Fast Evaluation and Interpolation

Alin Bostan

MPRI C-2-22
October 21, 2024

Solution of Ex. 2 from last week

Prove the following identity of formal power series:

$$\arcsin(x)^2 = \sum_{k\geq 0} \frac{k!}{\left(\frac{1}{2}\right)\cdots\left(k+\frac{1}{2}\right)} \frac{x^{2k+2}}{2k+2} \ .$$

For this:

1. Check that $y = \arcsin(x)$ can be represented by the differential equation $(1-x^2)y'' - xy' = 0$ and the initial conditions $y(0) = 0$, $y'(0) = 1$.
2. Compute a linear differential equation satisfied by $z(x) = y(x)^2$.
3. Deduce a linear recurrence relation satisfied by the coefficients of $z(x)$.
4. Conclude.

Solution, Part 1.

The starting point is the identity

$$(\arcsin(x))' = \frac{1}{\sqrt{1-x^2}}$$

and that fact that $u = (1-x^2)^\alpha$ satisfies $u'/u = -2x\alpha/(1-x^2)$.

This allows to represent $\arcsin(x)$ by the differential equation

$$(1-x^2)y'' - xy' = 0$$

together with the initial conditions

$$y(0) = \arcsin(0) = 0, \quad y'(0) = \frac{1}{\sqrt{1-0^2}} = 1.$$

Solution, Part 1.

Let $z = y^2$, with $y'' = \frac{x}{1-x^2} y'$. By successive differentiations, we get

$$z' = 2yy',$$

$$z'' = 2y'^2 + 2yy'' = 2y'^2 + \frac{2x}{1-x^2} yy',$$

$$z''' = 4y'y'' + \frac{2x}{1-x^2}(y'^2 + yy'') + \left(\frac{2}{1-x^2} + \frac{4x^2}{(1-x^2)^2}\right) yy'$$

$$= \left(\frac{2}{1-x^2} + \frac{6x^2}{(1-x^2)^2}\right) yy' + \frac{6x}{1-x^2} y'^2.$$

- z, z', z'', z''' are $\mathbb{Q}(x)$-linear comb. of y^2, yy', y'^2, thus $\mathbb{Q}(x)$-dependent
- A dependence relation is determined by computing the kernel of

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & \frac{2x}{1-x^2} & \frac{2}{1-x^2} + \frac{6x^2}{(1-x^2)^2} \\ 0 & 0 & 2 & \frac{6x}{1-x^2} \end{bmatrix}$$

- $\ker(M)$ is generated by $[0,\ 1,\ 3x,\ x^2-1]^T \longrightarrow (x^2-1)z''' + 3xz'' + z' = 0$.

Solution, Part 2., a variant

Let $z = y^2$, with $y'' = \frac{x}{1-x^2} y'$. By successive differentiations, we get

$$z' = 2yy',$$

$$z'' = 2y'^2 + 2yy'' = 2y'^2 + \frac{2x}{1-x^2} yy' = 2y'^2 + \frac{x}{1-x^2} z',$$

$$z''' = 4y'y'' + \frac{x}{1-x^2} z'' + \left(\frac{1}{1-x^2} + \frac{2\,x^2}{(1-x^2)^2} \right) z'$$

$$= \frac{4x}{1-x^2} y'^2 + \frac{x}{1-x^2} z'' + \frac{x^2+1}{(x^2-1)^2} z'$$

$$= \frac{2x}{1-x^2} \left(z'' - \frac{x}{1-x^2} z' \right) + \frac{x}{1-x^2} z'' + \frac{x^2+1}{(x^2-1)^2} z'.$$

▷ The corresponding differential equation is

$$(x^2-1)z''' + 3xz'' + z' = 0.$$

Solution, Part 3.

▷ Write $z(x) = \sum_n a_n x^n$. Then:

$$z' = \sum_n (n+1) a_{n+1} x^n,$$

$$z'' = \sum_n (n+1)(n+2) a_{n+2} x^n,$$

$$z''' = \sum_n (n+1)(n+2)(n+3) a_{n+3} x^n.$$

▷ The coefficient of x^n in $(x^2-1)z''' + 3xz'' + z'$ is

$$(n-1)n(n+1)a_{n+1} - (n+1)(n+2)(n+3)a_{n+3} + 3n(n+1)a_{n+1} + (n+1)a_{n+1}$$

▷ Thus, the recurrence corresponding to $(x^2-1)z''' + 3xz'' + z' = 0$ is

$$(n+1)(n+2)(n+3)a_{n+3} = (n+1)^3 a_{n+1}.$$

▷ Since $(n+1)$ has no roots in $\mathbb{N}$, it further simplifies to

$$(n+2)(n+3)a_{n+3} - (n+1)^2 a_{n+1} = 0.$$

Solution, Part 4.

- $z = \sum_n a_n x^n$ satisfies $\quad (n+2)(n+3)a_{n+3} - (n+1)^2 a_{n+1} = 0.$
- Initial conditions:

$$a_0 = z(0) = y(0)^2 = 0,\ a_1 = z'(0) = 2y(0)y'(0) = 0,\ a_2 = \frac{1}{2}z''(0) = y'(0)^2 = 1.$$

- Recurrence and $a_1 = 0$ imply $a_{2k+1} = 0$, so the series is even.
- Let $b_k = a_{2k+2}$. Then $z(x) = \sum_k b_k x^{2k+2}$ and

$$(2k+1)(2k+2)b_k = 4k^2 b_{k-1}, \quad b_0 = 1$$

- Thus, the sequence $(b_k)_k$ is hypergeometric and

$$b_k = 2\frac{k^2}{(k+1)(2k+1)}b_{k-1} = \cdots = 2^k \frac{k!^2}{(k+1)!(2k+1)(2k-1)\cdots 3}$$

$$= \frac{k!}{(k+1)\cdot(k+\frac{1}{2})(k-\frac{1}{2})\cdots\frac{3}{2}} = \frac{k!}{(k+\frac{1}{2})(k-\frac{1}{2})\cdots\frac{1}{2}}\frac{1}{2k+2}$$

□

Context

▷ Main concepts: Evaluation-interpolation paradigm and Modular algorithms

▷ Alternative representations of algebraic objects: e.g., polynomials given

- by list of coefficients: useful for fast division
- by list of values taken on given points: useful for fast multiplication (FFT)

▷ Modular algorithms based on fast conversions between representations, e.g. evaluation-interpolation, Chinese Remaindering

▷ Avoid intermediate expression swell, e.g. det of polynomial matrices

▷ Important issue: choice of the moduli (evaluation points), e.g. fast factorial

Main problems and results

Multipoint evaluation Given P in $\mathbb{A}[X]$, of degree $< n$, compute the values

$$P(a_0), \ldots, P(a_{n-1}).$$

Interpolation Given $v_0, \ldots, v_{n-1} \in \mathbb{A}$, with $a_i - a_j$ invertible in $\mathbb{A}$ if $i \neq j$, find the polynomial $P \in \mathbb{A}[X]$ of degree $< n$ such that

$$P(a_0) = v_0, \ldots, P(a_{n-1}) = v_{n-1}.$$

Theorem One can solve both problems in:

- $O(\mathsf{M}(n) \log n)$ ops. in $\mathbb{A}$
- $O(\mathsf{M}(n))$ ops. in $\mathbb{A}$ if the a_i's are in geometric progression

▷ Extension to fast polynomial/integer Chinese remaindering

Waring-Lagrange interpolation

THEOREM I.

Assume an equation $a+bx+cx^2+dx^3 \ldots\ldots x^{n-1}=y$, in which the co-efficients a, b, c, d, e, &c. are invariable; let $\alpha, \beta, \gamma, \delta, \varepsilon$, &c. denote n values of the unknown quantity x, whose correspondent values of y let be represented by $s^{\alpha}, s^{\beta}, s^{\gamma}, s^{\delta}, s^{\varepsilon}$, &c. Then will the equation $a+bx+cx^2+dx^3+ex^4 \ldots x^{n-1} = y =$

$$\frac{\overline{x-\beta}\times\overline{x-\gamma}\times\overline{x-\delta}\times\overline{x-\varepsilon}\times \&c.}{\overline{\alpha-\beta}\times\overline{\alpha-\gamma}\times\overline{\alpha-\delta}\times\overline{\alpha-\varepsilon}\times \&c.}\times s^{\alpha}+\frac{\overline{x-\alpha}\times\overline{x-\gamma}\times\overline{x-\delta}\times\overline{x-\varepsilon}\times \&c.}{\overline{\beta-\alpha}\times\overline{\beta-\gamma}\times\overline{\beta-\delta}\times\overline{\beta-\varepsilon}\times \&c.}\times s^{\beta}$$

$$+\frac{\overline{x-\alpha}\times\overline{x-\beta}\times\overline{x-\delta}\times\overline{x-\varepsilon}\times \&c.}{\overline{\gamma-\alpha}\times\overline{\gamma-\beta}\times\overline{\gamma-\delta}\times\overline{\gamma-\varepsilon}\times \&c.}\times s^{\gamma}+\frac{\overline{x-\alpha}\times\overline{x-\beta}\times\overline{x-\gamma}\times\overline{x-\varepsilon}\times \&c.}{\overline{\delta-\alpha}\times\overline{\delta-\beta}\times\overline{\delta-\gamma}\times\overline{\delta-\varepsilon}\times \&c.}\times s^{\delta}$$

$$+\frac{\overline{x-\alpha}\times\overline{x-\beta}\times\overline{x-\gamma}\times\overline{x-\delta}\times \&c.}{\overline{\varepsilon-\alpha}\times\overline{\varepsilon-\beta}\times\overline{\varepsilon-\gamma}\times\overline{\varepsilon-\delta}\times \&c.}\times s^{\varepsilon}+ \&c.$$

[Waring, 1779 – “Problems concerning Interpolations”]

qu'en faisant $x = p$ on ait

$$A = 1, \quad B = 0, \quad C = 0, \quad \ldots;$$

que de même, en faisant $x = q$, on ait

$$A = 0, \quad B = 1, \quad C = 0, \quad D = 0, \quad \ldots;$$

qu'en faisant $x = r$, on ait pareillement

$$A = 0, \quad B = 0, \quad C = 1, \quad D = 0, \quad \ldots, \quad \text{etc.};$$

d'où il est facile de conclure que les valeurs de A, B, C,... doivent être de cette forme

$$A = \frac{(x-q)(x-r)(x-s)\ldots}{(p-q)(p-r)(p-s)\ldots},$$

$$B = \frac{(x-p)(x-r)(x-s)\ldots}{(q-p)(q-r)(q-s)\ldots},$$

$$C = \frac{(x-p)(x-q)(x-s)\ldots}{(r-p)(r-q)(r-s)\ldots},$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots,$$

en prenant autant de facteurs, dans les numérateurs et dans les dénominateurs, qu'il y aura de points donnés de la courbe, moins un.

Cette dernière expression de y, quoique sous une forme différente, revient cependant au même, comme on peut s'en assurer par le calcul, en développant les valeurs des quantités Q_1, R_2, S_3,..., et ordonnant les termes suivant les quantités P, Q, R,...; mais elle est préférable par la simplicité de l'Analyse sur laquelle elle est fondée, et par sa forme même, qui est beaucoup plus commode pour le calcul.

[Lagrange, 1795 – "Sur l'usage des courbes dans la solution des problèmes"]

Fast polynomial division

Euclidean division for polynomials

[Strassen, 1973]

Pb: Given $F, G \in \mathbb{K}[x]_{\leq N}$, compute (Q, R) in Euclidean division $F = QG + R$

Naive algorithm: $O(N^2)$

Idea: look at $F = QG + R$ from infinity: $Q \sim_{+\infty} F/G$

Let $N = \deg(F)$ and $n = \deg(G)$. Then $\deg(Q) = N - n$, $\deg(R) < n$ and

$$\underbrace{F(1/x)x^N}_{\mathrm{rev}(F)} = \underbrace{G(1/x)x^n}_{\mathrm{rev}(G)} \cdot \underbrace{Q(1/x)x^{N-n}}_{\mathrm{rev}(Q)} + \underbrace{R(1/x)x^{\deg(R)}}_{\mathrm{rev}(R)} \cdot x^{N-\deg(R)}$$

Algorithm:

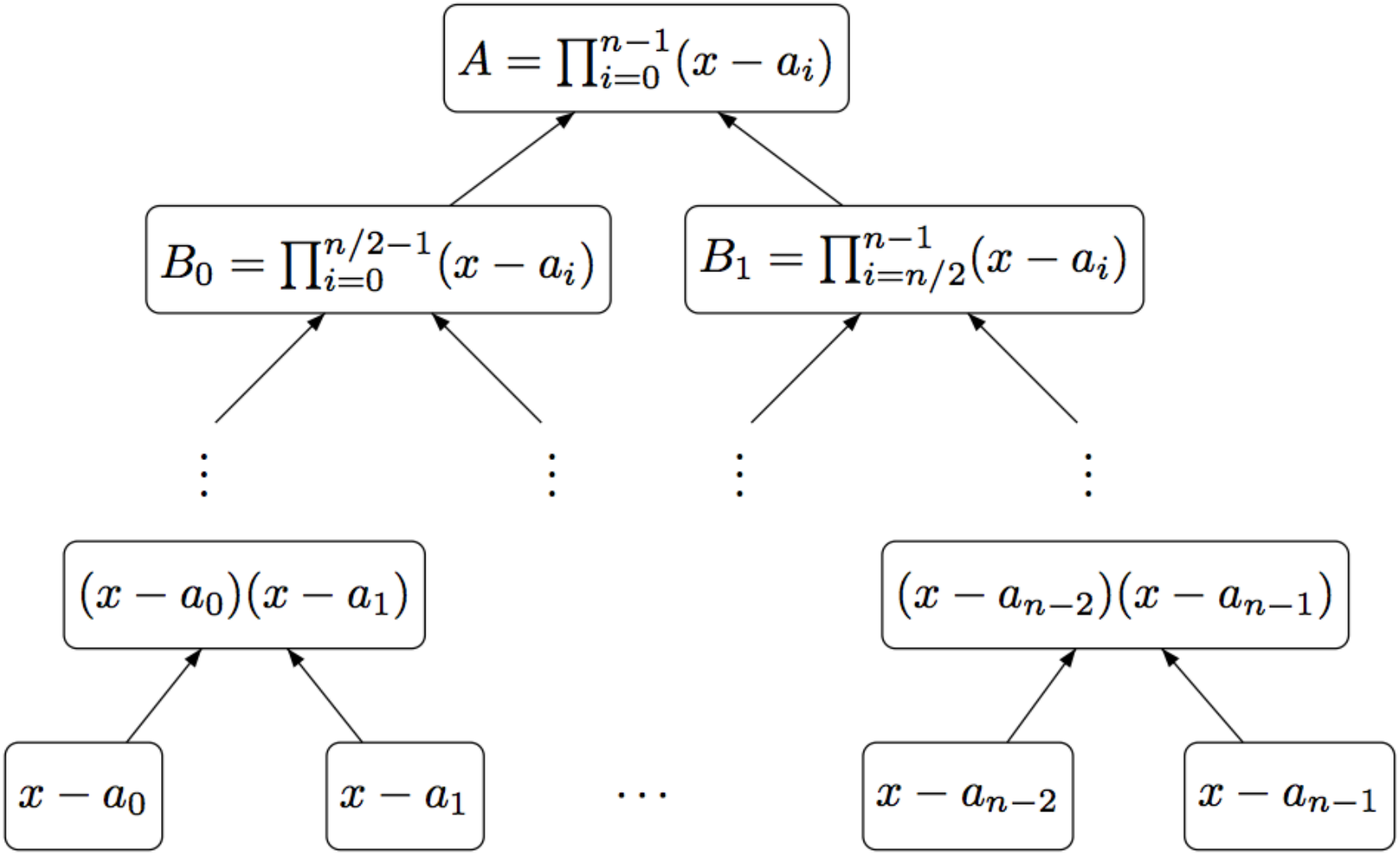
- Compute $\mathrm{rev}(Q) = \mathrm{rev}(F)/\mathrm{rev}(G) \mod x^{N-n+1}$ $O(\mathsf{M}(N))$
- Recover Q $O(1)$
- Deduce $R = F - QG$ $O(\mathsf{M}(N))$

Evaluation-interpolation, general case

Subproduct tree

[Horowitz, 1972]

Problem: Given $a_0, \ldots, a_{n-1} \in \mathbb{K}$, compute $A = \prod_{i=0}^{n-1}(x - a_i)$



DAC Theorem: $\mathsf{S}(n) = 2 \cdot \mathsf{S}(n/2) + O(\mathsf{M}(n)) \implies \mathsf{S}(n) = O(\mathsf{M}(n) \log n)$

Fast multipoint evaluation

[Borodin-Moenck, 1974]

Pb: Given $a_0, \ldots, a_{n-1} \in \mathbb{K}$ and $P \in \mathbb{K}[x]_{<n}$, compute $P(a_0), \ldots, P(a_{n-1})$

Naive algorithm: Compute $P(a_i)$ independently $O(n^2)$

Basic idea: Use recursively Bézout's identity $P(a) = P(x) \bmod (x - a)$

Divide and conquer: Same idea as for DFT = evaluation by repeated division

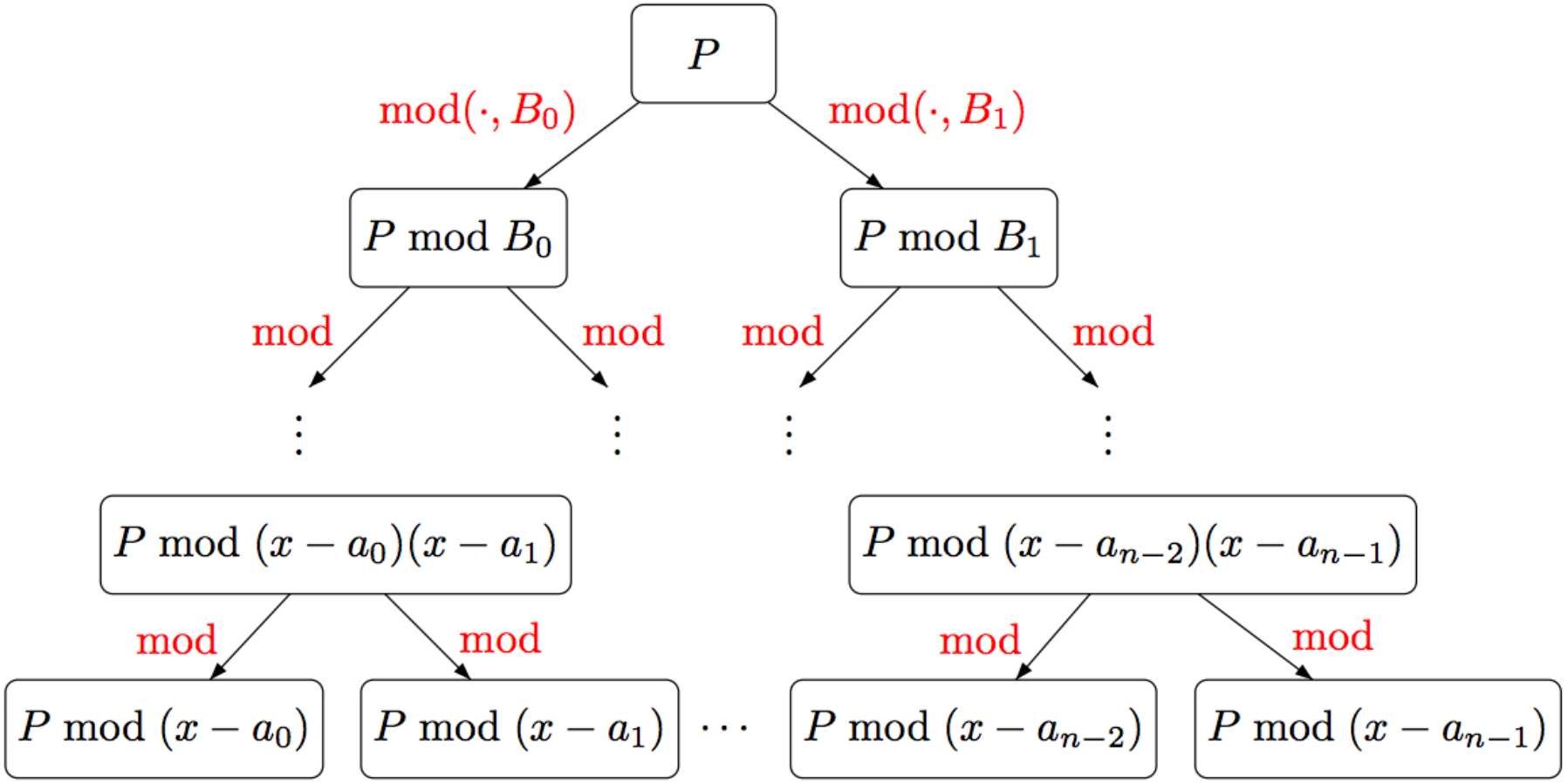
- $P_0 := P \mod \underbrace{(x - a_0) \cdots (x - a_{n/2-1})}_{B_0}$
- $P_1 := P \mod \underbrace{(x - a_{n/2}) \cdots (x - a_{n-1})}_{B_1}$

$$\Longrightarrow \quad \begin{cases} P(a_0) = P_0(a_0), & \ldots, \quad P(a_{n/2-1}) = P_0(a_{n/2-1}) \\ P(a_{n/2}) = P_1(a_{n/2}), & \ldots, \quad P(a_{n-1}) = P_1(a_{n-1}) \end{cases}$$

Fast multipoint evaluation

[Borodin-Moenck, 1974]

Pb: Given $a_0, \ldots, a_{n-1} \in \mathbb{K}$ and $P \in \mathbb{K}[x]_{<n}$, compute $P(a_0), \ldots, P(a_{n-1})$



DAC Theorem: $\mathsf{E}(n) = 2 \cdot \mathsf{E}(n/2) + O(\mathsf{M}(n)) \implies \mathsf{E}(n) = O(\mathsf{M}(n) \log n)$

Fast interpolation

[Borodin-Moenck, 1974]

Problem: Given $a_0, \ldots, a_{n-1} \in \mathbb{K}$ mutually distinct, and $v_0, \ldots, v_{n-1} \in \mathbb{K}$, compute $P \in \mathbb{K}[x]_{<n}$ such that $P(a_0) = v_0, \ldots, P(a_{n-1}) = v_{n-1}$

Naive algorithm: Linear algebra, Vandermonde system $O(\mathsf{MM}(n))$

Lagrange's algorithm: Use $P(x) = \sum_{i=0}^{n-1} v_i \frac{\prod_{j \neq i}(x - a_j)}{\prod_{j \neq i}(a_i - a_j)}$ $O(n^2)$

Fast algorithm: Based on the "modified Lagrange formula"

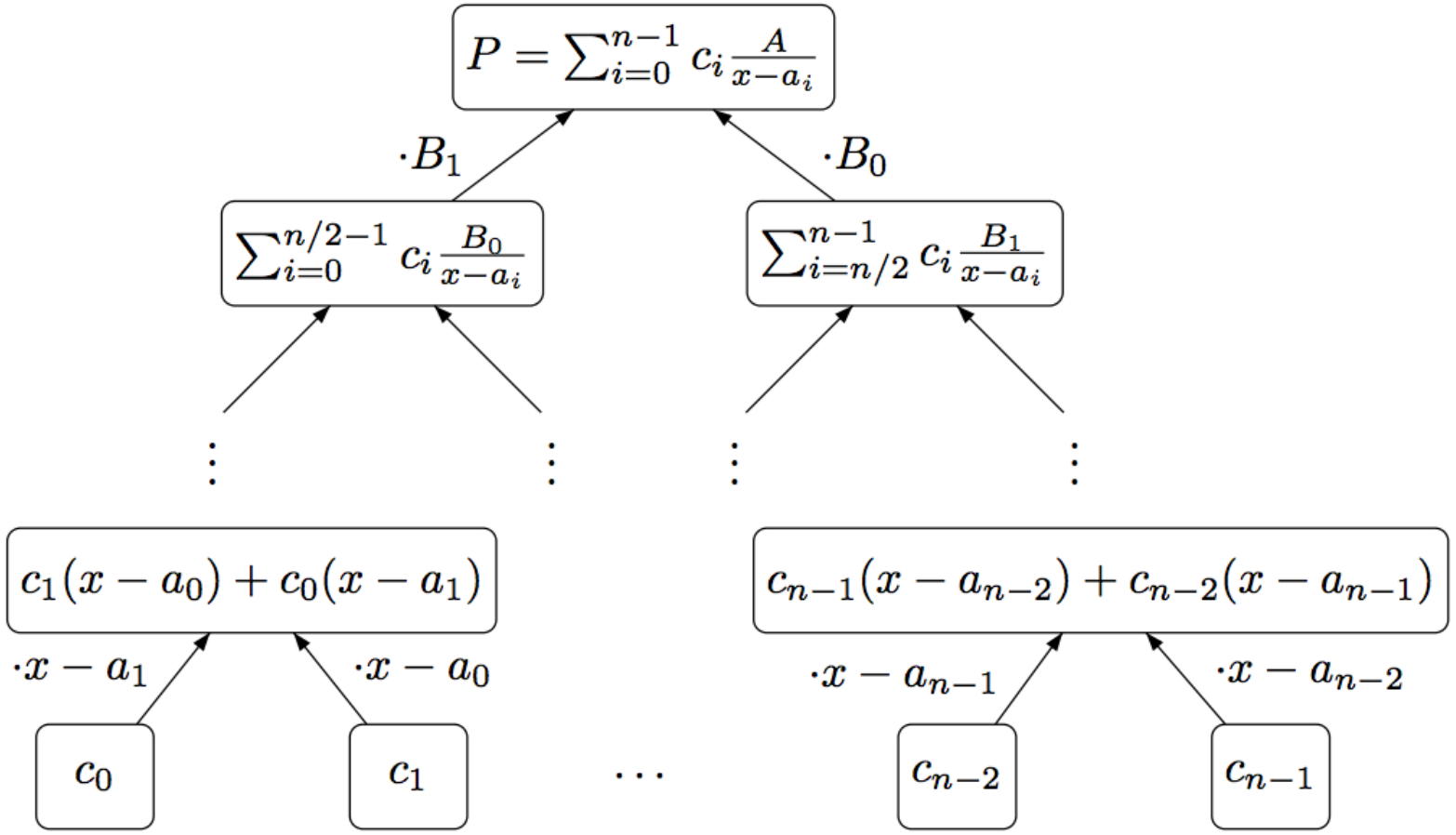
$$P(x) = A(x) \cdot \sum_{i=0}^{n-1} \frac{v_i / A'(a_i)}{x - a_i}$$

- Compute $c_i = v_i / A'(a_i)$ by fast multipoint evaluation $O(\mathsf{M}(n) \log n)$
- Compute $\sum_{i=0}^{n-1} \frac{c_i}{x - a_i}$ by divide and conquer $O(\mathsf{M}(n) \log n)$

Fast interpolation

[Borodin-Moenck, 1974]

Problem: Given $a_0, \ldots, a_{n-1} \in \mathbb{K}$ mutually distinct, and $v_0, \ldots, v_{n-1} \in \mathbb{K}$, compute $P \in \mathbb{K}[x]_{<n}$ such that $P(a_0) = v_0, \ldots, P(a_{n-1}) = v_{n-1}$



DAC Theorem: $\mathsf{I}(n) = 2 \cdot \mathsf{I}(n/2) + O(\mathsf{M}(n)) \implies \mathsf{I}(n) = O(\mathsf{M}(n) \log n)$

Evaluation-interpolation, geometric case

Subproduct tree, geometric case

[B.-Schost, 2005]

Problem: Given $q \in \mathbb{K}$, compute $A = \prod_{i=0}^{n-1}(x - q^i)$

Idea: Compute $B_1 = \prod_{i=n/2}^{n-1}(x - q^i)$ from $B_0 = \prod_{i=0}^{n/2-1}(x - q^i)$, by a homothety

$$B_1(x) = B_0\left(\frac{x}{q^{n/2}}\right) \cdot q^{(n/2)^2}$$

Decrease and conquer:

- Compute $B_0(x)$ by a recursive call
- Deduce $B_1(x)$ from $B_0(x)$ $O(n)$
- Return $A(x) = B_0(x)B_1(x)$ $\mathsf{M}(n/2)$

Master Theorem: $\mathsf{G}(n) = \mathsf{G}(n/2) + O(\mathsf{M}(n)) \quad \Longrightarrow \quad \mathsf{G}(n) = O(\mathsf{M}(n))$

Fast multipoint evaluation, geometric case

[Bluestein, 1970]

Problem: Given $q \in \mathbb{K}$ and $P \in \mathbb{K}[x]_{<n}$, compute $P(1), P(q), \ldots, P(q^{n-1})$

The needed values are: $$P(q^i) = \sum_{j=0}^{n-1} c_j q^{ij}, \qquad 0 \le i < n$$

Bluestein's trick: $$ij = \binom{i+j}{2} - \binom{i}{2} - \binom{j}{2} \implies q^{ij} = q^{\binom{i+j}{2}} \cdot q^{-\binom{i}{2}} \cdot q^{-\binom{j}{2}}$$

$$\implies \qquad P(q^i) = q^{-\binom{i}{2}} \cdot \underbrace{\sum_{j=0}^{n-1} c_j q^{-\binom{j}{2}} \cdot q^{\binom{i+j}{2}}}_{\text{convolution:}}$$

$$\text{coeff. of } x^{n-1+i} \text{ in } \left(\sum_{j=0}^{n-1} c_j q^{-\binom{j}{2}} x^{n-j-1}\right)\left(\sum_{\ell=0}^{2n-2} q^{\binom{\ell}{2}} x^{\ell}\right)$$

Conclusion: Fast evaluation on a geometric sequence in $O(\mathsf{M}(n))$

Fast interpolation, geometric case

[B.-Schost, 2005]

Problem: Given $q \in \mathbb{K}$, and $v_0, \ldots, v_{n-1} \in \mathbb{K}$, compute $P \in \mathbb{K}[x]_{<n}$ such that $P(1) = v_0, \ldots, P(q^{n-1}) = v_{n-1}$

Fast algorithm: Modified Lagrange formula

$$P = A(x) \cdot \sum_{i=0}^{n-1} \frac{v_i / A'(q^i)}{x - q^i}, \qquad A = \prod_i (x - q^i)$$

- Compute $A = \prod_{i=0}^{n-1} (x - q^i)$ by decrease and conquer $O(\mathsf{M}(n))$
- Compute $c_i = v_i / A'(q^i)$ by Bluestein's algorithm $O(\mathsf{M}(n))$
- Compute $\sum_{i=0}^{n-1} \frac{c_i}{x - q^i}$ by decrease and conquer $O(\mathsf{M}(n))$

Fast interpolation, geometric case

[B.-Schost, 2005]

Problem: Given $q \in \mathbb{K}$, and $v_0, \ldots, v_{n-1} \in \mathbb{K}$, compute $P \in \mathbb{K}[x]_{<n}$ such that $P(1) = v_0, \ldots, P(q^{n-1}) = v_{n-1}$

Subproblem: Given $c_0, \ldots, c_{n-1} \in \mathbb{K}$, compute $\quad R(x) = \sum_{i=0}^{n-1} \frac{c_i}{x - q^i}$

Idea: change of representation – enough to compute $R \mod x^n$

Second idea: $R \bmod x^n$ = multipoint evaluation at $\{1, q^{-1}, \ldots, q^{-(n-1)}\}$:

$$\sum_{i=0}^{n-1} \frac{c_i}{x - q^i} \mod x^n = -\sum_{i=0}^{n-1} \left(\sum_{j=0}^{n-1} c_i q^{-i(j+1)} x^j \right) = -\sum_{j=0}^{n-1} C(q^{-j-1}) x^j$$

Conclusion: Algorithm for interpolation at a geometric sequence in $O(\mathsf{M}(n))$ (generalization of the FFT algorithm computing the IDFT)

Product of polynomial matrices

[B.-Schost, 2005]

Problem: Given $A, B \in \mathcal{M}_n(\mathbb{K}[x]_{<d})$, compute $C = AB$

Idea: change of representation – evaluation-interpolation at a geometric sequence $\mathcal{G} = \{1, q, q^2, \ldots, q^{2d-2}\}$

- Evaluate A and B at $\mathcal{G}$ $O(n^2\,\mathsf{M}(d))$
- Multiply values $C(v) = A(v)B(v)$ for $v \in \mathcal{G}$ $O(d\,\mathsf{MM}(n))$
- Interpolate C from values $O(n^2\,\mathsf{M}(d))$

Total complexity $O(n^2\,\mathsf{M}(d) + d\,\mathsf{MM}(n))$

An exercise for next Monday

Let f and g be two polynomials in $\mathbb{K}[x, y]$ of degrees at most d_x in x and at most d_y in y.

(a) Show that it is possible to compute the product $h = fg$ using

$$O(\mathsf{M}(d_x d_y))$$

arithmetic operations in $\mathbb{K}$.
Hint: Use the substitution $x \leftarrow y^{2d_y+1}$ to reduce the problem to the product of univariate polynomials.

(b) Improve this result by proposing an evaluation-interpolation scheme which allows the computation of h in

$$O(d_x\,\mathsf{M}(d_y) + d_y\,\mathsf{M}(d_x))$$

arithmetic operations in $\mathbb{K}$.

1. Space-saving versions

	Time	Space	Reference
multipoint evaluation	$3/2\mathsf{M}(n)\log(n)$	$n\log(n)$	[2]
size-n polynomial on n points	$7/2\mathsf{M}(n)\log(n)$	n	[7], Lemma 3.1
	$(4+2\lambda_s/\log(\frac{c_s+3}{c_s+2}))\mathsf{M}(n)\log(n)$	$O(1)$	Theorem 3.4
interpolation	$5/2\mathsf{M}(n)\log(n)$	$n\log(n)$	[2]
size-n polynomial on n points	$5\mathsf{M}(n)\log(n)$	$2n$	[6, 7], Lemma 3.3
	$\simeq 105\mathsf{M}(n)\log(n)$	$O(1)$	Theorem 3.6

[Giorgi, Grenet & Roche, ISSAC, 2020]

[2] A. Bostan, G. Lecerf, and É. Schost. 2003. Tellegen's Principle into Practice. In ISSAC'03, 37–44.

[6] J. von zur Gathen and V. Shoup. 1992. Computing Frobenius maps and factoring polynomials. Comput. Complex. 2, 3 (1992), 187–224.

[7] P. Giorgi, B. Grenet, and D. S. Roche. 2019. Generic reductions for in-place polynomial multiplication. In ISSAC'19, 187–194.

2. More general evaluation and interpolation

SIAM J. COMPUT.
Vol. 5, No. 4, December 1976

A GENERALIZED ASYMPTOTIC UPPER BOUND ON FAST POLYNOMIAL EVALUATION AND INTERPOLATION*

FRANCIS Y. CHIN†

Abstract. It is shown in this paper that the evaluation and interpolation problems corresponding to a set of points, $\{x_i\}_{i=0}^{n-1}$, with $(c_i - 1)$ higher derivatives at each x_i such that $\sum_{i=1}^{n-1} c_i = N$, can be solved in $O([N \log N][(\log n) + 1])$ steps.[1] This upper bound matches perfectly with the known upper bounds of the two extreme cases, which are $O(N \log^2 N)$ and $O(N \log N)$ steps when $n = N$ and $n = 1$, respectively.

Key words. polynomial evaluation, polynomial interpolation, asymptotic upper bounds

	Evaluation	Interpolation
(a) N points	$O(N \log^2 N)$[6], [5]	$O(N \log^2 N)$ [6], [5]
(b) n points, $\{x_i\}_{i=0}^{n-1}$ and their corresponding $(c_i - 1)$ derivatives such that $\sum_{i=0}^{n-1} c_i = N$	$O([N \log N][(\log n) + 1])$ (Theorem 1)	$O([N \log N][(\log n) + 1])$ (Theorem 2)
(c) Single point and all its derivatives	$O(N \log N)$ [2], [9]	$O(N \log N)$ [2], [9]

[Chin, SIAM J. Comput., 1976]

3. Multivariate sparse interpolation

Q.-L. Huang, X.-S. Gao / Journal of Symbolic Computation 101 (2020) 367–386

Table 1
A "soft-Oh" comparison for SLP polynomials over an arbitrary ring $\mathcal{R}$.

Algorithms	Total Cost	Type
Dense	LD^n	Deterministic
Garg and Schost (2009)	$Ln^2T^4\log^2 D$	Deterministic
Randomized (Giesbrecht and Roche, 2011)	$Ln^2T^3\log^2 D$	Las Vegas
Arnold et al. (2013)	$Ln^3T\log^3 D$	Monte Carlo
This paper (Theorem 5.8)	$Ln^2T^2\log^2 D + LnT\log^3 D$	Deterministic
This paper (Theorem 6.8)	$LnT\log^3 D$	Monte Carlo

Table 2
A "soft-Oh" comparison for SLP polynomials over finite field $\mathbb{F}_q$.

Algorithms	Bit Complexity	Algorithm type
Garg and Schost (2009)	$Ln^2T^4\log^2 D\log q$	Deterministic
Randomized Garg-Schost (Giesbrecht and Roche, 2011)	$Ln^2T^3\log^2 D\log q$	Las Vegas
Giesbrecht and Roche (2011)	$Ln^2T^2\log^2 D(n\log D+\log q)$	Las Vegas
Arnold et al. (2013)	$Ln^3T\log^3 D\log q$	Monte Carlo
Arnold et al. (2014)	$LnT\log^2 D(\log D+\log q)+n^\omega T$	Monte Carlo
Arnold et al. (2016)	$Ln\log D(T\log D+n)(\log D+\log q)$ $+n^{\omega-1}T\log D+n^\omega\log D$	Monte Carlo
This paper (Theorem 5.8)	$Ln^2T^2\log^2 D\log q + LnT\log^3 D\log q$	Deterministic
This paper (Theorem 6.8)	$LnT\log^3 D\log q$	Monte Carlo
This paper (Theorem 6.11)	$LnT\log^2 D(\log q+\log D)$	Monte Carlo

[Huang & Gao, JSC, 2020]