Self-Testing/Correcting for Polynomials and for Approximate Functions

Peter Gemmell
Richard Lipton
Ronitt Rubinfeld
Madhu Sudan
Avi Wigderson

April 25, 2002

Abstract

The study of self-testing/correcting programs was introduced in [8] in order to allow one to use program $*P$* to compute function $*f$* without trusting that $*P$* works correctly. A self-tester for $*f$* estimates the fraction of $*x$* for which $*P*(*x*) = *f*(*x*)$; and a self-corrector for $*f$* takes a program that is correct on most inputs and turns it into a program that is correct on every input with high probability^1. Both access $*P$* only as a black-box and in some precise way are not allowed to compute the function $*f$*.

Self-correcting is usually easy when the function has the random self-reducibility property. One class of such functions that has this property is the class of multivariate polynomials over finite fields [4] [12]. We extend this result in two directions. First, we show that polynomials are random self-reducible over more general domains: specifically, over the rationals and over noncommutative rings. Second, we show that one can get self-correctors even when the program satisfies weaker conditions, i.e. when the program has more errors, or when the program behaves in a more adversarial manner by changing the function it computes between successive calls.

Self-testing is a much harder task. Previously it was known how to self-test for a few special examples of functions, such as the class of linear functions. We show that one can self-test the whole class of polynomial functions over $*Zp$* for prime $*p$* setting captures in particular the digital computation of real valued functions. We present a rigorous framework and obtain the first results in this area: namely that the class of linear functions, the log function and floating point exponentiation can be self-tested. All of the above functions also have self-correctors.