Root lifting techniques and applications to list decoding

Muhammad F. I. Chowdhury
The University of Western Ontario
London, Canada
mchowdh3@csd.uwo.ca

Romain Lebreton
École polytechnique
France
lebreton@lix.polytechnique.fr

Abstract

Motivatived by Guruswami and Rudra's construction of folded Reed-Solomon codes, we give algorithms to solve functional equations of the form $Q(x, f(x), f(\gamma x)) = 0$, where Q is a trivariate polynomial. We compare two approaches, one based on Newton's iteration and the second using relaxed series techniques.

1 Introduction

In a celebrated paper [6], Sudan introduced a list decoding algorithm for Reed-Solomon codes based on bivariate interpolation and root finding techniques. The techniques were then refined by Guruswami-Sudan [4], Parvaresh-Vardy [5] and in 2008, Guruswami and Rudra [3] achieved close to the information-theoretic limit by means of *folded* Reed-Solomon codes. Let $\mathbb{F}$ be a finite field and let γ be a primitive element of $\mathbb{F}$. The message polynomial $f(x)$ will be transmitted as the sequence $f(\gamma^i)$ for $i \in \{1, \ldots, n\}$. Let y be the received set and let $s \geq 2$ be a "folding" parameter; then, the decoding algorithm does the following

1. (*interpolation)* Find a multivariate polynomial $Q(x, z_1, \ldots, z_s)$ (with suitable degree properties) such that $Q(\gamma^{si}, y_{si+1}, \ldots, y_{si+s}) = 0$ holds for all i, with multiplicity m;
2. (*root-finding)* Return the polynomials $f(x)$ such that $Q(x, f(x), f(\gamma x), \ldots, f(\gamma^{s-1}x)) = 0$.

2 Lifting techniques

In this work we consider the second step, root-finding, by means of lifting techniques. For this first study, we consider only situations in three variables (that is, $s = 2$), and we also assume that the multiplicity m of each root is 1. The former assumption can easily be lifted; the latter would require more work (since it requires some desingularization process).

Let $Q(x, z_1, z_2)$ be the polynomial that we obtained during the interpolation step. Our goal here is to construct a polynomial $f(x)$ such that $Q(x, f(x), f(\gamma x)) = 0$. We will assume that $f(0) = 0$; this is actually not a real restriction, since we can impose it on our message polynomials without loss of generality.

We present two algorithms: one using a suitable version of Newton's iteration (similar to Augot-Pequet's approach for Sudan's list decoding algorithm [1]), the other one using van der Hoeven's relaxed techniques.

Newton iteration. The idea behind this approach is classical: assuming that we know $f_0 = f \bmod x^\ell$, we want to compute f at a higher precision, about 2ℓ, by solving a linearized equation. This is done by means of a Taylor expansion: writing $f = f_0 + h$, we obtain

$$\frac{\partial Q}{\partial z_2}(x, f_0(x), f_0(\gamma x))h(\gamma x) + \frac{\partial Q}{\partial z_1}(x, f_0(x), f_0(\gamma x))h(x) = -Q(x, f_0(x), f_0(\gamma x)) \mod x^{2\ell}.$$

If we define the γ-derivative

$$E : f \mapsto \frac{f(\gamma x) - f(x)}{x},$$

the former equation takes the form $A(x)E(h) + B(x)h = C(x)$, for some suitable A, B, C. The similarity between this equation and first-order linear differential equations allows us to propose an algorithm very close to Brent and Kung's algorithm for differential equations [2]. By construction, the equation is singular (that is, $A(0) = 0$), but it is possible to overcome this issue. The resulting algorithm runs in time $O(\mathsf{M}(n))$ to compute $f \bmod x^n$, where M denotes as usual a function such that degree-n polynomials can be multiplied in $\mathsf{M}(n)$ base field operations.

The relaxed algorithm. In [7], van der Hoeven introduced the relaxed model of multiplication, that allows for "lazy" polynomial multiplication with an amortized quasi-linear complexity. This model allows one to solve fixed-point equations of the form of $f(x) = \phi(f(x))$ where ϕ is an operator such that the first n coefficients of $\phi(f(x))$ depend only on the first $n-1$ coefficients of $f(x)$.

We show how to transform the equation $Q(x, f(x), f(\gamma x))$ into such a fixed-point equation. As a result, we are able to compute $f \bmod x^n$ in time $O(\mathsf{R}(n))$, where R is the cost of relaxed multiplication. In general, we have $\mathsf{R}(n) = O(\mathsf{M}(n)\log(n))$; for multiplication algorithms such as Karatsuba's, we have $\mathsf{R}(n) = O(\mathsf{M}(n))$, so that this approach is competitive with the one based on Newton iteration.

References

[1] D. Augot and L. Pecquet. A Hensel lifting to replace factorization in list-decoding of algebraic-geometric and Reed-Solomon codes. *IEEE Trans. Inf. Theory*, 46(7):2605–2614, 2000.

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

[3] V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Trans. Inf. Theory*, 54(1):135 –150, 2008.

[4] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. *IEEE Trans. Inf. Theory*, 45(6):1757 – 1767, 1999.

[5] F. Parvaresh and A. Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *FOCS'05*, pages 285 – 294. IEEE Computer Society, 2005.

[6] Madhu Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *J. Complexity*, 13:180–193, 1997.

[7] J. van Der Hoeven. Relax, but don't be too lazy. *J. Symbolic Computation*, 34:479–542, 2002.