Fast polynomial factorization, modular composition, and multipoint evaluation of multivariate polynomials in small characteristic

Christopher Umans*
Computer Science Department
California Institute of Technology
1200 E. California Blvd.
Pasadena, CA 91125

Abstract

We obtain randomized algorithms for factoring degree n univariate polynomials over $\mathbb{F}_q$ that use $O(n^{1.5+o(1)} + n^{1+o(1)} \log q)$ field operations, when the characteristic is at most $n^{o(1)}$. When $\log q < n$, this is asymptotically faster than the best previous algorithms (von zur Gathen & Shoup (1992) and Kaltofen & Shoup (1998)); for $\log q \geq n$, it matches the asymptotic running time of the best known algorithms.

The improvements come from a new algorithm for modular composition of degree n univariate polynomials, which is the asymptotic bottleneck in fast algorithms for factoring polynomials over finite fields. The best previous algorithms for modular composition use $O(n^{(\omega+1)/2})$ field operations, where ω is the exponent of matrix multiplication (Brent & Kung (1978)), with a slight improvement in the exponent achieved by employing fast rectangular matrix multiplication (Huang & Pan (1997)).

We show that modular composition and multipoint evaluation of multivariate polynomials are essentially equivalent in the sense that an algorithm for one achieving exponent α implies an algorithm for the other with exponent $\alpha + o(1)$, and vice versa. We then give a new algorithm that requires $O(n^{1+o(1)})$ field operations when the characteristic is at most $n^{o(1)}$, which is optimal up to lower order terms.

Our algorithms do not rely on fast matrix multiplication, in contrast to all previous subquadratic algorithms for these problems. The main operations are fast univariate polynomial arithmetic, multipoint evaluation, and interpolation, and consequently the algorithms could be feasible in practice.

*Supported by NSF CCF-0346991, BSF 2004329, a Sloan Research Fellowship, and an Okawa Foundation research grant.

1 Introduction

Polynomial factorization is one of the central problems in computer algebra. Milestones in the development of polynomial-time algorithms for factoring in $\mathbb{F}_q[X]$ are the algorithms of Berlekamp [Ber70], Cantor & Zassenhaus [CZ81], von zur Gathen & Shoup [vzGS92] and Kaltofen & Shoup [KS98]. See the surveys [vzGP01, Kal03, vzG06]. Presently, there are practical algorithms that factor degree n polynomials over $\mathbb{F}_q$ in $\widetilde{O}(n^2 + n \log q)$ operations, and sub-quadratic algorithms that rely on fast matrix multiplication [KS98]. Efficient algorithms for factoring polynomials over other domains (e.g. $\mathbb{Q}$, $\mathbb{Z}$, algebraic number fields) and for factoring multivariate polynomials in turn depend on factoring in $\mathbb{F}_q[X]$.

The bottleneck in most modern factoring algorithms (including the asymptotically fastest ones) turns out to be the computation of the "Frobenius power" polynomials, X^{q^i}, modulo the degree-n polynomial to be factored, for various i between 1 and n. When $i = n$, a repeated-squaring approach requires $n \log q$ modular multiplications of degree n polynomials. A clever improvement based on the so-called "polynomial representation of the Frobenius map" (an idea attributed to Kaltofen) was exploited in this context by von zur Gathen & Shoup [vzGS92]: first compute $X^q \bmod f(X)$ by repeated squaring, then *compose* that polynomial with itself modulo $f(X)$ to get

$$(X^q)^q \bmod f(X) = X^{q^2} \bmod f(X).$$

Repeating the composition $\log n$ times computes $X^{q^n} \bmod f(X)$ with only $\log q$ modular multiplications and $\log n$ modular compositions overall. There are sub-quadratic algorithms for modular composition, and so this approach is asymptotically superior to the straightforward repeated-squaring algorithm. The same idea can also be applied to other problems that arise in polynomial factorization, like computing the norm and trace maps, $X^{q^n+q^{n-1}+\cdots+q+1}$ and $X^{q^n} + X^{q^{n-1}} + \cdots + X^q + X$, with similar speedups.

Thus the modular composition problem emerges as a crucial component of the fastest factoring algorithms (as well as other problems, such as irreducibility testing and constructing irreducible polynomials [Sho94], and manipulating normal bases of finite fields [KS98]). Indeed, if we could compute $f(g(X)) \bmod h(X)$ for degree n polynomials $f, g, h \in \mathbb{F}_q[X]$ in n^α operations, then there are algorithms for factoring degree n polynomial over $\mathbb{F}_q$ using $O(n^{\alpha+0.5+o(1)} + n^{1+o(1)} \log q)$ operations. For comparison, the currently fastest algorithms take either $\widetilde{O}(n^2 + n \log q)$ [vzGS92] or $\widetilde{O}(n^{1.815} \log q)$ [KS98] operations (also, see the more precise accounting and detailed comparisons in Figure 1 of [KS98]).

1.1 Modular composition of polynomials

Using Horner's rule and fast modular polynomial arithmetic, a modular composition can be computed in $\widetilde{O}(n^2)$ operations; one could hope for $\widetilde{O}(n)$ operations. The only asymptotic improvement over the straightforward algorithm is the algorithm of Brent & Kung [BK78], with a slight improvement by Huang & Pan [HP98]. The first takes $\widetilde{O}(n^{1.5} + n^{(\omega+1)/2})$ operations, where ω is the exponent of square matrix multiplication (the best upper bound is currently $\omega < 2.376$ [CW90]), and the second takes $\widetilde{O}(n^{1.5} + n^{\omega_2/2})$ operations where ω_2 is the exponent of $n \times n$ by $n \times n^2$ matrix multiplication, for which Huang & Pan proved an upper bound slightly better than $2.376 + 1$. Devising an improved algorithm for modular composition has been mentioned as an important open problem in [Sho94], [KS98], and [vzGG99].

In this paper, we consider a slight generalization of modular composition, in which we are given a *multivariate polynomial* $f(X_1, X_2, \ldots, X_m) \in \mathbb{F}_q[X_1, X_2, \ldots, X_m]$ and m univariate polynomials

$$g_1(X), \ldots, g_m(X) \in \mathbb{F}_q[X]$$

together with the modulus $h(X) \in F_q[X]$. We wish to compute

$$f(g_1(X), \ldots, g_m(X)) \bmod h(X).$$

The relevant case will always have the individual degrees of f bounded by d, and the g_i and h polynomials of degree at most $N = d^m$. Here the straightforward algorithm takes $\widetilde{O}(N^2)$ operations, and we note that the Brent & Kung "baby-steps/giant-steps" approach generalizes to this case, giving an algorithm that takes $\widetilde{O}(N^{1/5} + N^{\omega_2/2})$ operations.

Our insight is that this modular composition problem and the *multipoint evaluation problem for multivariate polynomials* are essentially equivalent in the sense that an algorithm for one achieving exponent α implies an algorithm for the other with exponent $\alpha + o(1)$, and vice versa. Recall that one can evaluate a degree n *univariate* polynomial at n evaluations points in $O(n \log^2 n)$ operations, for an amortized cost of only $O(\log^2 n)$ operations per evaluation. However, nothing similar is known for multivariate polynomials. The only improvement over the straightforward algorithm is by Nüsken & Ziegler [NZ04], who show how to evaluate bivariate polynomial with individual degrees d at d^2 points in $\widetilde{O}(d^{\omega_2/2+1})$ operations; their algorithm generalizes to the m-variate case where it takes $\widetilde{O}(d^{(\omega_2/2)(m-1)+1})$ operations. Unfortunately, this is not enough to yield an improved algorithm for modular composition via the aforementioned equivalence.

As one can see from the proliferation of ω's and ω_2's in the preceding discussion, all of the non-trivial algorithms for modular composition and multipoint evaluation of multivariate polynomials rely on fast matrix multiplication. As a result, they are currently unlikely to be practical[1] and the same can be said for the asymptotically fastest algorithms for polynomial factorization that use them as subroutines. In addition, notice that the Brent & Kung algorithm for modular composition cannot achieve an exponent better than 1.5, even if we had optimal fast matrix multiplication algorithms.

1.2 A new algorithm in small characteristic

Our main technical contribution is a completely new algorithm for multipoint evaluation of multivariate polynomials, and using the above equivalence, for modular composition, *in small characteristic*. Our algorithm uses an asymptotically optimal number of operations (up to lower order terms), and in particular, solves the modular composition problem relevant for degree n polynomial factorization over $\mathbb{F}_q$ in $O(n^{1+o(1)})$ operations, when the characteristic is at most $n^{o(1)}$. We immediately obtain polynomial factorization algorithms for small characteristic, with running time $O(n^{1.5+o(1)} + n^{1+o(1)} \log q)$, and similar improvements to the fastest known algorithms for a number of other problems (see Section 5). Applications aside, our results represent the first new algorithmic ideas for modular composition since the algorithm of Brent & Kung in 1978 (as the Huang & Pan improvement is really an improved upper bound on the matrix multiplication step; the actual algorithm remains the same).

Another important feature of our algorithm is that it does *not* rely on fast matrix multiplication. The main operations are standard fast univariate polynomial arithmetic operations, and multipoint evaluation and interpolation of univariate polynomials. All of these problem have algorithms that are asymptotically optimal up to lower order terms, and that are very reasonable in practice. In all of the settings we have mentioned where modular composition is the crucial subroutine, the other parts of the algorithms are again these standard fast and practical operations, so the algorithms derived from our new algorithm could be feasible in practice.

[1] However, real implementations successfully use classical matrix multiplication, or even Strassen's algorithm, and find that even then the dominant operations on instances arising in practice are operations on polynomials, as opposed to matrix multiplication [ABS06, p. 25-26].

1.3 Techniques

The reductions between modular composition and multipoint evaluation of multivariate polynomials are not difficult, even though it appears that at least one direction of this equivalence – the one needed for our main result – was not known before (the other direction, reducing multipoint evaluation of multivariate polynomials to modular composition, is just beneath the surface of the results in [NZ04]).

Our algorithm for multipoint evaluation of multivariate polynomials is more involved (and we are only able to make it work in small characteristic), although it utilizes a very natural idea. The idea is to reduce to multipoint evaluation of a *univariate* polynomial *over an extension field*. Suppose we have a multivariate polynomial $f(X_0, X_1, \ldots, X_{m-1})$ with individual degrees $d-1$, with coefficients in $\mathbb{F}_q$. A related univariate polynomial f^* is obtained by the *Kronecker substitution*:

$$f^*(Z) = f(Z, Z^d, Z^{d^2}, \ldots, Z^{d^{m-1}}).$$

A tempting approach is to describe some (efficiently computable) mapping from evaluation points $\alpha = (\alpha_1, \cdots, \alpha_m) \in \mathbb{F}_q^m$ intended for f to evaluation points $\bar{\alpha}$ in an extension field, intended for f^*, with the property that $f(\alpha)$ can be easily recovered from $f^*(\bar{\alpha})$. Then we could perform multipoint evaluation of f by mapping all of the evaluation points to their counterparts in the extension field, and then invoking a fast *univariate* multipoint evaluation algorithm to evaluate f^* at these points.

We are able to make something very close to this strategy work. To do so we need to (1) define f^* by raising to successive powers of a parameter $h \approx dm^2$ instead of d, (2) carefully construct the extension field, and (3) arrange for h to be a power of the characteristic (this is why we need small characteristic) so that we can exploit properties of the Frobenius endomorphism.

A technical requirement of our algorithm is that it needs an element of multiplicative order $h-1$ in $\mathbb{F}_q$. If $\mathbb{F}_q$ does not contain the subfield $\mathbb{F}_h$, such an element does not even exist. As a result, we need to first extend $\mathbb{F}_q$ to guarantee such an element. This complication is not needed in settings where an order-$(h-1)$ element is already available.

1.4 Related work

The special case of modular composition in which $m = 1$ and the modulus $h(X)$ is X^d has an algorithm attributed to Brent & Kung that uses $\widetilde{O}(n^{1.5})$ operations (see Exercise 12.4 in [vzGG99]), and a different algorithm by Bernstein [Ber98] that is faster in small characteristic. This special case is not useful for polynomial factoring (and other applications), because in these applications $h(X)$ ends up being the input polynomial, and modular composition is used as a means of determining its (initially unknown) structure.

The problem of factoring a degree n polynomial in small characteristic has been considered before by Kaltofen & Shoup [KS97]. Their algorithm is advantageous when $q = p^k$ is quite large (i.e., when $k \approx n^{1+\alpha}$, for constant $\alpha > 0$). Its running time is expressed in terms of the running time for modular composition, and so we obtain improvements there as well; see Section 5.

The inspiration for our new algorithm for multivariate multipoint evaluation is two recent works in coding theory: a new variant of Reed-Solomon codes discovered by Parvaresh & Vardy [PV05] and a particular instantiation of these codes used by Guruswami & Rudra [GR06]. The analysis of the decoding algorithm in [PV05] uses the Kronecker substitution to obtain a univariate polynomial from a multivariate polynomial that carries information about the received word. This univariate polynomial is then viewed over an extension field, just as in this work. In [GR06], they utilize a particular extension field with the property that raising a polynomial (that is a canonical representative of a residue class in the extension field) to a Frobenius power is the same as shifting the polynomial by a generator of the field. We use the same trick to

"store" the coordinates of an intended evaluation point in a single extension ring element, and then "access" them by raising to successive Frobenius powers.

1.5 Outline

In Section 2 we give some preliminary definitions and conventions, and formally define the modular composition and multipoint evaluation problem for multivariate polynomials. In Section 3 we give the reductions showing that these two problems are essentially equivalent. Section 4 contains the main technical result – the new algorithm for multipoint evaluation of multivariate polynomials in small characteristic. Section 5 describes the resulting improvements in polynomial factorization algorithms, and some other applications. In this section we also reformulate the algorithm of [KS98] in order to highlight a self-contained open problem, whose resolution would lead to an $O(n^{1+o(1)} \log q)$ polynomial factorization algorithm.

2 Preliminaries

We have already discussed the Kronecker substitution, which can be viewed as a transformation that decreases the number of variables at the expense of increasing the degree. We now define a map that is (in a sense made precise following the definition) the "inverse" of the Kronecker substitution – it increases the number of variables while decreasing the degree:

Definition 2.1. *The map $\psi_{h,\ell}$ from $\mathbb{F}_q[X_0, X_1, \ldots, X_{m-1}]$ to $\mathbb{F}_q[Y_{0,0}, \ldots, Y_{m-1,\ell-1}]$ is defined as follows. Given X^a, write a in base h: $a = \sum_{j \geq 0} a_j h^j$ and define the monomial*

$$M_a(Y_0, \ldots, Y_{\ell-1}) \stackrel{\text{def}}{=} Y_0^{a_0} Y_1^{a_1} \cdots Y_{\ell-1}^{a_{\ell-1}}.$$

The map $\psi_{h,\ell}$ sends X_i^a to $M_a(Y_{i,0}, \ldots, Y_{i,\ell-1})$ and extends multilinearly to $\mathbb{F}_q[X_0, X_1, \ldots, X_{m-1}]$.

Note that $\psi_{h,\ell}(f)$ can be computed in linear time in the size of f, assuming f is presented explicitly by its coefficients. Also note that $\psi_{h,\ell}$ is injective on the set of polynomials with individual degrees at most $h^\ell - 1$. For such a polynomial f, if $g = \psi_{h,\ell}(f)$, then

$$f(X_0, \ldots, X_{m-1}) = g(X_0^{h^0}, X_0^{h^1}, \ldots, X_0^{h^{\ell-1}}, \cdots, X_{m-1}^{h^0}, X_{m-1}^{h^1}, \ldots, X_{m-1}^{h^{\ell-1}}).$$

In this sense, $\psi_{h,\ell}$ is the inverse of the Kronecker substitution.

The problems we are interested in are formally defined below:

Problem 1 (MULTIVARIATE MULTIPOINT EVALUATION**).** *Given $f(X_0, \ldots, X_{m-1})$ in $\mathbb{F}_q[X_0, \ldots, X_{m-1}]$ with individual degrees at most $d-1$, and evaluation points $\alpha_0, \ldots, \alpha_{d^m-1}$ in $\mathbb{F}_q^m$, output $f(\alpha_i)$ for $i = 0, 1, 2, \ldots, d^m - 1$.*

Defining $N \stackrel{\text{def}}{=} d^m$, the straightforward algorithm takes $\Omega(N^2)$ field operations. One may hope for an algorithm that uses only $O(N)$ field operations.

Problem 2 (MODULAR COMPOSITION**).** *Given $f(X_0, \ldots, X_{m-1})$ in $\mathbb{F}_q[X_0, \ldots, X_{m-1}]$ with individual degrees at most $d-1$, and polynomials $g_0(X), \ldots, g_{m-1}(X)$ and $h(X)$, all in $\mathbb{F}_q[X]$ and with degree at most $d^m - 1$, output $f(g_0(X), \ldots, g_{m-1}(X)) \bmod h(X)$.*

Operation	Input	Output	Operations
Multiplication	$f(X), g(X)$ of degree n	$f(X) \cdot g(X)$	$M(n) = O(n \log n)$
Remainder	$f(X), g(X)$ of degree $O(n)$	$f(X) \bmod g(X)$	$O(M(n))$
Evaluation	$f(X)$ of degree n; $\alpha_1, \cdots, \alpha_n$	$f(\alpha_i), i = 1, \ldots n$	$O(M(n) \log n)$
Interpolation	$\alpha_0, \cdots, \alpha_n, \beta_0, \cdots, \beta_n$	$f(X)$ of degree n, $f(\alpha_i) = \beta_i$	$O(M(n) \log n)$

Figure 1: Operation counts for standard operations on univariate polynomials over a commutative ring. For interpolation, we additionally require that $\alpha_i - \alpha_j$ is a unit, for $i \neq j$.

We note that the term "modular composition" more commonly refers to the special case of this problem in which $m = 1$. Again defining $N \stackrel{\text{def}}{=} d^m$, the straightforward algorithm takes $\Omega(N^2)$ field operations. One may hope for an algorithm that uses only $O(N)$ field operations.

For both problems, we sometimes refer to the problem "with parameters d, m" if we need to specify the individual degrees and number of variables explicitly.

Figure 1 gives the running time for standard operations on univariate polynomials that we use in the remainder of the paper. See, e.g. [vzGG99]. In this paper polynomials are always represented explicitly by a list of their coefficients. We use $M(n)$ throughout the paper as the number of operations sufficient to multiply two univariate polynomials of degree n (and we assume $M(O(n)) = O(M(n))$). Thus, when we construct an extension field (or ring) by adjoining an indeterminate X and mod-ing out by a polynomial of degree n, arithmetic operations in the extension field (or ring) take $O(M(n))$ operations in the base field, since they entail the addition or multiplication of degree $n-1$ polynomials followed by a remainder operation involving degree $O(n)$ polynomials.

We use the "soft-oh" notation $\widetilde{O}$ to suppress polylogarithmic factors. When the argument involves several variables, we explicitly record the quantity whose polylog factors we are suppressing by putting it in the subscript, like this: $\widetilde{O}_n(nm^2)$.

3 The reductions

In this section we give the reductions showing (essentially) that MULTIVARIATE MULTIPOINT EVALUATION and MODULAR COMPOSITION are equivalent. We first reduce MODULAR COMPOSITION to MULTIVARIATE MULTIPOINT EVALUATION (this is the direction that we use in order to give our improved algorithm for MODULAR COMPOSITION).

Theorem 3.1. *Given $f(X_0, \ldots, X_{m-1})$ in $\mathbb{F}_q[X_0, \ldots, X_{m-1}]$ with individual degrees at most $d-1$, and polynomials $g_0(X), \ldots, g_{m-1}(X)$ and $h(X)$, all in $\mathbb{F}_q[X]$ and with degree at most $d^m - 1$, there is, for every $d_0 < d$, an algorithm that outputs $f(g_0(X), \ldots, g_{m-1}(X)) \bmod h(X)$ in*

$$\widetilde{O}_{d^m}((d^m + T(d_0, \ell m))d_0)$$

field operations, where $\ell = \lceil \log_{d_0} d \rceil$, and $T(d_0, m_0)$ is the number of field operations to solve MULTIVARIATE MULTIPOINT EVALUATION *with parameters d_0, m_0.*

Proof. Set $N = d^m$. We perform the following steps:

1. Compute $f' = \psi_{d_0,\ell}(f)$.

2. Compute $g_{i,j}(X) \stackrel{\text{def}}{=} g_i(X)^{d_0^j} \bmod h(X)$ for all i, and $j = 0, 1, \ldots, \ell - 1$.

3. Set $R = Nm\ell d_0$, and select R distinct field elements $\beta_0, \ldots, \beta_{R-1}$ (if $q < R$, then we need to work in an extension field containing at least R elements, but this only affects the operation count by logarithmic factors). Compute $\alpha_{i,j,k} \stackrel{\text{def}}{=} g_{i,j}(\beta_k)$ for all i, j, k using fast multipoint evaluation.

4. Compute $f'(\alpha_{0,0,k}, \ldots, \alpha_{m-1,\ell-1,k})$ for $k = 0, \ldots, R-1$.

5. Interpolate to recover $f'(g_{0,0}(X), \ldots, g_{m-1,\ell-1}(X))$ (which is a univariate polynomial of degree less than R) from these evaluations. Output the result modulo $h(X)$.

Correctness follows from the observation that

$$f'(g_{0,0}(X), \ldots, g_{m-1,\ell-1}(X)) \equiv f(g_0(X), \ldots, g_{m-1}(X)) \pmod{h(X)}.$$

The first step takes $O(N)$ time. For each g_i, the second step takes $O(M(N)\log(d_0^j))$ operations to compute $g_i^{d_0^j}$ by repeated squaring, and this happens for $j = 0, 1, 2, \ldots, \ell - 1$, giving an upper bound of at most $O(M(N)\ell^2 \log d_0)$ operations to compute the required powers. This happens for each g_i for a total of $O(M(N)\ell^2 \log d_0 m)$ operations.

The third step takes $O(M(R)(\log R)\ell m)$ operations using fast univariate polynomial evaluation. The fourth step invokes fast multivariate polynomial evaluation at a cost of $T(d_0, m\ell)m\ell d_0$ operations (each invocation of fast multivariate polynomial evaluation can compute $d_0^{m\ell} \geq d^m = N$ evaluations, and we need to repeat this $m\ell d_0$ times to obtain all R evaluations). The final step requires $O(M(R)\log R)$ operations. Note that both of the $\log R$ terms can be removed if the field supports an FFT and the β's are chosen accordingly. □

Corollary 3.2. *Fix parameters d, m. For every $\varepsilon > 0$, if* MULTIVARIATE MULTIPOINT EVALUATION *with parameters $d_0 = d^\varepsilon$ and $m_0 = m/\varepsilon$ can be solved in $\widetilde{O}_{d_0^{m_0}}((d_0^{m_0})^\alpha)$ operations for some constant $\alpha > 1$, then* MODULAR COMPOSITION *with parameters d, m can be solved in $\widetilde{O}_{d^m}((d^m)^{\alpha+\varepsilon})$ operations.*

Now, we reduce MULTIVARIATE MULTIPOINT EVALUATION to MODULAR COMPOSITION, which demonstrates the equivalence of the two problems.

Theorem 3.3. *Given $f(X_0, \ldots, X_{m-1})$ in $\mathbb{F}_q[X_0, \ldots, X_{m-1}]$ with individual degrees at most $d-1$, and evaluation points $\alpha_0, \ldots, \alpha_{d^m-1}$ in $\mathbb{F}_q^m$, there is an algorithm that outputs $f(\alpha_i)$ for $i = 0, 1, \ldots, d^m - 1$, in*

$$\widetilde{O}_{d^m}(d^m + T(d, m))$$

field operations, where $T(d, m)$ is the number of field operations to solve MODULAR COMPOSITION *with parameters d, m.*

Proof. Set $N = d^m$. We perform the following steps:

1. Select distinct field elements $\beta_0, \ldots, \beta_{N-1}$ (if $q < N$, then we need to work in an extension field containing at least N elements, but this only affects the operation count by logarithmic factors). Find $g_i \in \mathbb{F}_q[X]$ for which $g_i(\beta_k) = (\alpha_k)_i$ for all i, k using fast univariate polynomial interpolation.

2. Produce the univariate polynomial $h(X) \stackrel{\text{def}}{=} \prod_k (X - \beta_k)$, and then compute $f(g_0(X) \ldots, g_{m-1}(X))$ modulo $h(X)$.

3. Evaluate this univariate polynomial at $\beta_0, \ldots, \beta_{N-1}$ using fast polynomial evaluation, and output these evaluations.

Correctness follows from the observation that

$$f(g_1(X), \ldots, g_m(X))(\beta_k) = f(\alpha_k)$$

and the same holds when taking the left-hand-side polynomial modulo $h(X)$ since h vanishes on the evaluation points β_k.

The first step takes $O(M(N)\log N)$ operations for each interpolation, and there are m such interpolations. The second step requires $O(M(N)\log N)$ time to compute $h(X)$, and then it invokes fast modular composition at a cost of $T(d,m)$ operations. The final step requires $O(M(N))$ operations. Note that both of the $\log N$ terms can be removed if the field supports an FFT and the β's are chosen accordingly. □

Corollary 3.4. *Fix parameters d, m. If* MODULAR COMPOSITION *with parameters d and m can be solved in $\widetilde{O}_{d^m}((d^m)^\alpha)$ operations for some constant $\alpha > 1$, then* MULTIVARIATE MULTIPOINT EVALUATION *with parameters d, m can be solved in $\widetilde{O}_{d^m}((d^m)^\alpha)$ operations.*

4 An optimal algorithm in small characteristic

In this section we describe our main algorithm – for MULTIVARIATE MULTIPOINT EVALUATION – which leads to a new algorithm for MODULAR COMPOSITION via Theorem 3.1. These algorithms work in small characteristic, and give operation counts that are optimal up to lower order terms.

As described in Section 1.3, our algorithm operates by reducing multipoint evaluation of the target *multivariate* polynomial f to multipoint evaluation of a related *univariate* polynomial f^* obtained by substituting h-th powers of a single variable for the m different variables of f (the "Kronecker substitution"). The given m-variate polynomial f will have coefficients in a field $\mathbb{F}_q$ and the parameter h will be a power of the characteristic. We will actually view f as a polynomial with coefficients in an extension *ring* $R = \mathbb{F}_q[W]/P(W)$ for some polynomial P (not necessarily irreducible over $\mathbb{F}_q$). The reason for this complication is that the algorithm needs a special element η that satisfies two properties:

1. the multiplicative order of η is $h-1$, and
2. $\eta^i - \eta^j$ is invertible for all $i, j \in \{0, 1, 2, \ldots, m-1\}$, with $i \neq j$.

We will construct R so that we can easily get our hands on such a η. If an element of order $h-1$ is already available in $\mathbb{F}_q$, then it automatically satisfies the second property because $\mathbb{F}_q$ is a field, and there is no need to pass to the extension ring R.

We now describe in detail how to construct the extension ring R, and find η. Fix parameters d and m, and a field $\mathbb{F}_q$ with characteristic p. Let $h = p^c$ be the smallest integer power of p that is larger than $m^2 d$. Construct the ring $R = \mathbb{F}_q[W]/P(W)$, where $P(W)$ is a degree c polynomial with coefficients in $\mathbb{F}_p$, that is irreducible over $\mathbb{F}_p$. Notice that $\mathbb{F}_p[W]/P(W) \subseteq R$ and also that $\mathbb{F}_q \subseteq R$, and that these embeddings are easy to compute. Choose η to be a primitive element of the field $\mathbb{F}_p[W]/P(W)$. This η clearly has multiplicative order $h-1$, and because the elements η^i for $i = 0, 1, \ldots, m-1$ are distinct elements of a field, the second property above is also satisfied. Figure 2 depicts the construction of R.

Given the m-variate polynomial f over R, we want to be able to evaluate it at many points in $\mathbb{F}_q^m \subseteq R^m$. Our strategy will be to lift the evaluation points to elements of an extension ring S, evaluate a related univariate polynomial f^* at those points, and then project back to an element of R. We choose the ring S to be the extension ring $R[Z]/E(Z)$, where $E(Z) \stackrel{\text{def}}{=} Z^{h-1} - \eta$. Refer to Figure 2.

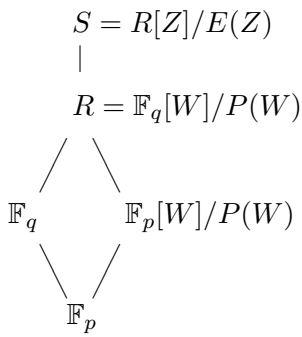

Figure 2: Containment diagram. Our input polynomial will be over $\mathbb{F}_q$, but we view it as a polynomial over the extension ring R. We will end up evaluating a related polynomial at elements of the further extension S.

Let σ be (a power of) the Frobenius endomorphism from R to R, given by $x \mapsto x^h$. The "lift" map $\phi : \mathbb{F}_q^m \to S$ is defined as follows: given $\alpha = (\alpha_0, \ldots, \alpha_{m-1}) \in \mathbb{F}_q^m \subseteq R^m$, $\phi(\alpha)$ is the (residue class whose canonical representative is the) degree $m-1$ polynomial $g_\alpha(Z) \in R[Z]$ which has

$$g_\alpha\left(\eta^i\right) = \sigma^{-i}(\alpha_i) \text{ for } i = 0, 1, 2 \ldots, m-1. \tag{1}$$

Note that g_α is well defined because although σ^i is only an *endomorphism* of R (under which certain elements may have no preimage), we only demand preimages of elements of $\mathbb{F}_q \subseteq R$, and σ^i is an *automorphism* when restricted to $\mathbb{F}_q$.

The "project" map $\pi : S \to R$ that recovers the evaluation of the original multivariate polynomial f from an evaluation of the univariate polynomial f^* is defined as follows: given an element of S whose canonical representative is the degree $< h-1$ polynomial $g(Z) \in R[Z]$, $\pi(g)$ is the evaluation $g(1)$.

Our main lemma shows how to recover the evaluation of the m-variate polynomial f at a point $\alpha \in \mathbb{F}_q^m \subseteq R^m$, from the evaluation of the univariate polynomial f^* at an element of the extension ring S.

Lemma 4.1. *Let $f(X_0, X_1, \ldots, X_{m-1})$ be a polynomial in $\mathbb{F}_q[X_0, X_1, \ldots, X_{m-1}]$ with individual degrees $d-1$, and suppose $\mathbb{F}_q$ has characteristic p. Define h, R, E, S, ϕ, π as above, and define the univariate polynomial $f^*(Y) \in S[Y]$ by:*

$$f^*(Y) \stackrel{\text{def}}{=} f(Y, Y^h, Y^{h^2}, \ldots, Y^{h^{m-1}}).$$

For every $\alpha \in \mathbb{F}_q^m \subseteq R^m$, the following identity holds: $\pi(f^*(\phi(\alpha))) = f(\alpha)$.

Proof. Fix $\phi(\alpha)$, which is an element of $R[Z]/E(Z)$. Let $g_\alpha(Z) \in R[Z]$ be its (degree $m-1$) canonical representative, and denote by $\sigma^i(g_\alpha)$ the polynomial obtained by applying σ^i to the coefficients of g_α. Then we have:

$$\begin{aligned}
(g_\alpha(Z))^{h^i} &= \sigma^i(g_\alpha)(Z^{h^i}) \\
&= \sigma^i(g_\alpha)(Z^{h^i-1}Z) \\
&\equiv \sigma^i(g_\alpha)(\eta^{(h^i-1)/(h-1)}Z) \pmod{E(Z)} \\
&= \sigma^i(g_\alpha)(\eta^i Z),
\end{aligned}$$

where the last equality used the fact that η has order $h-1$ and so it is fixed under σ. For convenience, let us denote by $g_\alpha^{(i)}(Z)$ the polynomial $(g_\alpha(Z))^{h^i} \bmod E(Z)$. A crucial point that we will use shortly is that $\deg(g_\alpha^{(i)}) = \deg(g_\alpha)$. The above equation implies that

$$g_\alpha^{(i)}(1) = \sigma^i(g_\alpha)\left(\eta^i\right) = \sigma^i\left(g_\alpha\left(\sigma^{-i}\eta^i\right)\right) = \sigma^i\left(g_\alpha\left(\eta^i\right)\right) = \sigma^i(\sigma^{-i}\alpha_i) = \alpha_i, \tag{2}$$

where the third equality again used the fact that η is fixed under σ, and the fourth equality used Eq. (1).

When we evaluate the polynomial f^* at the element of S whose canonical representative is g_α we get the element of S whose canonical representative is:

$$f(g_\alpha^{(0)}(Z), g_\alpha^{(1)}(Z), \ldots, g_\alpha^{(m-1)}(Z)) \bmod E(Z).$$

Now f is a polynomial with total degree at most dm, and each $g_\alpha^{(i)}$ is a polynomial of degree at most $m-1$. Therefore, since E has degree at least $dm^2 > dm(m-1)$, this polynomial is just

$$f(g_\alpha^{(0)}(Z), g_\alpha^{(1)}(Z), \ldots, g_\alpha^{(m-1)}(Z)),$$

and evaluating at 1 gives (using Eq. (2)):

$$f(g_\alpha^{(0)}(1), g_\alpha^{(1)}(1), \ldots, g_\alpha^{(m-1)}(1)) = f(\alpha_0, \alpha_1, \ldots, \alpha_{m-1})$$

as claimed. □

The next theorem applies the strategy we have developed above to the MULTIVARIATE MULTIPOINT EVALUATION problem. It achieves an optimal operation count (up to lower order terms) when the characteristic $p = d^{o(1)}$, and when m is not too small and not too large (to be precise, we need $\omega(1) \le m \le d^{o(1)}/\log m$). When used via Theorem 3.1 for the case of main interest (modular composition of univariate polynomials) we are able to choose the parameters d and m of the invoked MULTIVARIATE MULTIPOINT EVALUATION instance to satisfy these constraints.

Theorem 4.2. *Given $f(X_0, \ldots, X_{m-1})$ in $\mathbb{F}_q[X_0, \ldots, X_{m-1}]$ with individual degrees at most $d-1$, and evaluation points $\alpha_0, \ldots, \alpha_{d^m-1}$ in $\mathbb{F}_q^m$, there is an algorithm that outputs $f(\alpha_i)$ for $i = 0, 1, 2, \ldots, d^m-1$, in*

$$\widetilde{O}_{d^m}(d^m(m^2p)^m \mathbf{poly}(d,p))$$

field operations.

Proof. Set $N = d^m$. We perform the following steps:

1. Choose $h = p^c$ to be the smallest power of p that is at least $m^2 d$. Find a degree c irreducible polynomial $P(W)$ over $\mathbb{F}_p$, and a primitive element η of $\mathbb{F}_p[W]/P(W)$. Define the ring $R = \mathbb{F}_q[W]/P(W)$, and the ring $S = R[Z]/E(Z)$, where $E(Z) = Z^{h-1} - \eta$, as above.
2. For $i = 0, 1, 2, \ldots, N-1$, compute the canonical representative of $\phi(\alpha_i)$: the degree $m-1$ polynomial $g_{\alpha_i}(Z) \in R[Z]$.
3. Produce the univariate polynomial $f^*(Y) = f(Y, Y^h, Y^{h^2}, \ldots, Y^{h^{m-1}})$ over S.
4. Evaluate f^* at the points $g_{\alpha_i}(Z)$, and for each evaluation apply π to recover $f(\alpha_i)$.

Step 1 requires constructing the field $\mathbb{F}_h$ and finding a primitive element. This can be done by brute force in time poly(h), although much better algorithms are available.

Each polynomial g_{α_i} computed in Step 2 requires the following operations (recall Eq. 1): first, we need to compute $\sigma^{-j}(\alpha_i)_j$ for $j = 0, 1, \ldots, m-1$. A single field operation gives us $(\alpha_i)_j^{-1}$, and then using repeated squaring we can apply σ^j using at most $O(\log(h^m))$ $\mathbb{F}_q$-operations. The overall cost of doing this for all i is $O(Nm^2 \log h)$. Next, we perform N polynomial interpolations in R, each costing $O(M(m) \log m)$ operations in R, or $O(M(m) \log mM(c))$ operations in $\mathbb{F}_q$. Note that for every two interpolation points η^i, η^j, the difference $\eta^i - \eta^j$ is a unit in R (since η is an element of $\mathbb{F}_p(W)/P(W)$ which is a field). This is required for the interpolation step. The total cost for Step 2 is

$$O(N(m^2 \log h + M(m) \log mM(c)))$$

$\mathbb{F}_q$-operations.

Step 4 is a univariate multiple evaluation problem. We have N elements of S, and a univariate polynomial f^* over S, of degree at most dmh^m (and this quantity is greater than N). Using fast multipoint evaluation then, this step requires $O(M(dmh^m) \log(dmh^m))$ operations in S, or

$$O(M(dmh^m) \log(dmh^m) M(h) M(c))$$

$\mathbb{F}_q$-operations. Recalling that $h \leq dm^2 p$, and that poly(m) factors are polylogarithmic in the main size measure d^m (and so are suppressed by the soft-oh notation) the claimed bound follows. □

Corollary 4.3. *The* MODULAR COMPOSITION *problem with parameters* $d, 1$ *can be solved in*

$$O(d^{1+o(1)})$$

operations, provided the characteristic $p = d^{o(1)}$.

Proof. It is sufficient to be able to choose, for any $\varepsilon > 0$, the parameter $d_0 \leq d^\varepsilon$ so that $m_0 \stackrel{\text{def}}{=} (\log d)/(\log d_0)$ satisfies: $m_0^{2m_0} \leq d^\varepsilon$ and $p^{m_0} \leq d^\varepsilon$. We then apply Theorem 3.1. For sufficiently large d (and using the assumption that $p \leq d^{o(1)}$) these demands are met by choosing $d_0 = \max\{(\log d)^{2/\varepsilon}, p^{1/\varepsilon}\}$. □

5 Applications

In this section we describe some improved algorithms that arise as a consequence of our new algorithm for modular composition. Of primary interest is univariate polynomial factorization, so we begin with that. In this section, we let $C(n)$ denote the number of operations sufficient to perform a modular composition with parameters $n, 1$. As shown in the previous section, in small characteristic, we now have $C(n) = O(n^{1+o(1)})$.

5.1 Polynomial factorization

There are three stages in variants of the Cantor-Zassenhaus algorithm for factoring a degree n univariate polynomial over $\mathbb{F}_q$: square-free factorization, distinct-degree factorization, and equal-degree factorization (see [vzGG99] for a thorough presentation).

The first stage, square-free factorization, can be performed in $O(n^{1+o(1)} + n \log q)$ operations, using an algorithm attributed by [KS98] to Yun. The second stage, distinct-degree factorization, has a deterministic algorithm due to Kaltofen & Shoup [KS98] that takes

$$O(C(n)n^{0.5+o(1)} + M(n) \log q).$$

The third stage, equal-degree factorization, has a randomized algorithm due to von zur Gathen & Shoup [vzGS92] that takes an expected number $O(M(n)\log n + C(n)\log n + M(n)\log q)$ operations.

Notice that with our improvements (i.e., $C(n) = O(n^{1+o(1)})$) in small characteristic, the first and third stages use $O(n^{1+o(1)} + n^{1+o(1)}\log q)$ operations and the second stage improves to $O(n^{1.5+o(1)} + n^{1+o(1)}\log q)$. The second stage remains the barrier to an "exponent 1" algorithm, so we describe the algorithm of Kaltofen & Shoup in enough detail here (and in a manner differing somewhat from the original) to highlight a self-contained open problem whose resolution would improve its efficiency to $O(n^{1+o(1)} + n^{1+o(1)}\log q)$ operations. We will also see the critical role played by modular composition in this algorithm.

The problem we are trying to solve is:

Problem 3 (DISTINCT-DEGREE FACTORIZATION**).** *Given a monic, squarefree polynomial $f \in \mathbb{F}_q[X]$ of degree n, output $f_1, f_2, \ldots, f_n \in \mathbb{F}_q[X]$ where f_i is either 1 or the product of degree i irreducible polynomials, and $f_1 f_2 \cdots f_n = f$.*

The crucial (standard) algebraic fact used in these algorithms is:

Proposition 5.1. *The polynomial $s_i(X) \stackrel{\text{def}}{=} (X^{q^i} - X) \in \mathbb{F}_q[X]$ is the product of all monic irreducible polynomials over $\mathbb{F}_q$ whose degree divides i.*

Therefore, computing $\gcd(s_i(X), f(X))$ splits off those irreducible factors of f whose degrees divide i. In preparing the polynomial $s_i(X)$ for this purpose, we are free to compute it modulo $f(X)$.

The main step in the algorithm for distinct-degree factorization will be to split the input polynomial f into two non-constant polynomials $f_1 f_2 \cdots f_m$ and $f_{m+1} f_{m+2} \cdots f_n$ for some $m \in \{1, 2, \ldots, n\}$. One could do this by computing $\gcd(s_i(X), f(X))$ for $i = 1, 2, \ldots, n$ and stopping at the first non-trivial gcd, but in the worst case, a non-trivial split will not be found until $i \approx n/2$ which spoils any chance of a subquadratic algorithm. Instead, we will perform a "binary search:" we begin with $m = n/2$, and if this does not yield a non-trivial split, we proceed to either $m = n/4$ or $m = 3n/4$ depending on whether $f_1 f_2 \cdots f_{n/2}$ equals f or 1, and so on.

For this purpose we need to be able to solve the following sub-problem, which gives us the polynomials needed for the "splits" in the above binary-search strategy (and note that for our intended application we do not care if the $s_i(X)$ factors are repeated, which explains the a_i's below):

Problem 4. *Given a monic, squarefree polynomial $f \in \mathbb{F}_q[X]$ of degree n, a positive integer m, and the polynomial $X^q \bmod f(X)$, compute the polynomial*

$$s_1(X)^{a_1} \cdot s_2(X)^{a_2} \cdot \cdots \cdot s_m(X)^{a_m} \bmod f(X) = \prod_{i=1}^{m} (X^{q^i} - X)^{a_i} \bmod f(X)$$

for any positive integers a_i.

Now, it is easy to see that this problem can be solved in $O((C(n) + M(n))m)$ operations: with m successive modular compositions with X^q, we can obtain $X^{q^i} \bmod f(X)$ for $i = 1, 2, \ldots, m$, and then m further polynomial additions and multiplications modulo f suffice to compute $\prod_{i=1}^{m}(X^{q^i} - X) \bmod f(X)$.

Kaltofen & Shoup describe a clever algorithm that reduces the exponent on m from 1 to 0.5:

Lemma 5.2 (implicit in [KS98]). *Problem 4 can be solved in $O(C(n)\sqrt{m} + M(n)\sqrt{m}\log\sqrt{m})$ operations.*

Proof. First, compute X^{q^i} for $i = 0, 1, 2, \ldots, \sqrt{m} - 1$; and then $X^{q^{j\sqrt{m}}}$ for $j = 1, 2, \ldots, \sqrt{m}$, all modulo $f(X)$. This requires $O(C(n)\sqrt{m})$ operations (since we are given $X^q \bmod f(X)$ to begin with). Form the degree $\sqrt{m}$ polynomial $P(Z)$ over the ring $\mathbb{F}_q[X]/f(X)$ defined as:

$$P(Z) \stackrel{\text{def}}{=} \prod_{i=0}^{\sqrt{m}-1} (Z - X^{q^i}) \bmod f(X).$$

This requires $O(\sqrt{m} \log \sqrt{m})$ operations in the ring, or $O(M(n)\sqrt{m} \log \sqrt{m})$ operations in $\mathbb{F}_q$. Finally, evaluate $P(Z)$ at the elements $X^{q^{j\sqrt{m}}} \bmod f(X)$ for $j = 1, 2, \ldots, \sqrt{m}$, and take the product of these evaluations modulo $f(X)$, yielding:

$$\prod_{j=1}^{\sqrt{m}} \prod_{i=0}^{\sqrt{m}-1} (X^{q^{j\sqrt{m}}} - X^{q^i}) \bmod f(X)$$

which equals:

$$\prod_{j=1}^{\sqrt{m}} \prod_{i=0}^{\sqrt{m}-1} (X^{q^{j\sqrt{m}-i}} - X)^{q^i} \bmod f(X),$$

which is a polynomial of the desired form (the a_i are various powers of q). Using fast multipoint evaluation this final step entails $O(M(\sqrt{m}) \log \sqrt{m})$ operations in the ring, or $O(M(n)M(\sqrt{m}) \log \sqrt{m})$ operations in $\mathbb{F}_q$. □

We consider it a very interesting open problem to devise an algorithm for Problem 4 that takes only $O(n^{1+o(1)}m^{o(1)})$ operations (under the assumption that $C(n) = O(n^{1+o(1)})$).

Using Problem 4 as a subroutine, it is not hard to describe a fast algorithm for DISTINCT-DEGREE FACTORIZATION:

Theorem 5.3. *If Problem 4 can be solved in $O(n^\alpha m^\beta)$ operations (with $\alpha > 1$), then there is an algorithm for* DISTINCT-DEGREE FACTORIZATION *that uses $\widetilde{O}(n^{\alpha+\beta} + M(n) \log q)$ operations.*

Proof. We first prepare the polynomial $X^q \bmod f(X)$ needed as input to Problem 4, at a cost of $O(M(n) \log q)$ operations.

Now, in addition to the input of a squarefree $f(X) \in \mathbb{F}_q[X]$ of degree n, we assume we are given a range within which we know all of the degrees of the irreducible factors of f lie. Initially, this is just $1 \ldots n$.

If the range consists of only a single integer, then we can output $f(X)$ itself and halt. Otherwise, set m to the midpoint of this range, and compute a polynomial as specified in Problem 4; call this polynomial $S(X)$. Compute $\gcd(S(X), f(X))$. If this gcd is $f(X)$, then we reduce the range to the first half and recurse; if this gcd is a constant polynomial, then we reduce the range to the second half and recurse; if this gcd is a non-trivial polynomial $f_{\text{lower}}(X)$, then we compute $f_{\text{upper}}(X) = f(X)/f_{\text{lower}}(X)$, and these two polynomials represent a successful "split." Notice that $\deg(f_{\text{lower}}) + \deg(f_{\text{upper}}) = \deg(f)$. We now recurse on f_{lower} (with the range reduced to the first half) and f_{upper} (with the range reduced to the second half).

We now analyze the operation count of this recursive algorithm when factoring a degree n input polynomial. Notice that we never set m larger than n throughout the entire algorithm, so we will pessimistically assume it is always n to simplify the analysis.

Let $T(n', r)$ denote the operation count of the procedure, when called with a polynomial of degree n' and range of size r. If $r = 1$, the cost is zero. Otherwise, the procedure solves Problem 4 at a cost of at most $c_1 n'^{\alpha} n^{\beta}$, and the other operations before the recursive call (a gcd, and possibly a polynomial division) cost at most $c_2 n' \log^2 n'$ for some constants c_1, c_2. Set $c = c_1 + c_2$.

We will prove that for all $T(n', r)$ with $n', r \leq n$,

$$T(n', r) \leq c n'^{\alpha} \log^2 n' n^{\beta} \log r,$$

by induction on r. The base case, when $r = 1$, is clear. In general we have that

$$T(n', r) \leq c_1 n'^{\alpha} n^{\beta} + c_2 n' \log^2 n' + \max_{1<i<n'} \begin{cases} T(n', r/2) \\ T(i, r/2) + T(n' - i, r/2) \end{cases}$$

where the two lines in the inequality correspond to the cases that result in recursive calls. In the first case we have:

$$\begin{aligned} &c_1 n'^{\alpha} n^{\beta} + c_2 n' \log^2 n' + T(n', r/2) \\ &\qquad \leq c_1 n'^{\alpha} n^{\beta} + c_2 n' \log^2 n' + c n'^{\alpha} \log^2 n' n^{\beta} (\log r - 1) \leq c n'^{\alpha} \log^2 n' n^{\beta} \log r \end{aligned}$$

as required. In the second case, we have:

$$\begin{aligned} &c_1 n'^{\alpha} n^{\beta} + c_2 n' \log^2 n' + T(i, r/2) + T(n' - i, r/2) \\ &\qquad \leq c_1 n'^{\alpha} n^{\beta} + c_2 n' \log^2 n' + c[i^{\alpha} \log^2 i + (n' - i)^{\alpha} \log^2 (n' - i)] n^{\beta} (\log r - 1) \\ &\qquad\qquad \leq c_1 n'^{\alpha} n^{\beta} + c_2 n' \log^2 n' + c[n'^{\alpha} \log^2 n'] n^{\beta} (\log r - 1) \leq c n'^{\alpha} \log^2 n' n^{\beta} \log r \end{aligned}$$

as required. The claimed upper bound in the theorem follows by considering $T(n, n)$. □

We now see how our new modular composition algorithm yields the fastest univariate factorization algorithm in small characteristic:

Theorem 5.4. *There is an algorithm that returns the irreducible factors of a degree n polynomial $f \in \mathbb{F}_q[X]$ and uses $O(n^{1.5+o(1)} + n^{1+o(1)} \log q)$ operations, if the characteristic is at most $n^{o(1)}$.*

Proof. As noted the first and third phases already fall within this bound. Plugging Corollary 4.3 into Lemma 5.2 yields an algorithm for Problem 4 using $O(n^{1+o(1)} m^{0.5+o(1)})$ operations. Theorem 5.3 then yields the claimed result. □

5.2 Other problems

Modular composition is a core operation in a variety of algebraic algorithms. We briefly outline improvements our new algorithm implies for a few of these problems (note that this section is not intended to be an exhaustive survey of such problems), in small characteristic.

- As mentioned in the introduction, Kaltofen & Shoup [KS97] considered the polynomial factorization problem in small characteristic. They give a new algorithm for equal-degree factorization of a degree n polynomial over $\mathbb{F}_q$ with $q = p^k$, that uses

$$\widetilde{O}_n(\log k(nC_p(k) + C_q(n)M(k) + M(n)M(k)))$$

operations in $\mathbb{F}_p$. Here $C_p(\cdot)$ and $C_q(\cdot)$ distinguish between modular compositions over the field $\mathbb{F}_p$ and the field $\mathbb{F}_q$. Our algorithm improves $C_q(\cdot)$, and consequently the running time of their algorithm, when $p = n^{o(1)}$. In fact, in many of the algorithms discussed in this section, there is a $O(M(n) \log q)$ term that arises solely from computing X^q modulo some degree n polynomial. One of the insights of [KS97] is that when $q = p^k$, and when counting $\mathbb{F}_p$ operations, this term can be improved. As above, their approach uses modular composition over both $\mathbb{F}_p$ and $\mathbb{F}_q$. Our algorithm for modular composition in small characteristic improves the operation count for the $\mathbb{F}_q$-modular compositions, and consequently realizes improvements in algorithms with the $O(M(n) \log q)$ term in their running time, when one does the accounting over $\mathbb{F}_p$, as in [KS97].

- Their are simple algorithms for testing irreducibility [Rab80] that can be implemented to take

$$O(M(n) \log q + C(n)(\log n)\delta(n))$$

operations (see Algorithm 14.36 in [vzGG99]), where $\delta(n)$ is the number of distinct prime factors of n (so $\delta(n) \leq \log n$). We improve the running time in small characteristic, yielding the asymptotically fastest irreducibility test in this setting.

- Kaltofen & Shoup [KS98] also study several problems related to manipulating normal bases: the problem of *basis selection* (given a degree n irreducible $f(X)$, find a normal element of $\mathbb{F}_q[X]/f(X)$); and the problems of converting to a normal basis representation from a power-basis representation, and vice-versa. Their algorithms use modular composition as well as fast matrix multiplication. In small characteristic we obtain improvements to the running times, but stating the precise operation counts is a bit messy in this case, so the reader is referred to [KS98].

- By the "transposition principle" (see the discussion in, e.g., [KS98]), the complexity of the *transpose* of modular composition, "modular power projection[2]," has complexity at most a constant factor larger than the complexity of modular composition (provided the algorithm for modular composition computes only linear forms in the polynomial f, which ours does). As a result, we obtain improved algorithms for this transposed problem in small characteristic, which in turn implies faster algorithms for computing minimal polynomials in these fields, via the algorithms in [Sho99].

6 Open problems

We consider it an appealing and important open problem to give an algorithm for Problem 4 using only

$$O(n^{1+o(1)}m^{o(1)})$$

operations (under the assumption that $C(n) = O(n^{1+o(1)})$). Improvements in small characteristic are interesting, as well as in arbitrary characteristic. In fact, any algorithm with operation count $O(n^{1+o(1)}m^{\beta})$ for constant $\beta < 1/2$ seems to require a new idea.

It remains open to obtain any improvement over the Brent & Kung and Huang & Pan algorithms for modular composition in arbitrary characteristic. Perhaps a fruitful route is to focus on the multipoint evaluation problem for multivariate polynomials (which is essentially equivalent via Theorems 3.1 and 3.3), but seems to have received comparatively less attention.

[2]Modular power projection is the problem: given a linear form $\pi : \mathbb{F}_q^n \to \mathbb{F}_q$, and degree $n-1$ polynomials $g(X), h(X) \in \mathbb{F}_q[X]$, compute $\pi(g(X)^i \bmod h(X))$ for $i = 0, 1, \ldots, n-1$.

It would also be interesting to adapt our algorithms so that they work in a commutative *ring* of small characteristic. Currently we require a field (see the discussion following Eq. 1).

Finally, the general idea of dealing with multivariate polynomials by lifting to an extension ring and working with a related univariate polynomial seems to be a very natural one. We wonder if this strategy (which is successfully implemented here, in small characteristic) could be useful elsewhere.

7 Acknowledgements

We thank Henry Cohn, Joachim von zur Gathen, Erich Kaltofen, and Eyal Rozenman for useful discussions, and Éric Schost for helpful comments on a draft of this paper.

References

[ABS06] B. Salvy A. Bostan, P. Flajolet and E. Schost. Fast computation of special resultants. *Journal of Symbolic Computation*, 41(1):1–29, 2006.

[Ber70] E. R. Berlekamp. Factoring polynomials over large finite fields. *Mathematics of Computation*, 24(111):713, 1970.

[Ber98] D. J. Bernstein. Composing power series over a finite ring in essentially linear time. *J. Symb. Comput.*, 26(3):339–341, 1998.

[BK78] R. P. Brent and H. T. Kung. Fast algorithms for manipulating formal power series. *J. ACM*, 25(4):581–595, 1978.

[CW90] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *J. Symb. Comput.*, 9(3):251–280, 1990.

[CZ81] D.G. Cantor and H. Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, 36(154):587, 1981.

[GR06] V. Guruswami and A. Rudra. Explicit capacity-achieving list-decodable codes. In Jon M. Kleinberg, editor, *STOC*, pages 1–10. ACM, 2006.

[HP98] X. Huang and V. Y. Pan. Fast rectangular matrix multiplication and applications. *J. Complexity*, 14(2):257–299, 1998.

[Kal03] E. Kaltofen. Polynomial factorization: a success story. In J. Rafael Sendra, editor, *ISSAC*, pages 3–4. ACM, 2003.

[KS97] E. Kaltofen and V. Shoup. Fast polynomial factorization over high algebraic extensions of finite fields. In *ISSAC*, pages 184–188, 1997.

[KS98] E. Kaltofen and V. Shoup. Subquadratic-time factoring of polynomials over finite fields. *Mathematics of Computation*, 67(223):1179–1197, 1998.

[NZ04] M. Nüsken and M. Ziegler. Fast multipoint evaluation of bivariate polynomials. In Susanne Albers and Tomasz Radzik, editors, *ESA*, volume 3221 of *Lecture Notes in Computer Science*, pages 544–555. Springer, 2004.

[PV05] F. Parvaresh and A. Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *FOCS*, pages 285–294. IEEE Computer Society, 2005.

[Rab80] M. O. Rabin. Probabilistic algorithms in finite fields. *SIAM J. Comput.*, 9(2):273–280, 1980.

[Sho94] V. Shoup. Fast construction of irreducible polynomials over finite fields. *J. Symb. Comput.*, 17(5):371–391, 1994.

[Sho99] V. Shoup. Efficient computation of minimal polynomials in algebraic extensions of finite fields. In *ISSAC*, pages 53–58, 1999.

[vzG06] J. von zur Gathen. Who was who in polynomial factorization. In Barry M. Trager, editor, *ISSAC*, page 2. ACM, 2006.

[vzGG99] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999.

[vzGP01] J. von zur Gathen and D. Panario. Factoring polynomials over finite fields: A survey. *J. Symb. Comput.*, 31(1/2):3–17, 2001.

[vzGS92] J. von zur Gathen and V. Shoup. Computing Frobenius maps and factoring polynomials. *Computational Complexity*, 2:187–224, 1992.