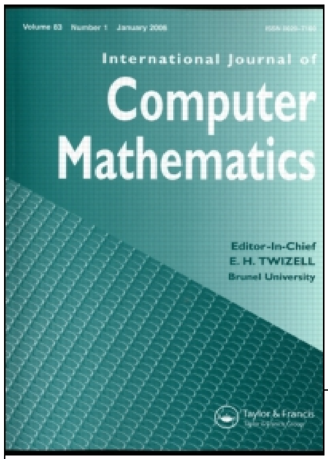

International Journal of Computer Mathematics

Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713455451

Calculating the square root with arbitrary order of convergence

Rezaul Alam Chowdhury [a]; M. Kaykobad [a]
[a] Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh

Online Publication Date: 01 January 2000

To cite this Article: Chowdhury, Rezaul Alam and Kaykobad, M. (2000) 'Calculating the square root with arbitrary order of convergence', International Journal of Computer Mathematics, 75:3, 297 — 302

To link to this article: DOI: 10.1080/00207160008804985
URL: http://dx.doi.org/10.1080/00207160008804985

Intern. J. Computer Math., Vol. 75, pp. 297–302
Reprints available directly from the publisher
Photocopying permitted by license only

CALCULATING THE SQUARE ROOT WITH ARBITRARY ORDER OF CONVERGENCE

REZAUL ALAM CHOWDHURY and M. KAYKOBAD*

Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka-1000, Bangladesh

(Received 27 April 1999)

In this paper an iterative algorithm has been presented for calculating the square root of a real number with arbitrary order of convergence using formulae derived by applying binomial theorem. The primary objective is to reduce the number of division operations required.

Keywords: Square root; division; convergence; Belaga parameters; polynomial; error

C. R. Category: G.1.0

1. INTRODUCTION

In real time digital signal processing, high performance modules for division and square root operations are essential for implementing many powerful algorithms. So VLSI array architectures have been proposed in [5] to implement division and square root operations. Schwarz and Flynn [8] have proposed a hardware starting approximation for the square root operation. Moreover, efficient algorithms for solving quadratic equations have been proposed by Morii and Yoshizu [6] and Ebrahimpour, Dorsy and Demko [1]. In general computation, the frequency of occurrence of division and square root operations are lower than that of multiplication and addition. So, many arithmetic processors do not provide any hardware support for them. That is why, these operations need to be emulated

*Corresponding author. e-mail: shaikat@bdonline.com by software, which is considerably slower than hardware implemented operations. Software routines for division operation can be up to an order of magnitude slower than multiplication operation. Again, since iterative methods for calculating square root generally rely on division operation at each iteration step, they become more time consuming. The amount of time consumed can be reduced by eliminating the need for a division operation at each step of iteration or by using a method of higher order of convergence and thus reducing the number of iterations required. In this paper, we take the second approach.

2. ALGORITHMS

In this section we discuss the well-known Newton-Raphson method for calculating square roots of a real number, and then present how it can be generalized to obtain an algorithm for solving the same problem with arbitrary order of convergence.

2.1. Newton-Raphson Method

There are many iterative methods for calculating the square root of a number [2,3,9]. The most commonly used one is the Newton-Raphson method, which is described as follows:

$$\boxed{y_{n+1} = \frac{1}{2} \times \left(y_n + \frac{x}{y_n} \right)}$$

where, y_n is the n-th approximation to the root. The Newton-Raphson method converges quadratically since the absolute error of the $n+1$-th iteration is described as follows:

$$\varepsilon_{n+1} = |y_{n+1} - \sqrt{x}|$$
$$\varepsilon_{n+1} = \frac{\varepsilon_n^2}{2y_n}$$
$$\text{If } 0.25 \le x \le 1.0$$
$$\text{then } 0.5 \le y_n \le 1$$
$$\text{and } \varepsilon_{n+1} < \varepsilon_n^2$$

This method uses 1 multiplication and 1 division per iteration. There are several other algorithms with shorter latency since they do not require a division operation at each step [7]. There are also algorithms of higher convergence [4].

In this paper we present a generalisation of Newton-Raphson's method.

2.2. New Algorithm

Let, y_n be the n-th approximation to $\sqrt{x}$ and m (≥ 2) be an integer. Now, expanding $(y_n - \sqrt{x})^m$ using binomial theorem we get,

$$\begin{aligned}(y_n - \sqrt{x})^m &= {}^mC_0 y_n^m - {}^mC_1 y_n^{m-1}\sqrt{x} + {}^mC_2 y_n^{m-2} x \\ &\quad - {}^mC_3 y_n^{m-3} x\sqrt{x} + \cdots + (-1)^m\, {}^mC_m (\sqrt{x})^m \\ &= \sum\nolimits_{0\leq i\leq \left\lfloor \frac{m-1}{2}\right\rfloor} {}^mC_{2i} y_n^{m-2i} x^i \\ &\quad - \sqrt{x}\sum\nolimits_{0\leq i\leq \left\lfloor \frac{m-1}{2}\right\rfloor} {}^mC_{2i+1} y_n^{m-2i-1} x^i\end{aligned}$$

Dividing both sides by $\sum_{0\leq i\leq \lfloor \frac{m-1}{2}\rfloor} {}^mC_{2i+1} y_n^{m-2i-1} x^i$,

$$\frac{(y_n - \sqrt{x})^m}{\sum_{0\leq i\leq \lfloor \frac{m-1}{2}\rfloor} {}^mC_{2i+1} y_n^{m-2i-1} x^i} = \frac{\sum_{0\leq i\leq \lfloor \frac{m-1}{2}\rfloor} {}^mC_{2i} y_n^{m-2i} x^i}{\sum_{0\leq i\leq \lfloor \frac{m-1}{2}\rfloor} {}^mC_{2i+1} y_n^{m-2i-1} x^i} - \sqrt{x}$$

Let, $p_n \sum_{0\leq i\leq \lfloor \frac{m-1}{2}\rfloor} {}^mC_{2i} y_n^{m-2i} x^i$, $q_n = \sum_{0\leq i\leq \lfloor \frac{m-1}{2}\rfloor} {}^mC_{2i+1} y_n^{m-2i-1} x^i$ and (p_n/q_n) = next approximation to $\sqrt{x} = y_{n+1}$

Then,

$$\begin{aligned} y_{n+1} - \sqrt{x} &= \frac{(y_n - \sqrt{x})^m}{q_n} \\ \Rightarrow |y_{n+1} - \sqrt{x}| &= \frac{|y_n - \sqrt{x}|^m}{|q_n|} \\ \Rightarrow \varepsilon_{n+1} &= \frac{\varepsilon_n^m}{|q_n|}, \quad \text{where, } \varepsilon_i = |y_i - \sqrt{x}| \\ &\qquad\qquad = \text{absolute error in } i\text{-th approximation}\end{aligned}$$

Hence, if our current approximation to $\sqrt{x}$ be y_n, then $\left(\sum_{0\leq i\leq \lfloor \frac{m-1}{2}\rfloor} {}^mC_{2i} y_n^{m-2i} x^i / \sum_{0\leq i\leq \lfloor \frac{m-1}{2}\rfloor} {}^mC_{2i+1} y_n^{m-2i-1} x^i\right)$ will be the next approximation with convergence of order m.

Now, for $m = 2$,

$$y_{n+1} = \frac{1}{2}\cdot\left(y_n + \frac{x}{y_n}\right),$$ which is the Newton-Raphson approximation.

For $m = 3$,

$$y_{n+1} = y_n \cdot \left(\frac{1}{3} + \frac{(8/9)x}{y_n^2 + (1/3)x} \right) = y_n \cdot \left(\frac{1}{3} + \frac{k_1}{y_n^2 + k_2} \right)$$

Here, k_1 and k_2 are constants for particular x.

Now, for example, suppose, the absolute error in our initial approximation is 0.1 and we wish to find the square root correct up to 240 decimal places. In this case, if we use Newton-Raphson method, we will require 8 iterations and hence 8 divisions and 8 multiplications. But with the approximation of order 3, we will be able to reach the goal in just 5 iterations using only 5 divisions and 10 multiplications.

In this method, we will require $n-1$ multiplications at each step of iteration. But the number of multiplication operations required for higher order approximations can be reduced to about $\lfloor \frac{n}{2} \rfloor + 3$, using Belaga parameters [4] in a slightly modified form.

For example, for $m = 16$, we will have the following iterative equation:

$$y_{n-1} = \frac{y_n^{16} + 120y_n^{14}x + 1820y_n^{12}x^2 + 8008y_n^{10}x^3 + 12870y_n^8x^4 + 8008y_n^6x^5 + 1820y_n^4x^6 + 120y_n^2x^7 + x^8}{16y_n^{15} + 560y_n^{13}x + 4368y_n^{11}x^2 + 11440y_n^9x^3 + 11440y_n^7x^4 + 4368y_n^5x^5 + 560y_n^3x^6 + 16y_nx^7}$$

Assuming $t_n = (x/y_n^2)$, we get,

$$y_{n+1} = y_n \times \frac{t_n^8 + 120t_n^7 + 1820t_n^6 + 8008t_n^5 + 12870t_n^4 + 8008t_n^3 + 1820t_n^2 + 120t_n + 1}{16t_n^7 + 560t_n^6 + 4368t_n^5 + 11440t_n^4 + 11440t_n^3 + 4368t_n^2 + 560t_n + 16}$$

Now, using Belaga parameters for evaluating the polynomials in the numerator and the denominator, we arrive at the following scheme:

Modified Belaga Parameters:

$a_1 = 29.75$
$a_2 = -223091.53125$
$a_3 = 218785.7399511727891442126 47064$
$a_4 = 48807899997.9794921875$
$a_5 = 475.0856311696440029325365 66871$
$a_6 = 613611483.2116461069832580 25383$
$a_7 = 251.0806676575668528548163 69101$
$a_8 = 6460773981.660262296829678 01438$

$$\begin{aligned}
b_1 &= 11.\dot{3}\\
b_2 &= -4651.\dot{7}0\dot{3}\\
b_3 &= 4310.196602998122599923495874988\\
b_4 &= 20047079.067215363511659807956 2\\
b_5 &= 71.507100705581103780207828729 0\\
b_6 &= 191807.095687792482600266125667
\end{aligned}$$

Pre-processing:

$$z = \frac{1}{y_0}$$

Iterative Steps:

$$\begin{aligned}
t &= x \times z \times z\\
t_2 &= t + t\\
\\
v_1 &= (t + a_1) \times t\\
v_2 &= (v_1 + t + a_2) \times (v_1 + a_3) + a_4\\
v_3 &= v_2 \times (v_1 + a_5) + a_6\\
p &= v_3 \times (v_1 + a_7) + a_8\\
\\
w_1 &= (t + b_1) \times t_2\\
w_2 &= (w_1 + t_2 + b_2) \times (w_1 + b_3) + b_4\\
w_3 &= w_2 \times (w_1 + b_5) + b_6\\
q &= t_2 \times w_3 + 16\\
\\
s &= \frac{q}{p}\\
z &= z \times s
\end{aligned}$$

Result:

$$\boxed{\sqrt{x} = \frac{1}{z}}$$

3. CONCLUSION

Here, we have described a simple but powerful technique to achieve arbitrary order of convergence in the iterative process of determining the square root. By increasing the order of convergence it will be possible to reduce arbitrarily the number of division operations required. This is important in the context that division operations are still costlier than multiplication operations. We have also shown how to reduce the number of multiplication operations for higher order of convergence.

References

[1] Ebrahimpour, M. R., Dorsey, J. F. and Demko, S. G. (1991). An Acceleration Technique for the Newton Solution of Quadratic Equations, *Computers Math. Applic.*, **22**(12), 79–87.

[2] Fike, C. T. (1968). *Computer Evaluation of Mathematical Functions*, Prentice-Hall, Inc., Englewood Cliffs, NJ.

[3] Hwang, K. (1979). *Computer Arithmetic: Principles, Architecture and Design*, John Wiley & Sons, New York.

[4] Kronsjö, L. I. (1978). *Algorithms: Their Complexity and Efficiency*, John Wiley & Sons, New York.

[5] McQuillan, S. E., McCanny, J. V. and Hamill, R. (1993). "New Algorithms and VLSI Architectures for SRT Division and Square Root", *Proceedings of the 11th IEEE Symposium on Computer Arithmetic*, pp. 80–86.

[6] Morii, M. and Yoshizu, S., "An Accelerated Solution of Quadratic Equations over GF(2^m)", *The Transactions of the IEICE*, **E-73**(11), November, 1990.

[7] Ramamoorthy, C. V., Goodman, J. R. and Kim, K. H., "Some properties of iterative square-rooting methods using high-speed multiplication", *IEEE Trans. Comput.*, **C-21**(8), 837–847, August, 1972.

[8] Schwarz, E. M. and Flynn, M. J. (1993). Hardware Starting Approximation for the Square Root Operation, *Proceedings of the 11th IEEE Symposium on Computer Arithmetic*, pp. 103–111.

[9] Waser, S. and Flynn, M. J. (1982). *Introduction to Arithmetic for Digital System Designers*, CBS College Publishing, New York.