EXERCISE ON NÜSKEN-ZIEGLER ALGORITHM

1. Multipoint evaluation of bivariate polynomials

We consider the following two algorithmic problems:

$\mathsf{BivMPEval}$: Given parameters $n, d_x, d_y \in \mathbb{Z}_{>0}$, given n points $(a_1, b_1), \ldots, (a_n, b_n)$ in $\mathbb{K}^2$ with the a_i's pairwise distinct, given a polynomial $F(x, y) \in \mathbb{K}[x, y]$ with $\deg_x(F) < d_x$ and $\deg_y(F) < d_y$, compute the evaluations $F(a_1, b_1), \ldots, F(a_n, b_n)$.

This is bivariate multipoint evaluation, with the restriction that the x-coordinates are distinct. At the price of some "mild randomization", one can reduce to this situation from the general case of n distinct points; this reduction will not be discussed here.

$\mathsf{BivModComp}$: Given parameters $n, d_x, d_y \in \mathbb{Z}_{>0}$, given a polynomial $F(x, y) \in \mathbb{K}[x, y]$ with $\deg_x(F) < d_x$ and $\deg_y(F) < d_y$, given polynomials $P, Q \in \mathbb{K}[x]$ with $\deg(P) < \deg(Q) = n$, compute the modular composition $F(x, P(x)) \bmod Q(x)$.

This is one type of bivariate extension of the modular composition of univariate polynomials.

(1) Give a complexity bound for a naive algorithm for $\mathsf{BivMPEval}$, which performs n evaluations of F at a single point.
(2) Exploiting the polynomials $Q = \prod_{1 \le i \le n}(x - a_i)$ and $P \in \mathbb{K}[x]_{<n}$ such that $P(a_i) = b_i$ for $1 \le i \le n$, prove that: from an algorithm $\mathscr{A}$ for $\mathsf{BivModComp}$ with complexity $\mathsf{C}(n, d_x, d_y)$, one can derive an algorithm $\mathscr{B}$ for $\mathsf{BivMPEval}$ whose complexity is bounded by $\mathsf{C}(n, d_x, d_y) + O(\mathsf{M}(n) \log(n))$.
(3) Give an algorithm for $\mathsf{BivModComp}$ based on the writing $F = \sum_{j<d_y} F_j(x) y^j$ and on "naive" evaluation at $y = P \bmod Q$. Show that its complexity is in $O(d_y \mathsf{M}(d_x) + d_y \mathsf{M}(n))$, and give a range for (d_x, n) for which this is quasi-linear in the input size.

From here on, we focus on the case $d_x \in O(n)$. Our goal is to obtain a better complexity bound for $\mathsf{BivModComp}$ (thus also for $\mathsf{BivMPEval}$) than the one $O(d_y \mathsf{M}(n))$ from Question (3).

(4) If $d_y \in O(n)$ and $d_x = 1$, do you know a better complexity bound for $\mathsf{BivModComp}$?
(5) For simplicity, we now also assume (in addition to $d_x \in O(n)$) that d_y is a square $d_y = \delta^2$ with $\delta \in \mathbb{Z}_{>0}$. Writing $F = \sum_{j<\delta} \left(\sum_{i<\delta} F_{i+\delta j}(x) y^i\right) y^{\delta j}$, give an algorithm for $\mathsf{BivModComp}$ which exploits polynomial matrix multiplication.
[*Hint*: compute $\sum_{i<\delta} F_{i+\delta j}(x) P^i \bmod Q$ simultaneously for all j via the multiplication of a $\delta \times \delta$ univariate matrix of degree $< d_x$ by a $\delta \times 1$ univariate vector of degree $< n$.]
(6) Recall how to perform the above matrix-vector product using $O(\delta^\omega \mathsf{M}(\frac{n}{\delta} + d_x))$ operations in $\mathbb{K}$. Deduce that the algorithm of Question (5) costs $O(\delta \mathsf{M}(n) + \delta^\omega \mathsf{M}(\frac{n}{\delta} + d_x))$.
(7) As soon as $d_x \in O(\frac{n}{\delta})$ (in particular, in the frequent case $d_x d_y \in \Theta(n)$), the above bound is within $O(\delta^{\omega-1} \mathsf{M}(n))$. How does this compare to $O(d_y \mathsf{M}(n))$?

Solution:

(1) The evaluation of F at a single point (a_i, b_i) costs $O(d_x d_y)$ operations in $\mathbb{K}$. Evaluating naively at each point independently will thus cost $O(n d_x d_y)$ operations in $\mathbb{K}$.
(2) Define the *univariate* polynomial $R = F(x, P) \bmod Q$. Then $R = F(x, P(x)) + A(x)Q(x)$ for some $A \in \mathbb{K}[x]$. Since Q has roots $a_1, \ldots, a_n$, we get, for $1 \le i \le n$, $R(a_i) = F(a_i, P(a_i)) + A(a_i)Q(a_i) = F(a_i, b_i)$. Therefore an algorithm $\mathscr{B}$ to solve $\mathsf{BivMPEval}$ is
   - compute P and Q from the points (subproduct tree & Lagrange interpolation);
   - use algorithm $\mathscr{A}$ to solve $\mathsf{BivModComp}$ on input P, Q, F, which provides the polynomial R above;
   - compute $R(a_1), \ldots, R(a_n)$ (univariate multipoint evaluation).

   The complexity of the first and third steps is $O(\mathsf{M}(n) \log(n))$, hence the sought overall bound $\mathsf{C}(n, d_x, d_y) + O(\mathsf{M}(n) \log(n))$.

(3) Writing $F = \sum_{j<d_y} F_j(x)y^j$, we obtain

$$F(x, P(x)) \bmod Q(x) = \left(\sum_{j<d_y} F_j(x) \left(P(x)^j \bmod Q(x) \right) \right) \bmod Q(x).$$

This formula leads straightforwardly to solving $\mathsf{BivModComp}$ in $O(d_y \mathsf{M}(d_x) + d_y \mathsf{M}(n))$ operations in $\mathbb{K}$. The above complexity is quasi-linear in the size of the input when $n \in O(d_x)$.

(4) In this case, Paterson and Stockmeyer's (or Brent and Kung's) baby step giant step algorithm uses $O(n^{\frac{\omega+1}{2}} + n^{\frac{1}{2}} \mathsf{M}(n))$ operations in $\mathbb{K}$.

(5) Rewriting indices, we have

$$F = \sum_{i,j<\delta} F_{i+\delta j}(x) y^{i+\delta j} = \sum_{j<\delta} \left(\sum_{i<\delta} F_{i+\delta j}(x) y^i \right) y^{\delta j} = \sum_{j<\delta} \hat{F}_j(x, y) y^{\delta j}$$

where $\hat{F}_j(x, y) = \sum_{i<\delta} F_{i+\delta j}(x) y^i$, for $j < \delta$. We can solve $\mathsf{BivModComp}$ by solving several instances of it (each with smaller y-degree) with the polynomials $\hat{F}_j(x, y)$:

- for $j < \delta$, compute $R_j = \hat{F}_j(x, P(x)) \bmod Q(x)$ [δ instances of $\mathsf{BivModComp}$]
- for $j < \delta$, compute $P_j = P^{\delta j} \bmod Q$ [total cost: $O(\delta \mathsf{M}(n))$]
- compute and return $\sum_{j<\delta} (R_j P_j \bmod Q)$ [cost $O(\delta \mathsf{M}(n))$]

The complexity gain is obtained by realizing the computations of the R_j's simultaneously, using polynomial matrix multiplication:

$$\begin{bmatrix} R_0 \\ R_1 \\ \vdots \\ R_{\delta-1} \end{bmatrix} = \begin{bmatrix} F_0 & F_1 & \cdots & F_{\delta-1} \\ F_\delta & F_{1+\delta} & \cdots & F_{2\delta-1} \\ \vdots & \vdots & \cdots & \vdots \\ F_{(\delta-1)\delta} & F_{(\delta-1)\delta+1} & \cdots & F_{\delta^2-1} \end{bmatrix} \begin{bmatrix} 1 \\ P \\ P^2 \bmod Q \\ \cdots \\ P^{\delta-1} \bmod Q \end{bmatrix} \bmod Q$$

This can be split into several steps:

- compute the powers $P^i \bmod Q$ for $i < \delta$ [total $O(\delta \mathsf{M}(n))$]
- compute the matrix-vector product, efficiently by expanding the vector into a $\delta \times \delta$ matrix of degree less than n/δ [total $O(\delta^\omega \mathsf{M}(\frac{n}{\delta} + d_x))$]
- the previous step has yielded δ polynomials of degree less than $n + d_x \in O(n)$ $\to$ reduce them mod Q [total $O(\delta \mathsf{M}(n))$].

(6) Complexity bounds are given in the answer just above.

(7) Yes, since $O(\delta^{\omega-1} \mathsf{M}(n)) = O(d_y^{\frac{\omega-1}{2}} \mathsf{M}(n))$, and $\frac{\omega-1}{2} < 0.7$ with the best known value of $\omega < 2.4$.