立方根計算のハードウェアアルゴリズムについて

5H−8

高木 直史　田中 優
名古屋大学大学院工学研究科

1. Introduction

With advances of VLSI technologies, it becomes attractive to accelerate important complex operations by special hardware. It is also nice to avoid intermediate rounding errors between atomic FP-operations, but only have a bounded error for the final result of a complex operation. In this report, we propose a digit-recurrence algorithm for cube rooting which is used for solving algebraic equations of third or fourth degree.

We consider computation of the cube root of the mantissa part of a floating-point number. We compute $C = X^{\frac{1}{3}}$, where $2^{-3} \leq X < 1$. We assume X is represented as an n-digit r-ary fraction where $r = 2^b$. We intend to compute the result C in n-digit precision.

2. Algorithm

As digit-recurrence algorithms for division or square rooting [1], the cube root digit q_j is obtained step by step. Let $C[j]$ be the partial result after j iterations. Then, $C[j] = C[0] + \sum_{i=1}^{j} q_i r^{-i}$, where $C[0]$ is the initial value of the partial result. The recurrence equation on the partial result is

$$C[j+1] := C[j] + q_{j+1} r^{-j-1}. \tag{1}$$

We select the cube root digit q_{j+1} from a redundant digit set $\{-a, \cdots, -1, 0, 1, \cdots, a\}$, where $\frac{r}{2} \leq a < r$. The final result is $C = C[n] = \sum_{i=1}^{n} q_i r^{-i}$. The result has to be computed in n-digit precision. Namely,

$$-r^{-n} \leq X^{\frac{1}{3}} - C < r^{-n}. \tag{2}$$

We define a residual $W[j]$ as

$$W[j] = r^j (X - C[j]^3). \tag{3}$$

Subtracting r times (3) from the equation for $W[j+1]$, we get the recurrence equation on the residual as

$$W[j+1] := rW[j] - 3C[j]^2 q_{j+1} - 3C[j] q_{j+1}^2 r^{-j-1} - q_{j+1}^3 r^{-2j-2}. \tag{4}$$

Since this equation includes the term $-3C[j]^2 q_{j+1}$, we need squaring of j digit number $C[j]$ for the calculation. To avoid the squaring, we keep $C[j]^2$ and update it by addition/subtraction and shift.

Let $C[j]^2$ be $S[j]$. Then, the recurrence equation on $W[j]$ is rewritten as

$$W[j+1] := rW[j] - 3S[j] q_{j+1} - 3C[j] q_{j+1}^2 r^{-j-1} - q_{j+1}^3 r^{-2j-2}. \tag{5}$$

The recurrence equation on $S[j]$ is

$$S[j+1] := S[j] + q_{j+1} r^{-j-1} (2C[j] + q_{j+1} r^{-j-1}). \tag{6}$$

Since $C = C[j] + \sum_{i=j+1}^{n} q_i r^{-i}$ and the minimum and the maximum cube root digit values are $-a$ and a, respectively, from (2) and (3),

$$-3C[j]^2 \rho + 3C[j] \rho^2 r^{-j} - \rho^3 r^{-2j} \leq W[j] < 3C[j]^2 \rho + 3C[j] \rho^2 r^{-j} + \rho^3 r^{-2j} \tag{7}$$

must hold, where $\rho = a/(r-1)$ is the redundancy factor of the cube root digit set.

Since $2^{-3} \leq X < 1$ and $\frac{1}{2} < \rho \leq 1$, we can satisfy the bounds for $j = 0$ by letting $C[0] = 1$ and $W[0] = X - 1$. When $\rho = 1$, we can also satisfy the bounds by letting $C[0] = 0$ and $W[0] = X$.

The algorithm for cube rooting consists in performing n iterations of calculation of the recurrence equations (1), (5) and (6). In each iteration, we first produce the shifted residual $rW[j]$, and then select the cube root digit q_{j+1} by examining the shifted residual and the partial result $C[j]$. We will discuss on this letter. Finally, we perform the calculation of the recurrence equations.

The general algorithm is summarized as follows:

Algorithm [CBRT]

Step 1:
  $C[0] := 1$; $S[0] := 1$; $W[0] := X - 1$;
Step 2:
  for $j := 0$ to $n - 1$ do
  {
    Select q_{j+1} from $\{-a, \cdots, -1, 0, 1, \cdots, a\}$;
    $C[j+1] := C[j] + q_{j+1} r^{-j-1}$;
    $S[j+1] := S[j] + q_{j+1} r^{-j-1} (2C[j] + q_{j+1} r^{-j-1})$;
    $W[j+1] := rW[j] - 3S[j] q_{j+1} - 3C[j] q_{j+1}^2 r^{-j-1} - q_{j+1}^3 r^{-2j-2}$;
  }

□

On a Hardware Algorithm for Cube Rooting
Naofumi Takagi and Yu Tanaka
Nagoya University
Nagoya 464-8603, Japan
email: ntakagi@nuie.nagoya-u.ac.jp

When $\rho = 1$, we can replace Step 1 by $C[0] := 0$, $S[0] := 0$, and $W[0] := X$.

We can increase the speed of the implementation with a small increase in hardware complexity by performing the addition/subtractions in the recurrence equations without carry/borrow propagation by the use of a redundant representation. Therefore, in this report, we concentrate on this type of implementations. Namely, we represent the residual $W[j]$ and the square of the partial result $S[j]$ in a redundant representation, such as the carry-save form or the (binary) signed-digit representation, and perform the addition/subtractions without carry/borrow propagation. Since $-3 \le W[j] < 4$, we can represent $W[j]$ by either a two's complement carry-save form with 3-bit integer part (including the sign bit) or a binary signed-digit representation with 3-bit integer part.

Although we may represent the partial result $C[j]$ in a redundant representation as well, we keep the non-redundant representation of it by the on-the-fly conversion [2].

Now we consider selection of cube root digit q_{j+1}. We have to select q_{j+1} from $\{-a, \cdots, -1, 0, 1, \cdots, a\}$ so that the bounds for $W[j+1]$, i.e., $-3C[j+1]^2\rho + 3C[j+1]\rho^2 r^{-j-1} - \rho^3 r^{-2j-2} \le W[j+1] < 3C[j+1]^2\rho + 3C[j+1]\rho^2 r^{-j-1} + \rho^3 r^{-2j-2}$, are satisfied.

Let the interval of $W[j]$ where k can be selected as q_{j+1} be $[L_k[j], U_k[j])$. Then,

$$L_k[j] = 3C[j]^2(k-\rho) + 3C[j](k-\rho)^2 r^{-j-1} + (k-\rho)^3 r^{-2j-2}, \quad (8)$$

$$U_k[j] = 3C[j]^2(k+\rho) + 3C[j](k+\rho)^2 r^{-j-1} + (k+\rho)^3 r^{-2j-2}. \quad (9)$$

Note that the lower bound of the interval for $k = -a$ and the upper bound of the interval for $k = a$ are equal to the lower bound and the upper bound of $W[j]$, respectively.

The continuity condition $U_{k-1}[j] \ge L_k[j]$ yields

$$(2\rho - 1)(3C[j]^2 + 3C[j](2k-1)r^{-j-1} + (3k^2 - 3k + \rho^2 - \rho + 1)r^{-2j-2}) \ge 0. \quad (10)$$

The left hand side of (10) indicates the overlap between consecutive selection intervals, which is used to simplify the selection function.

q_{j+1} depends on $rW[j]$ and $C[j]$. Using the overlap, we can select q_{j+1} by estimates of them. Let the digit selection function be $Select(r\hat{W}[j], \hat{C}[j])$ where $r\hat{W}[j]$ and $\hat{C}[j]$ are estimates of $rW[j]$ and $C[j]$, respectively. Then the function is described by a set of selection constants, $\{m_k(\hat{C}[j]) | k \in \{-a+1, \cdots, -1, 0, 1, \cdots, a\}\}$, where $q_{j+1} = k$ if $m_k(\hat{C}[j]) \le r\hat{W}[j] < m_{k+1}(\hat{C}[j])$.

We obtain $r\hat{W}[j]$ by truncating $rW[j]$, which is in a redundant representation, to t fractional bits. We obtain $\hat{C}[j]$ by truncating $C[j]$ to d fractional bits. (Note that not r-ary digits but bits.) Since $C[j]$ is in the non-redundant representation, $\hat{C}[j] = C[j]$ for small j's such that $r^{-j} \ge 2^{-d}$ and $\hat{C}[j] \le C[j] \le \hat{C}[j] + 2^{-d} - r^{-j}$ for the other j's.

When $W[j]$ is in the carry-save form, $r\hat{W}[j] \le rW[j] < r\hat{W}[j] + 2^{-t+1}$. Therefore,

$$m_k(\hat{C}[j]) \ge \max_{\hat{C}[j]}(L_k[j]), \quad (11)$$

$$(m_k(\hat{C}[j]) - 2^{-t}) + 2^{-t+1} \le \min_{\hat{C}[j]}(U_{k-1}[j]) \quad (12)$$

must be satisfied. Namely, $m_k(\hat{C}[j])$ must be a multiple of 2^{-t} that satisfies (11) and (12). Here, $\max_{\hat{C}[j]}(L_k[j])$ denotes the maximum value of the lower bound of the interval of $rW[j]$ where k can be selected as q_{j+1} when the estimate of $C[j]$ is $\hat{C}[j]$. $\min_{\hat{C}[j]}(U_{k-1}[j])$ denotes the minimum value of the upper bound of the interval of $rW[j]$ where $k-1$ can be selected as q_{j+1} when the estimate of $C[j]$ is $\hat{C}[j]$. Note that the maximum value of $r\hat{W}[j]$ for which $k-1$ is selected as q_{j+1} is $m_k(\hat{C}[j]) - 2^{-t}$.

Since $\max_{\hat{C}[j]}(L_k[j])$ and $\min_{\hat{C}[j]}(U_{k-1}[j])$ depend on j, a different selection function might result for different j. When $r = 2$, $a = 1$ and $\rho = 1$, a single selection function for all j exists. When $r \ge 4$, no single selection function for all j exists, and therefore, we should find J so that a single selection function can be used for $j \ge J$ and consider the cases for $j < J$ separately.

When $r = 2$, $a = 1$ ($\rho = 1$), and $W[j]$ is in the carry-save form, by letting $C[0] = 0$ and $W[0] = X$, we obtain a single selection function $\{m_0 = -2^{-1}, m_1 = 0\}$ with $t = 1$, which is independent of $C[j]$.

3. Conclusion

We have proposed a digit-recurrence algorithm for cube rooting. We have shown a general algorithm. Different specific versions of the algorithm are possible, depending on the radix, the redundancy factor of the digit set of the cube root, the type of representation of the residual and the square of the partial result (carry-save or signed-digit), and the digit selection function. Implementation of any version of the algorithm can be sequential, or combinational, or a combination of both. Pipelining can also be used. Any implementation has a regular structure suitable for VLSI.

References

[1] M. D. Ercegovac and T. Lang. *Division and Square Root - Digit-Recurrence Algorithms and Implementations*, Kluwer Academic Publishers, 1994.

[2] M. D. Ercegovac and T. Lang. On-the-fly conversion of redundant into conventional representations, *IEEE Trans. Comput.*, C-36(7): 895-897, July 1987.