

Testing Programs Computing Numerical Function

Feodor Vainstein,
Professor of Computer Engineering
College of Engineering, GTREP & ECE
Georgia Institute of Technology
6001 Chatham Center Dr. Suite 350
Savannah, GA 31405
Phone: 912-790-3440
Fax: 912-651-7279
Email: fvainste@gtrep.gatech.edu

Computation of numerical functions is a very common sort of computation and therefore it is important to develop reliable software to implement it. M. Blum [2,3] and F. Vainstein [1] proposed to use functional equations to test the programs computing numerical functions.

The method proposed by M. Blum can be demonstrated by the following example. Suppose that we have to check the computation of the function $f(x) = \exp x$ at the point x . We randomly select a point y , compute $f(y)$ and $f(x+y)$ and then verify the computation using the formula $f(x+y) = f(x)f(y)$. Random selection of a point can be repeated several times thus increasing reliability of the testing.

The method proposed by F. Vainstein is similar but does not involve randomly selected points. In his paper [1] he introduced definition of polynomially checkable (PC) functions – the functions for which functional equations are polynomials, and proved that the class of PC functions is large and includes many commonly used functions.

In this paper we are going to introduce a new definition of polynomially

checkable functions so that the PC functions under the old definition still will be PC functions under the new definition. Moreover, with the new definition it is possible to use randomly selected points.

Definition.

A (complex) function $f : V \rightarrow \mathbf{C}$, ($V \subset \mathbf{C}$) is called polynomially checkable (PC) on U , where $U \subset V$ is some vicinity of 0 if there exists a polynomial $P \in \mathbf{C}[T_1, \dots, T_{k+2}]$, $P \neq 0$, and linear forms $l_1(x, y), \dots, l_k(x, y)$, $l_i(x, y) = a_i x + b_i y$, $l_i \neq l_j$ if $i \neq j$ (at least for one i $l_i \neq 0$), such that for every $x, y \in U$ if $l_i(x, y) \subset U$ ($i = 1, \dots, k$) then $P(x, y, f(l_1(x, y)), \dots, f(l_k(x, y))) = 0$. (1)

A real PC function can be defined similar way.

A polynomial P is called a checking polynomial of the function f and the equation (1) is called an addition theorem.

Denote by S the set of the three functions $x, \sin x, e^x$. Let $A \subseteq S$; denote by $\mathbf{R}(A)$

the field of all rational functions in $g_j \in A$ and by $\overline{\mathbf{R}(A)}$ its algebraic closure.

The following theorem holds.

Theorem

Let $f: \mathbf{R} \rightarrow \mathbf{R}$ belong to the field $\overline{\mathbf{R}(A)}$, $A \subseteq \{x, \text{Sin}x, e^x\}$. Then f is polynomially checkable with $k = |A|$.

Applications

Computation of a PC function at the point x can be verified the way similar to the example given above: we randomly select a point y and then substitute corresponding values in (1).

If the computation is correct then the addition theorem has to be satisfied. If we want to take into consideration limited accuracy of computations, equation (1) has to be substituted by

$$|P(x, y, f(l_1(x, y)), \dots, f(l_k(x, y)))| \leq \delta$$

where δ is a small positive number specified by precision of computation.

The theorem stated above shows that the class of functions which allow this approach is very broad and properly includes the set of all functions that can be obtained by application of a finite number of additions, subtractions, multiplications, divisions and raising to a rational power to the functions

$$\text{Const}, e^x, \text{Sin}(r_i x + a_i), \text{Cos}(q_j x + b_j),$$

where $r_i, q_j (i, j = 1, 2, \dots)$ are rational numbers.

This method can be used for detection and extended for correction [4] of errors in numerical computations caused by both software and/or hardware faults. This approach does not depend on the

form of representation or on the specific features of the implementation of a program or a device computing the given function and, being implemented in hardware, results in a substantial reduction of the hardware overhead required for multiple error detection and correction as compared to the check sum method or other methods previously known [5].

References.

1. F. S. Vainstein "Error Detection and Correction in Numerical Computations by Algebraic Methods", Lecture Notes in Computer Science 539. Springer-Verlag, 1991.
2. M. Blum and S. Kannan "Designing Programs That Check Their Work", 21st ACM Symposium on Theory of Computing, pp. 86-97, 1989.
3. M. Blum, M. Luby, and R. Rubinfeld "Self-Testing and Self-Correcting Programs with Applications to Numerical Problems", 22nd ACM Symposium on Theory of Computing, pp. 73-83, 1990
4. F. S. Vainstein "Low Redundancy Polynomial Checks for Numerical Computations", *Applicable Algebra in Engineering, Communication and Computing*, vol. 7, No. 6, 1996, pp. 439-447.
5. F. S. Vainstein "Self-Checking Design Procedure for Numerical Computations" *VLSI Design*, 1998, vol. 5, No. 4, pp. 385-392.