\langle P_n, q \rangle = 0$ .
  - ④ Therefore  $\langle xP_n | P_i \rangle = \langle P_n | xP_i \rangle = 0$  for  $i < n - 1$ , and
 

$$P_{n+1}(x) = (a_n x + b_n)P_n(x) + c_n P_{n-1}(x), \quad a_n c_n \neq 0.$$

(R)
- Starting from  $P_{-1} = 0, P_0 = 1$ , this defines the  $(P_n)$ .



Direct Conversion in  $\mathcal{O}(M(N) \log N)$  ops

$$\begin{pmatrix} P_{n+1} \\ P_n \end{pmatrix} = \underbrace{\begin{pmatrix} a_n x + b & c_n \\ 1 & 0 \end{pmatrix}}_{:= M^{(n-1,n)}} \begin{pmatrix} P_n \\ P_{n-1} \end{pmatrix} = M^{(n-1,n)} \dots M^{(0,1)} \begin{pmatrix} P_1 \\ P_0 \end{pmatrix}.$$

$$\text{Aim: } \sum_{i=0}^{2n+1} \alpha_i P_i = \sum_{i=0}^n (\alpha_{2i} \quad \alpha_{2i+1}) M^{(0,2i)} \begin{pmatrix} P_1 \\ P_0 \end{pmatrix} =: S^{(0,2n)} \begin{pmatrix} P_1 \\ P_0 \end{pmatrix}.$$

## Algorithm Direct Conversion

Input:  $(\alpha_{2m}, \dots, \alpha_{2p+1})$

Output:  $M^{(2m,2p)}$  and  $S^{(2m,2p)}$

- 1  $q := \lceil \frac{m+p}{2} \rceil$ ;
- 2 Compute recursively  $M^{(2m,2q)}$ ,  $S^{(2m,2q)}$ ,  $M^{(2q+2,2p)}$ ,  $S^{(2q+2,2p)}$ ;
- 3 Return  $M^{(2q+2,2p)} M^{(2m,2q)}$  and  $S^{(2m,2q)} + S^{(2q+2,2p)} M^{(2m,2q)}$ .

Note: This gives fast multiplication by  $\mathbf{P} := ([x^i] P_j)$ .

Inverse Conversion in  $\mathcal{O}(M(N) \log N)$  operations

$$P_{n+1}(x) = (a_n x + b_n)P_n(x) + c_n P_{n-1}(x), \quad a_n c_n \neq 0. \quad (R)$$

Starting from  $P_{-1} = 0$ ,  $P_0 = 1$ , this defines the  $(P_n)$ .

Theorem (Favard (35); Shohat (36))

*Conversely*, given  $(a_n), (b_n), (c_n)$  and  $(R)$ , there exists  $\ell$  such that

- $\ell(P_j P_i) = 0, j < i; d_i := \ell(P_i^2) = (-1)^i c_2 \cdots c_{i+1} / a_{i+1} \neq 0;$

- $L(x) := \sum_{i \geq 0} \ell(x^i) x^i = \frac{Q_{n-1}}{\text{Rec}(P_n)} + \mathcal{O}(x^{2n}),$  where

$$Q_n = (a_{n+1} + b_{n+1}x)Q_{n-1} + c_{n+1}x^2 Q_{n-2}, \quad Q_{-1} = 0, \quad Q_0 = 1.$$

Inverse Conversion in  $\mathcal{O}(M(N) \log N)$  operations

Theorem (Favard (35); Shohat (36))

*Conversely*, given  $(a_n), (b_n), (c_n)$  and  $(R)$ , there exists  $\ell$  such that

- $\ell(P_j P_i) = 0, j < i; d_i := \ell(P_i^2) = (-1)^i c_2 \cdots c_{i+1} / a_{i+1} \neq 0;$

- $L(x) := \sum_{i \geq 0} \ell(x^i) x^i = \frac{Q_{n-1}}{\text{Rec}(P_n)} + \mathcal{O}(x^{2n}),$  where

$$Q_n = (a_{n+1} + b_{n+1}x)Q_{n-1} + c_{n+1}x^2Q_{n-2}, Q_{-1} = 0, Q_0 = 1.$$

$$\mathbf{P}^t \cdot (\ell(x^{i+j})) \cdot \mathbf{P} = \text{Diag}(d_i) \Rightarrow \mathbf{P}^{-1} = \text{Diag}(d_i)^{-1} \cdot \mathbf{P}^t \cdot (\ell(x^{i+j})).$$

Algorithm  $(a) \mapsto \mathbf{P}^{-1}(a)$  (Bostan, S., Schost 08b)

- 1 Compute  $(d_i);$
- 2 Compute  $L(x) \bmod x^{2n}$  using Direct Conversion;
- 3  $(a) \mapsto (\ell(x^{i+j})) \cdot (a)$  is a transposed multiplication by  $L(x);$
- 4 Algorithm  $(a) \mapsto \mathbf{P}^t(a)$  obtained by transposing the code for Direct Conversion.

## VI Conclusion

# Conclusion, Further Work, Work in Progress

- **Summary:** transposition helps conversions to and from polynomials and power series with generating function  $u(x)v(t)f(g(x)h(t))$ .
- **General orthogonal polynomials** in  $\mathcal{O}(M(N) \log N)$  by a fast effective Shohat-Favard theorem and again transposition;
- **$k$ -term recurrences** with  $k > 3$ ?
- **D-finite series** decompose on orthogonal polynomials in  $\mathcal{O}(N)$  ops: A. Benoit, extending (Pazkowski, Lewanowicz, Rebillard, . . . );
- **Numerically stable algorithms?**
- $\mathcal{O}(M(N))$ : How far can one go?