[7] W.J. Dally and C.L. Sietz, "Deadlock-free message routing in multiprocessor interconnection networks," *IEEE Trans. Computers*, vol. 36, pp. 547-553, 1987.
[8] L.M. Ni and P.K. McKinley, "A survey of wormhole routing techniques in direct networks," *Computer*, vol. 26, no. 2, pp. 62-76, 1993.
[9] J.M. Rosario, "High performance parallel I/O on the nCUBE 2," *Trans. Inst. of Electronics, Information, and Comm. Engineers*, vol. J75-D-I, no. 8, pp. 626-636, 1992.
[10] T. Shirakawa, T. Kageyama, H. Abe, and T. Hoshino, "Processor array PAX-128," *Trans. Inst. of Electronics and Comm. Engineers*, vol. J67-D, no. 8, pp. 853-860, 1984.
[11] H. Ishihata, T. Horie, S. Inano, T. Shimizu, and S. Kato, "An architecture of highly parallel computer AP1000," *Proc. IEEE Pacific Rim Conf. Communications, Computers, and Signal Processing*, pp. 13-16, 1991.
[12] L.N. Bhuyan and D.P. Agrawal, "Generalized hypercube and hyperbus structures for a computer network," *IEEE Trans. Computers*, vol. 33, pp. 323-333, 1984.
[13] P.W. Dowd and K. Jabbour, "Spanning multiaccess channel hypercube computer interconnection," *IEEE Trans. Computers*, vol. 37, pp. 1,137-1,142, 1988.
[14] C.M. Fiduccia, "Bused hypercubes and other pin-optimal networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 3, pp. 14-24, 1992.

Error Analysis of Approximate Chinese-Remainder-Theorem Decoding

Ching Yu Hung and Behrooz Parhami

***Abstract*—Approximate Chinese-remainder-theorem decoding of residue numbers is a useful operation in residue arithmetic. The decoding yields an approximation to (X mod M)/M, in the range [0, 1), where X is the input number and M is the product of all moduli. We show the error distribution and worst-case errors for both the truncation and rounding versions of the approximate decoding procedure. We also prove that, contrary to some published accounts, limiting the dynamic range is ineffective in reducing the maximal error.**

***Index Terms*—Computation errors, computer arithmetic, residue numbers, RNS representation, scaled decoding.**

I. INTRODUCTION

Residue number system (RNS) offers fast, carry-free, addition and multiplication, as well as useful properties for error detection and correction in arithmetic operations. Magnitude information, however, is not explicit in the representation. Decoding refers to the process of obtaining some magnitude information from a residue number. This paper deals exclusively with scaled decoding, where the decoding output is scaled to the range [0, 1), or equivalently, [0, 2^k) for some integer k. By performing summation modulo a power of 2, as opposed to modulo M, the product of all moduli, scaled decoding can be carried out more efficiently.

Approximate CRT (Chinese-remainder-theorem) decoding is an essential operation in many RNS arithmetic algorithms, including sign detection, division, and overflow handling. This operation is used in several proposals for RNS arithmetic and residue-to-binary conversion [2], [3], [4], [5], [7]. The precision of decoding ranges from low, for approximate sign detection [3], to medium, for scaled decoding [4], [5], to full representation, for exact sign detection [7] and scaled conversion to positional notation [2]. However, we have been unable to locate any work that analyzes the decoding error formally. Some obtain a simple upper-bound error by treating the problem as rounding or truncation error, while others use exhaustive search to find the range of errors for specific moduli.

Soderstrand et al. [5] show the maximal error for one set of moduli. They also suggest that the error can be reduced by excluding a fraction of the dynamic range. Unfortunately, their results, based on exhaustive computer search, are incorrect. Thus, the question of whether restricting the dynamic range can effectively reduce the error remains open. Vu [7] proposes a scaled CRT decoding in the range [0, 2) for (exact) sign detection. He concludes that the minimal width of tables and adders used in the decoding is roughly $\lceil \log_2 nM \rceil$ bits, where n is the number of moduli. The method used is exact rather than approximate decoding. Griffin et al. [2] use exhaustive computer search and plot the error distribution for a modulus set of the form $\{2^k-1, 2^k, 2^k+1\}$. Kim et al. [4] compute the maximal error for a few sets of moduli through computer search. The rounding error compensation method proposed in the paper requires extra lookup tables and an adder tree to reduce the error. It appears that extending the precision of the lookup table would achieve the same effect with simpler hardware.

Manuscript received Dec. 22, 1993; revised Apr. 28, 1994.
C.Y. Hung is with the Integrated Systems Laboratory, Texas Instruments, Dallas, TX 75243.
B. Parhami is with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106-9560; e-mail: parhami@ece.ucsb.edu.
To order reprints of this article, e-mail: transactions@computer.org, and reference IEEECS Log Number C95121.

In this paper, we analyze the error distribution of approximate CRT decoding, for both *truncation* and *rounding* versions. When n, the number of moduli, is not very small, the distribution is shown to resemble a normal distribution. The worst-case error is derived and is shown to be close to the simple upper bound. Moreover, it is proven that input numbers that cause near-worst-case errors are so evenly distributed throughout the dynamic range that no subrange larger than M/m_i can avoid all such numbers. It is therefore infeasible to reduce the maximal error by restricting the dynamic range, as suggested by Soderstrand et al. [5].

II. Approximate CRT Decoding

We denote the list of moduli for the RNS as $\{m_1, m_2, \ldots, m_n\}$. An unsigned number X is represented by a list of residues with respect to the n moduli. $X = (x_1, x_2, \ldots, x_n)$, $x_i = |X|_{m_i}$. Furthermore, let $M = \prod_{1\le i\le n} m_i$, $\hat{m}_i = M/m_i$, and $q_i = \left|\hat{m}_i^{-1}\right|_{m_i}$.

We define the exact scaled CRT decoding result as $f(X)$. It is computed as follows.

$$f(X) = \left|\sum_{1\le i\le n} b_i(x_i)\right|_1 = X/M \tag{1}$$

$$b_i(j) = \frac{|jq_i|_{m_i}}{m_i}. \tag{2}$$

The truncation and rounding versions of approximate CRT decoding are denoted as $f^{T,d}(X)$ and $f^{R,d}(X)$, where d is the number of bits kept after truncation or rounding.

$$f^{T,d}(X) = \left|\sum_{1\le i\le n} b_i^{T,d}(x_i)\right|_1, \tag{3}$$

$$b_i^{T,d}(j) = \text{Truncate}\left(\frac{|jq_i|_{m_i}}{m_i}, d\right), \tag{4}$$

$$f^{R,d}(X) = \left|\sum_{1\le i\le n} b_i^{R,d}(x_i)\right|_1, \tag{5}$$

$$b_i^{R,d}(j) = \text{Round}\left(\frac{|jq_i|_{m_i}}{m_i}, d\right). \tag{6}$$

Each component, $b_i^{T,d}(x_i)$ or $b_i^{R,d}(x_i)$, can be computed on-line or precomputed and stored in a lookup table. The parameter d dictates the cost of computing and storing these components. The Truncate and Round functions are defined as Truncate$(z, d) = 2^{-d}\lfloor 2^d z\rfloor$ and Round$(z, d) = 2^{-d}\left\lfloor 2^d z + \frac{1}{2}\right\rfloor$.

We are interested in the decoding errors $\Delta f^{T,d}(X) \equiv f^{T,d}(X) - f(X)$ and $\Delta f^{R,d}(X) \equiv f^{R,d}(X) - f(X)$. Viewing the decoding error as the summation of n truncation or rounding errors, each with a value of *ulp* or $\frac{1}{2}$*ulp* (*ulp* = unit in last position = 2^{-d}) depending on whether truncation or rounding is used, we conclude that

$$-n2^{-d} < \Delta f^{T,d}(X) \le 0, \tag{7}$$

$$-n2^{-d-1} < \Delta f^{R,d}(X) \le n2^{-d-1}. \tag{8}$$

Vu [7] derives the minimal d that allows accurate sign detection of residue numbers, and in the process implies (8) without explicitly stating it. We call the bounds in (7) and (8) the simple error bounds for (scaled) CRT decoding. Figs. 1 and 2 are scatter plots for the decoding error of the truncation and rounding versions for all input numbers with the modulus set {5, 7, 9, 11}. The simple error bounds clearly hold for this example.

Note that (7) and (8) may not hold when $f(X)$ is close to the boundaries 0 and 1. In such cases, due to the modulo-1 operations in (3) and (5), errors may cause $f^{T,d}$ or $f^{R,d}$ to cross the boundary, leading to a large value for $\Delta f^{T,d}$ or $\Delta f^{R,d}$. Therefore, our analyses, as presented in this paper, are valid only if a small upper portion of the dynamic range near the boundary is excluded and appropriate compensation is applied to the decoded values that fall in the excluded interval. However, if one defines the total error as the sum of individual error terms, rather than as the difference between $f(X)$ and its approximate version, then our results and conclusions will apply to these boundary cases as well.

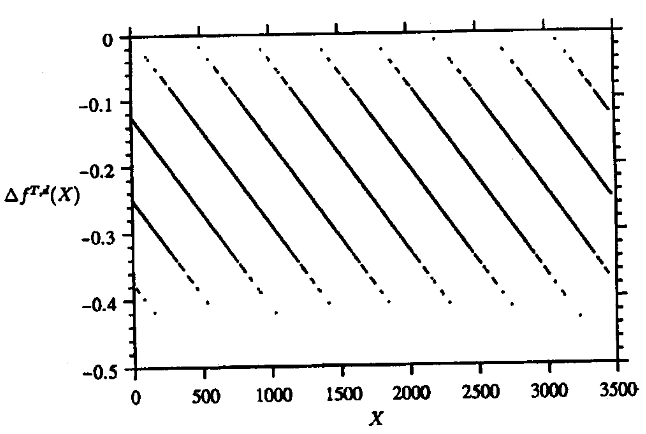

Fig. 1. Error of truncated CRT decoding as a function of the input X. Modulus set is {5, 7, 9, 11}, $d = 3$, so the decoding error is bounded in (−0.5, 0].

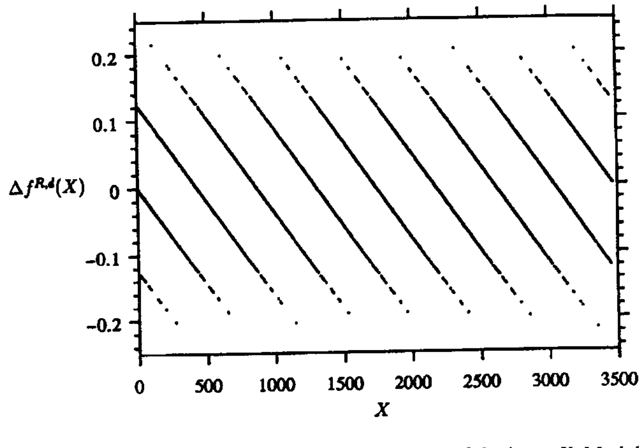

Fig. 2. Error of rounded CRT decoding as a function of the input X. Modulus set is {5, 7, 9, 11}, $d = 3$, so the decoding error is bounded in (−0.25, 0.25].

III. Computing the Error Sets

We are interested in the decoding errors $\Delta f^{T,d}(X)$ and $\Delta f^{R,d}(X)$. First, we derive the error for individual terms $\Delta b_i^{T,d}(j) \equiv b_i^{T,d}(j) - b_i(j)$ and $\Delta b_i^{R,d}(j) \equiv b_i^{R,d}(j) - b_i(j)$. From (4), we have

$$b_i^{T,d}(j) = 2^{-d}\left\lfloor \frac{2^d |jq_i|_{m_i}}{m_i} \right\rfloor = \frac{|jq_i|_{m_i}}{m_i} - \frac{2^{-d}\left|2^d jq_i\right|_{m_i}}{m_i}.$$

It follows that

$$\Delta b_i^{T,d}(j) = -2^{-d}\frac{\left|2^d jq_i\right|_{m_i}}{m_i}. \tag{9}$$

For rounded CRT decoding, we derive from (6) that

$$b_i^{R,d}(j) = 2^{-d}\left\lfloor \frac{2^d\left|jq_i\right|_{m_i} + \frac{m_i}{2}}{m_i}\right\rfloor = \frac{\left|jq_i\right|_{m_i}}{m_i} + 2^{-d-1} - 2^{-d}\frac{\left|2^d jq_i + \frac{m_i}{2}\right|_{m_i}}{m_i},$$

$$\Delta b_i^{R,d}(j) = 2^{-d}\left(\frac{1}{2} - \frac{\left|2^d jq_i + \frac{m_i}{2}\right|_{m_i}}{m_i}\right). \tag{10}$$

We are interested in the error variations for all X in the dynamic range. For example, we ask how the error is distributed, what the worst-case values are, and if it is possible to reduce the error by limiting the range of represented numbers, as suggested in [5]. To gain such information we look at the errors as a set generated by all possible inputs in the dynamic range.

The set of moduli are assumed to be pairwise relatively prime. Furthermore, we restrict our argument to the case that all moduli are odd. The other case, when there is one even modulus, is slightly more complicated and is discussed in the appendix. We shall use the following lemma from number theory [1]:

Let a and m be integers. If a is relatively prime to m, we have

$$\{|ja|_m \mid 0 \le j \le m-1\} = \{0, 1, 2, \ldots, m-1\}.$$

Since both factors, 2^d and q_i, inside the modulo operation in (9) and (10) are relatively prime to m_i, $a = 2^d q_i$ is also relatively prime to m_i. By the above lemma, we obtain all possible error values for each component as

$$\left\{\Delta b_i^{T,d}(j) \middle| 0 \le j \le m_i - 1\right\} = \left\{-2^{-d}\frac{j}{m_i} \middle| 0 \le j \le m_i - 1\right\}, \tag{11}$$

$$\begin{aligned}\left\{\Delta b_i^{R,d}(j) \middle| 0 \le j \le m_i - 1\right\} &= \left\{2^{-d}\frac{\frac{m_i}{2} - \left(j + \frac{1}{2}\right)}{m_i} \middle| 0 \le j \le m_i - 1\right\} \\ &= \left\{2^{-d}\frac{j}{m_i} \middle| -\frac{m_i - 1}{2} \le j \le \frac{m_i - 1}{2}\right\}.\end{aligned} \tag{12}$$

Thus, as the input j to the error component $\Delta b_i^{T,d}(j)$ varies from 0 to $m_i - 1$, the error takes values from the set $\{0, -2^{-d}/m_i, \ldots, -2^{-d}(m_i - 1)/m_i\}$, not necessarily in that order. Similarly for the rounding version, as j varies from 0 to $m_i - 1$, $\Delta b_i^{R,d}(j)$ takes values from the set on the right-hand side of (12). Note that the truncation error set is nonpositive, and the rounding error set is symmetric about zero.

Next we look at the decoding errors $\Delta f^{T,d}(X)$ and $\Delta f^{R,d}(X)$, again as sets for all input numbers. Error accumulates as $b^{T,d}(x_i)$ or $b^{R,d}(x_i)$ terms are summed modulo 1. Except when X is close to the two extremes of the dynamic range, we have

$$\Delta f^{T,d}(X) = \sum_{1 \le i \le n} \Delta b_i^{T,d}(x_i),$$

$$\Delta f^{R,d}(X) = \sum_{1 \le i \le n} \Delta b_i^{R,d}(x_i).$$

As the input X sweeps the entire dynamic range, $0 \le X \le M - 1$, the n error components have each of their inputs x_i stepping through all combinations of $0 \le x_i \le m_i - 1$. The decoding errors of all inputs in the dynamic range, as a set, thus contain all possible combinations ($\prod m_i = M$ of them) of the n error components:

$$\begin{aligned}&\left\{\Delta f^{T,d}(X) \middle| 0 \le X \le M - 1\right\} \\ &= \left\{-2^{-d}\sum_{1 \le i \le n}\frac{y_i}{m_i} \middle| 0 \le y_i \le m_i - 1, 1 \le i \le n\right\},\end{aligned} \tag{13}$$

$$\begin{aligned}&\left\{\Delta f^{R,d}(X) \middle| 0 \le X \le M - 1\right\} \\ &= \left\{2^{-d}\sum_{1 \le i \le n}\frac{y_i}{m_i} \middle| -\frac{m_i - 1}{2} \le y_i \le \frac{m_i - 1}{2}, 1 \le i \le n\right\}.\end{aligned} \tag{14}$$

In the above equations, n variables y_1 through y_n are used in the n-term summation, and each variable y_i can assume one of m_i values. Each one of the M summations generates a unique sum. The uniqueness can easily be proven for the truncation version: We multiply all set elements by $-2^d M$ to obtain $\{\sum y_i \hat{m}_i \mid 0 \le y_i \le m_i - 1, 1 \le i \le n\}$. Let two sums be equal; i.e., $\sum y_i \hat{m}_i = \sum z_i \hat{m}_i$. Taking modulo m_j, $1 \le j \le n$, on both sides, we have $\left|y_j \hat{m}_j\right|_{m_j} = \left|z_j \hat{m}_j\right|_{m_j}$, and therefore we must have $y_j = z_j$, for $1 \le j \le n$. The proof for the rounding version is similar.

Equations (13) and (14) contain the exact quantity of every possible decoding error. However, they do not provide explicit information on how these errors are distributed. To get an intuitive feeling for the distribution of error magnitudes, we resort to an approximate analysis. Each error component is approximated by a uniformly distributed random variable, roughly in $(-2^{-d}, 0]$ and $(-2^{-d-1}, 2^{-d-1})$, respectively for truncation and rounding. The decoding error is the sum of the n identical and independent uniformly distributed random variables. Therefore the distribution function of decoding error is just the convolution of the n uniform distributions. As n assumes the values 1, 2, ..., the shape of our approximate distribution changes from uniform to triangular (a rising linear part followed by a declining segment), ..., resembling normal distribution for moderate values of n, and eventually converging to a perfect normal distribution.

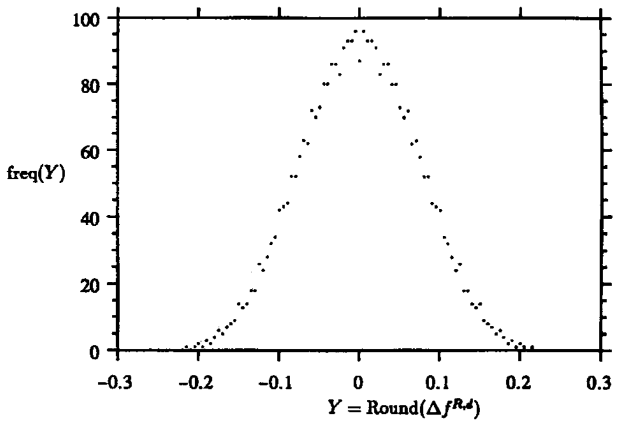

Fig. 3. Error distribution of rounded CRT decoding. Modulus set is {5, 7, 9, 11}, $d = 3$, interval size is 0.005, so the plot appears as a jagged normal-like distribution.

The probability-distribution view of the decoding error (rounding version) shown in Fig. 3 has been obtained by dividing the real line into equal-size intervals and counting the number of errors falling into each interval as all integers in the dynamic range are decoded with the procedure. The result is the same as randomly selecting a number X in the dynamic range and finding the probability of the decoding error falling into each interval. The size of intervals affects the shape of the distribution. When interval size is small, viz $< 2^{-d}/m_i$, the analogy is not so good because error components do not fall into every interval. The truncation version has a similar shape in the error distribution, but the peak is shifted from 0 to $n2^{-d-1}$.

A possible application on the above insight on the distribution of errors is the modeling of decoding errors. When CRT decoding is performed on a time-domain signal, received as a residue number, the decoding errors may be viewed collectively as a single noise source added to the signal, which may also be affected by other noise sources prior to and after the decoding. When the number of moduli, n, is not very small, the decoding error can be approximated by a normally distributed random variable, also known as *white noise* in signal processing.

IV. Analysis of Errors

We are now ready to answer the questions we have posed. The worst-case errors are easy to derive from (13) and (14). For the truncation version, there is an input $X^{(\text{worst})}$ such that

$$\Delta f^{T,d}\left(X^{(\text{worst})}\right) = -2^{-d} \sum_{1\le i\le n} \frac{m_i - 1}{m_i}. \tag{15}$$

This is very close to the simple bound $-2^{-d}n$. For the rounding version, there are $X^{(\text{worstA})}$ and $X^{(\text{worstB})}$ for which the errors are close to the simple bounds $\pm 2^{-d-1}n$:

$$\Delta f^{R,d}\left(X^{(\text{worstA})}\right) = 2^{-d} \sum_{1\le i\le n} \frac{-m_i + 1}{2m_i}, \tag{16}$$

$$\Delta f^{R,d}\left(X^{(\text{worstB})}\right) = 2^{-d} \sum_{1\le i\le n} \frac{m_i - 1}{2m_i}. \tag{17}$$

The next question is whether it is possible to reduce the decoding error by restricting the dynamic range. The following argument for the truncation version shows that the answer is negative. The argument for the rounding version is similar.

Let $X^{(\text{worst})} = (x_1, x_2, \ldots, x_n)$ be the number that causes the maximum error, $\Delta b^{T,d}(x_i) = -2^{-d}(m_i - 1)/m_i$. Consider m_1 distinct numbers $Y^{(j)} = (j, x_2, x_3, \ldots, x_n)$, $0 \le j \le m_1 - 1$, which include $X^{(\text{worst})}$. Since $n - 1$ out of the n residues of these numbers are identical, these m_1 numbers differ by multiples of M/m_1. They must therefore spread out evenly throughout the dynamic range $[0, M - 1]$. We proceed to evaluate their decoding errors. Since these numbers are different only in the first residue, and each error component only depends on its corresponding residue, $n - 1$ of the error components are the same for all the numbers. Therefore, their decoding error must be at least

$$Y^{(j)} \ge -2^{-d} \sum_{2\le i\le n} \frac{m_i - 1}{m_i} \approx -2^{-d}(n-1),$$

only a fraction, $1/n$, less than the peak error. We conclude that no continuous range larger than M/m_1 can avoid these near-peak decoding errors. Since the choice of using the first modulus is arbitrary, no range larger than M/m, m being the largest modulus, can avoid near-peak decoding errors. It is therefore not possible to significantly reduce the decoding error without substantial sacrifice in the dynamic range. The above argument is confirmed by Figs. 1 and 2: For both versions of CRT decoding, close-to-peak errors are observed throughout the dynamic range.

Another observation from Figs. 1 and 2 is that the errors line up on parallel sloped lines. These lines overlap vertically so that for input numbers in a small interval, the decoding errors spread out into several clusters that are 2^{-d} apart. The regularity in the plots appears to convey some information about the decoding error. Actually, no useful information can be gained. Take the rounding version for example. The decoding produces multiples of 2^{-d} as output. If the decoding error were within 2^{-d-1} in all cases, then we would have perfect rounding and the plot would have been saw-tooth like, without the sloped lines vertically overlapping one anther. However, the n components of $b^{R,d}(x_i)$, each rounded up or down, often add up to a decoding result that is a few multiples of 2^{-d} above or below the perfect result. This explains the overlapping of sloped lines that are separated by a vertical distance of 2^{-d}.

V. Conclusions

We have obtained closed-form solutions for the error in approximate CRT decoding of residue numbers. Error distribution and worst-case errors are shown. Our results on the worst-case errors confirm that the simple upper bound first observed by Vu [7] is close to the actual worst-case errors. Since RNS representations are nonredundant, it would have been surprising if worst-case error components did not combine.

Possible future research directions include: seeking to control the decoding error based on the knowledge of error distribution; developing a more efficient sign detection algorithm by incrementally evaluating the decoding error, thereby improving RNS division algorithms; and tailoring the analyses to special sets of moduli.

Appendix
The Case of RNS with One Even Modulus

Because all the moduli are pairwise relatively prime, there is at most one even modulus. Without loss of generality, let this modulus be m_1. From the analysis in Section IV we see that the decoding error is the sum of n independent components, each dependent only on the corresponding residue. An easy way out is to rewrite the lower bounds of error sets and of worst-case error by eliminating the first component from (13), (14), (15), (16), and (17). We can do slightly better than that.

We need a generalized version of the lemma used in Section III [1].

Let a and m be integers, $g = \gcd(a, m)$ and $k = m/g$. We have

$$\{|ja|_m \mid 0 \le j \le m - 1\} = \{0, g, 2g, \ldots, (k-1)g\},$$

Let $m_1' = m_1/\gcd(2^d, m_1)$. It follows from the lemma that, for the even modulus, (11) becomes

$$\Delta b_1^{T,d} = \left\{-2^{-d}\frac{j}{m_1'} \,\middle|\, 0 \le j \le m_1' - 1\right\}. \tag{18}$$

The rounding version is more difficult. Let $m_1 = r2^s$ with r odd. There are two cases:

Case I. $s \le d$. In this case, $m_1' = m_1/\gcd(2^d, m_1) = m_1/2^s = r$, an odd number. The modulo m_i expression in (10) becomes

$$\left|2^d jq_i + r2^{s-1}\right|_{r2^s} = 2^s\left|2^{d-s} jq_i + \frac{r}{2}\right|_r$$
$$= 2^s\left(j' + \frac{1}{2}\right), \text{ for some } j',\ 0 \le j' \le r - 1.$$

Thus, (12) becomes

$$\begin{aligned}\Delta b_1^{R,d} &= \left\{ 2^{-d}\left(\frac{1}{2}-\frac{2^s\left(j+\frac{1}{2}\right)}{r2^s}\right) \middle| 0 \le j \le r-1 \right\} \\ &= \left\{ 2^{-d}\frac{\frac{r}{2}-\left(j+\frac{1}{2}\right)}{r} \middle| 0 \le j \le r-1 \right\} \\ &= \left\{ 2^{-d}\frac{j}{r} \middle| -\frac{r-1}{2} \le j \le \frac{r-1}{2} \right\} \\ &= \left\{ 2^{-d}\frac{j}{m_1'} \middle| -\frac{m_1'-1}{2} \le j \le \frac{m_1'-1}{2} \right\}.\end{aligned} \tag{19}$$

CASE II. $s > d$. In this case, $m_1' = m_1/\gcd(2^d, m_1) = m_1/2^d = r2^{s-d}$, an even number. The modulo m_i expression in (10) becomes

$$\begin{aligned}\left|2^d jq_i + r2^{s-1}\right|_{r2^s} &= 2^d \left|jq_i + r2^{s-d-1}\right|_{r2^{s-d}} \\ &= 2^d j', \text{ for some } j', 0 \le j' \le m_1' - 1.\end{aligned}$$

Equation (12) now becomes

$$\begin{aligned}\Delta b_1^{R,d} &= \left\{ 2^{-d}\left(\frac{1}{2}-\frac{2^d j}{r2^s}\right) \middle| 0 \le j \le m_1'-1 \right\} \\ &= \left\{ 2^{-d}\left(\frac{1}{2}-\frac{j}{m_1'}\right) \middle| 0 \le j \le m_1'-1 \right\} \\ &= \left\{ 2^{-d}\frac{j}{m_1'} \middle| -\frac{m_1'}{2}+1 \le j \le \frac{m_1'}{2} \right\}.\end{aligned} \tag{20}$$

Comparing (19) and (20) with the original (12), we see the effect of the even modulus is just replacing m_1 with m_1' and changing boundary conditions for the enumerator j. Instead of having m_1 levels of error values, now there are only $m_1' = m_1/\gcd(2^d, m_1)$ levels. The argument on error distribution, worst-case errors, and uniform spread of near-peak errors can be modified for the case of an even modulus. In the special case that $m_1 = 2^d$, we have $m_1' = 1$, and the even modulus contributes nothing to the decoding error. Because the first error component skips levels, the error distribution should become somewhat more jagged when n is small. The argument on uniform spread of near-peak errors is still valid, if we select some odd modulus for constructing $Y^{(i)}$.

REFERENCES

[1] R.L. Graham, D.E. Knuth, and O. Patashnik, *Concrete Mathematics.* Reading, Mass.: Addison-Wesley, 1989.

[2] M. Griffin, F. Taylor, and M. Sousa, "New scaling algorithms for the Chinese remainder theorem," *Proc. 22nd Asilomar Conf. Signal, Systems, and Computers*, vol. 1, pp. 375–378, 1988.

[3] C.Y. Hung and B. Parhami, "An approximate sign detection method for residue numbers and its application to RNS division," *Computers and Mathematics with Applications*, vol. 27, pp. 23–35, Feb. 1994.

[4] J.Y. Kim, K.H. Park, and H.S. Lee, "Efficient residue-to-binary conversion technique with rounding error compensation," *IEEE Trans. Circuits and Systems*, vol. 38, pp. 315–317, Mar. 1991.

[5] M.A. Soderstrand, C. Vernia, and J.-H. Chang, "An improved residue number system digital-to-analog converter," *IEEE Trans. Circuits and Systems*, vol. 30, pp. 903–907, Dec. 1983. Also in [6], pp. 47-50.

[6] *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*, M.A. Soderstrand, W.K. Jenkins, G.A. Jullien, and F.J. Taylor, eds. IEEE Press, 1986.

[7] T.V. Vu, "Efficient implementations of the Chinese remainder theorem for sign detection and residue decoding," *IEEE Trans. Computers*, vol. 34, pp. 646–651, July 1985.

A Continued-Fraction Analysis of Trigonometric Argument Reduction

Roger Alan Smith

Abstract—The calculation of a trigonometric function of a large argument x is effectively carried out by finding the integer N and $0 \le \alpha < 1$ such that $x = (N+\alpha)\frac{\pi}{4}$. This reduction modulo $\frac{\pi}{4}$ makes it possible to calculate a trigonometric function of a reduced argument, either $\alpha\frac{\pi}{4}$ or $(1-\alpha)\frac{\pi}{4}$, which lies in the interval $\left(0,\frac{\pi}{4}\right)$. Payne and Hanek [1] described an efficient algorithm for computing α to a predetermined level of accuracy. They noted that if x differs only slightly from an integral multiple of $\frac{\pi}{2}$, the reduction must be carried out quite accurately to avoid a large loss of significance in the reduced argument. We present a simple method using continued fractions for determining, for all numbers x represented in an IEEE floating-point format, the specific x for which the greatest number of insignificant leading bits occur. Applications are made to IEEE single-precision and double-precision formats and two extended-precision formats.

***Index Terms*—Argument reduction, computer arithmetic, continued fractions, nonlinear optimization, trigonometric functions.**

I. INTRODUCTION

The IEEE Standard for Binary Floating-Point Arithmetic [2] established the simple but powerful concept that the result of an elementary arithmetic operation on floating-point numbers should be obtained as if the exact result were simply rounded to fit in the floating-point format. For the elementary arithmetic operations, many manufacturers have achieved this standard in hardware with fixed-precision arithmetic.

The extension of this concept to elementary transcendental functions would be that the evaluation of an elementary function of floating-point numbers should be obtained as if the exact result were rounded to fit in the floating-point format. Although this is a nice principle, implementation of it is far from trivial. Gal and Bachelis [3] describe how this can be done for almost all argument values.

For many purposes, a less stringent requirement is that the answer should be close to the exact result. One can use a two-step procedure to calculate a trigonometric function of an argument $x > 0$ by first reducing the argument to the first quadrant or octant and then calculating an appropriate function of the reduced argument.

For octant reduction, the first step computes the reduced argument

$$\alpha = \frac{4}{\pi}x - N \text{ where } N = \left\lfloor \frac{4}{\pi}x \right\rfloor; \tag{1}$$

here $x = (N+\alpha)\frac{\pi}{4}$ with N an integer and $0 \le \alpha < 1$. The second step computes $\sin(x)$ or $\cos(x)$ by noting that each of these functions is one of $\pm\left\{\sin\left(\alpha\frac{\pi}{4}\right), \cos\left(\alpha\frac{\pi}{4}\right)\right\}$ when N is even and one of $\pm\left\{\sin\left((1-\alpha)\frac{\pi}{4}\right), \pm\cos\left((1-\alpha)\frac{\pi}{4}\right)\right\}$ when N is odd; the value of N mod 8 determines which sine or cosine must be computed.

Manuscript received Jan. 19, 1994; revised Aug. 12, 1994.

The author is with RISC Software, Motorola, Austin, TX 78735; e-mail: rogers@risc.sps.mot.com.

To order reprints of this article, e-mail: transactions@computer.org, and reference IEEECS Log Number C95125.