Journal of Symbolic Computation 75 (2016) 4–24

Contents lists available at ScienceDirect

Journal of Symbolic Computation

www.elsevier.com/locate/jsc

# Faster sparse multivariate polynomial interpolation of straight-line programs

Andrew Arnold [a], Mark Giesbrecht [a], Daniel S. Roche [b]

[a] Cheriton School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada
[b] Computer Science Department, United States Naval Academy, Annapolis, MD, USA

ARTICLE INFO

*Article history:*
Received 9 December 2014
Accepted 10 July 2015
Available online 10 November 2015

*Keywords:*
Sparse interpolation
Complexity
Randomized algorithms
Straight-line program

ABSTRACT

Given a straight-line program whose output is a polynomial function of the inputs, we present a new algorithm to compute a concise representation of that unknown function. Our algorithm can handle any case where the unknown function is a multivariate polynomial, with coefficients in an arbitrary finite field, and with a reasonable number of nonzero terms but possibly very large degree. It is competitive with previously known sparse interpolation algorithms that work over an arbitrary finite field, and provides an improvement when there are a large number of variables.

Published by Elsevier Ltd.

## 1. Introduction

We consider the problem of interpolating a sparse multivariate polynomial $F$ over $\mathbb{F}_q$, the finite field of size $q$:

$$F = \sum_{\ell=1}^{t} c_\ell z_1^{e_{\ell 1}} z_2^{e_{\ell 2}} \cdots z_n^{e_{\ell n}} \in \mathbb{F}_q[z_1, \ldots, z_n]. \quad (1)$$

We suppose $F$ is given by a *Straight-Line Program* (SLP), a list of simple instructions performing operations $+$, $-$ and $\times$ on inputs and previously computed values, which evaluates the polynomial at any point. We further suppose we are given bounds $D > \max_j \deg_{z_j}(F)$ and $T \geq t$. It is expected that the

*E-mail addresses:* a4arnold@uwaterloo.ca (A. Arnold), mwg@uwaterloo.ca (M. Giesbrecht), roche@usna.edu (D.S. Roche).
*URLs:* http://www.AndrewArnold.ca (A. Arnold), http://www.uwaterloo.ca/~mwg (M. Giesbrecht), http://www.usna.edu/cs/roche (D.S. Roche).

http://dx.doi.org/10.1016/j.jsc.2015.11.005
0747-7171/Published by Elsevier Ltd.

bound $T$ tells us that $F$ is *sparse*, i.e., that $T \ll D^n$, the maximum number of terms. The goal of our interpolation algorithm is to obtain the $t$ nonzero coefficients $c_\ell \in \mathbb{F}_q$ and corresponding exponents $\mathbf{e}_\ell = (e_{\ell_1}, \ldots, e_{\ell_n}) \in \mathbb{Z}^n$ of $F$. Our contribution is as follows.

**Theorem 1.** *Let $F \in \mathbb{F}_q[z_1, \ldots, z_n]$, and suppose we are given a division-free straight-line program $\mathcal{S}_F$ of length $L$ which evaluates $F$, an upper bound $D \geq \max_j \deg_{z_j}(F)$, and an upper bound $T$ on the number of nonzero terms $t$ of $F$. There exists a probabilistic algorithm which interpolates $F$ with probability at least $3/4$. The algorithm requires*

$$\widetilde{\mathcal{O}}\left(Ln(T \log D + n)(\log D + \log q) \log D + n^{\omega-1} T \log D + n^\omega \log D\right)$$

*bit operations.*[1,2]

This probability may be increased to $1 - \epsilon$ using standard techniques, with cost increased by a factor $\mathcal{O}(\log(\epsilon^{-1}))$.

The rest of this introductory section puts our work in context and defines the notation and problem definitions for the rest of the paper. The reader who is already familiar with the area may wish to glance at our list of notation in Appendix A, then skip to Section 2, where we give a high-level overview of the algorithm referred to by Theorem 1 and work out a small illustrative example in full detail. The end of Section 2 provides an outline for the remainder of the paper.

### 1.1. Background and related work

Polynomial interpolation is a fundamental problem of computational mathematics that dates back centuries to the classic work of Newton, Waring, and Lagrange. In such settings, given a list of $(n+1)$-dimensional points and some degree bounds, the coefficients of the unique $n$-variate polynomial interpolating those points is produced.

If the number of nonzero coefficients is relatively small, the unknown function can be treated as an exponential sum, and the task becomes that of finding the exponents and coefficients of only the nonzero terms. This is the *sparse interpolation* problem, and it differs crucially from other interpolation problems not only in the representation of the output, but also that of the input. Every efficient sparse interpolation algorithm of which we are aware requires some control over where the unknown function is sampled, and typically takes as input some procedure or black box that can evaluate the unknown sparse polynomial at any chosen point.

The sparse interpolation problem has received considerable interest over fields of characteristic zero. The classical Prony's method for exponential sums from 1795 (which can be regarded as the genesis of sparse interpolation) was later applied to sparse interpolation over the integers (Ben-Or and Tiwari, 1988; Kaltofen, 2010) and approximate complex numbers (Giesbrecht et al., 2009; Kaltofen et al., 2011). Compressive sensing is a different approach for approximate sparse interpolation which has the advantage of allowing the evaluation points to be chosen at random from a certain distribution (Candés et al., 2006; Donoho, 2006). Sparse Fourier and Hadamard–Walsh transforms allow for the interpolation of a sparse, complex-valued polynomial given by its discrete Fourier transform, and can find reasonable sparse approximations to non-sparse polynomials (Hassanieh et al., 2012; Kushilevitz and Mansour, 1993).

Straight-line programs are central to the study of algebraic complexity. One measure of the complexity of a rational function is the least size of a straight-line program that computes it. In 2003, in the celebrated paper (Kabanets and Impagliazzo, 2004) by Kabanets of Impagliazzo, it is shown that a deterministic polynomial-time algorithm for the identity testing of $F \in \mathbb{Z}[x]$ given by a straight-line

[1] For two functions $\phi, \psi$, we say $\phi \in \widetilde{\mathcal{O}}(\psi)$ if and only if $\phi \in \mathcal{O}(\psi \log^c \psi)$ for some constant $c \geq 0$.

[2] The constant $\omega < 2.38$ is the exponent of matrix multiplication, meaning that the product of two $n \times n$ matrices can be computed in $O(n^\omega)$ field operations.