NASA Technical Memorandum 101384
ICOMP-88-19

On the Accuracy of Solving Triangular Systems in Parallel

Nai-kuan Tsao
Wayne State University
Detroit, Michigan

and Institute for Computational Mechanics in Propulsion
Lewis Research Center
Cleveland, Ohio

November 1988

On the Accuracy of Solving Triangular Systems in Parallel

Nai-kuan Tsao[*]
Wayne State University
Detroit, Michigan 48202
and
Institute for Computational Mechanics in Propulsion
Lewis Research Center
Cleveland, Ohio 44135

Summary

An error complexity analysis of two algorithms for solving a unit-diagonal triangular system is given. The results show that the usual sequential algorithm is optimal in terms of having the minimal maximum and cumulative error complexity measures. The parallel algorithm described by Sameh and Brent is shown to be essentially equivalent to the optimal sequential one. Some numerical experiments are also included.

[*]This work was supported in part by an NASA-ASEE 1988 Summer Faculty Fellowship and in part by the Space Act Agreement C99066G while the author was visiting ICOMP, NASA Lewis Research Center.

1. Introduction

In [1] Sameh & Brent have shown that, given $\frac{n^3}{68} + O(n^2)$ processors, a triangular system of n equations $Ax = b$ may be solved in $O(\log^2 n)$ steps. They have also shown that if $\tilde{x}$ is the computed solution then it satisfies the equation $(A + \delta A_p)\tilde{x} = b$, where δA_p is bounded by , $||\delta A_p|| \le \alpha(n)\varepsilon\kappa^2(A)||A||$. Here, $||.||$ stands for the ∞-norm, $\alpha(n) = O(n^2 \log n)$, ε is the unit roundoff, $\kappa(A)$ is the condition number of A. On the otherhand, if $\hat{x}$ is the solution computed by the standard sequential algorithm, then it satisfies[2] the equation $(A + \delta A_s)\hat{x} = b$, where $||\delta A_s|| \le n\varepsilon ||A||$. Thus the bound on $||\delta A_p||$ can be very large compared to that on $||\delta A_s||$.

In this paper we present an alternative approach to the error analysis of these two algorithms and show that the parallel algorithm described by Sameh and Brent is essentially equivalent to the usual sequential one in terms of our error complexity measures. Some numerical experiments confirming the theoretical prediction are also presented.

2. Some Preliminary Results

Given a normalized floating-point system with a t-digit base β mantissa, the additive and multiplicative operations can be modelled by the following equations [2]:

$$\begin{aligned} fl(x \times y) &= xy\delta \\ fl(x \pm y) &= (x \pm y)\Delta = x\Delta \pm y\Delta \end{aligned} \tag{2.1}$$

where

$$|\delta|, |\Delta| \le 1 + u, \quad u \le \begin{cases} \frac{1}{2}\beta^{1-t} \text{ for rounded operation} \\ \beta^{1-t} \text{ for chopped operation} \end{cases}$$

and x and y are given machine floating-point numbers and $fl(.)$ is used to denote the computed floating-point result of the given argument. We shall call Δ(or δ) the unit Δ(or δ)-factor.

In general, one can apply (2.1) repeatedly to a sequence of division-free computational steps, and the computed result z can be expressed as:

(2.2) $$z = fl\left(\sum_{j=1}^{\lambda(z)} z_j\right) = \sum_{j=1}^{\lambda(z)} z_j \Delta^{\sigma_j} \delta^{\tilde{\sigma}_j}$$

where each z_j is an exact product of error-free data, and Δ^k (or δ^k) stands for the product of k possibly different Δ (or δ)-factors. Following [3], we shall henceforth call such an exact product of error-free data a basic term. $\lambda(z)$ is then the total number of basic terms whose sum constitutes z.

Note that in (2.2), the computed z is expressed as the exact sum of $\lambda(z)$ perturbed z_j's. Thus the size of σ_j (or $\tilde{\sigma}_j$) is an indication of the possible number of round-off occurrences during the computational process. We define the following two measures:

<u>maximum error complexity:</u>

(2.3) $$\sigma(z) \equiv \max_{1 \le j \le \lambda(z)} [\sigma_j + \tilde{\sigma}_j]$$

<u>cumulative error complexity:</u>

(2.4) $$s(z) \equiv \sum_{j=1}^{\lambda(z)} [\sigma_j + \tilde{\sigma}_j]$$

Different algorithms used to compute the same quantity

$$z = fl\left(\sum_{j=1}^{\lambda(z)} z_j\right)$$

can then be compared using the above error complexity measures.

From (2.3) and (2.4) we can further define the following:

(2.5) $$\sigma_a(z) \equiv \max_{1 \le j \le \lambda(z)} \sigma_j \ , \ \sigma_m(z) \equiv \max_{1 \le j \le \lambda(z)} \tilde{\sigma}_j$$

(2.6) $$s_a(z) \equiv \sum_{j=1}^{\lambda(z)} \sigma_j \ , \ s_m(z) \equiv \sum_{j=1}^{\lambda(z)} \tilde{\sigma}_j$$

Thus $\sigma_a(z)$, $s_a(z)$ or $\sigma_m(z)$, $s_m(z)$ are error complexities due to additive or multiplicative operations. In other words, $\sigma_a(z)$, $s_a(z)$ or $\sigma_m(z)$,$s_m(z)$ are $\sigma(z)$, $s(z)$ evaluated assuming exact multiplications or additions, respectively. Also,

$$\sigma(z) \le \sigma_a(z) + \sigma_m(z),$$
$$s(z) = s_a(z) + s_m(z).$$

Applying (2.3) and (2.4) to (2.1), it is straightforward to establish the following lemma [4]:

Lemma 2.1 If $z = fl(x \pm y)$, then

$$(i) \quad \sigma(z) = 1 + \max(\sigma(x), \sigma(y)),$$
$$s(z) = s(x) + s(y) + \lambda(z),$$
$$\lambda(z) = \lambda(x) + \lambda(y).$$

If $z = fl(x \times y)$, then

$$(ii) \quad \sigma(z) = 1 + \sigma(x) + \sigma(y),$$
$$s(z) = s(x)\lambda(y) + s(y)\lambda(x) + \lambda(z),$$
$$\lambda(z) = \lambda(x)\lambda(y).$$

Often it is more convenient to express a computed result in terms of a sum of some intermediate results. In such cases, we have the following lemma[5]:

Lemma 2.2 If the computed result z can be expressed as

$$z = \sum_{j=1}^{n} z_j \Delta^{\sigma_j} \delta^{\tilde{\sigma}_j}$$

where each z_j is a product of intermediate results, then

$$\lambda(z) = \sum_{j=1}^{n} \lambda(z_j), \quad \sigma(z) = \max_{1 \leq j \leq n} (\tilde{\sigma}_j + \sigma_j + \sigma(z_j))$$

$$s_m(z) = \sum_{j=1}^{n} \lambda(z_j)\tilde{\sigma}_j + \sum_{j=1}^{n} s_m(z_j), \quad s_a(z) = \sum_{j=1}^{n} \lambda(z_j)\sigma_j + \sum_{j=1}^{n} s_a(z_j)$$

In general a basic term is of the form

$$x = \prod_{i=1}^{k} x_i^{\alpha_i}, \quad \alpha_i \geq 1, \quad \lambda(x_i) = 1$$

where each x_i is a single distinct error-free data. We shall now define the multiplicative index of x, or $\mu(x)$, as follows:

$$\mu(x) \equiv \sum_{i=1}^{k} \alpha_i - 1.$$

In other words, $\mu(x)$ is simply the number of sequential multiplications needed to form x. We need the following lemma[5]:

Lemma 2.3 Let

$$z = fl(x) = fl\left(\prod_{i=1}^{k} x_i^{\alpha_i} \right),$$

then

$$\sigma_m(z) = s_m(z) = \mu(x).$$

Lemma 2.3 simply states that the multiplicative error complexities are invariant to the algorithms used to form z, provided that only multiplications are used. We now establish the following lemma:

Lemma 2.4 Given basic terms a, b and it is desired to form

$$c = fl(a \pm b),$$

then

$$\sigma_m(c) = \max(\mu(a), \mu(b)),$$
$$s_m(c) = \mu(a) + \mu(b).$$

provided only associative laws are allowed to find c and the computation of the type $a + ab$ is not evaluated by $a(1 + b)$.

Proof If there are no common factor between a and b, then a and b have to be evaluated separately before the final addition. By Lemma 2.2 we have

$$fl(a) = a\delta^{\mu(a)}, \quad fl(b) = b\delta^{\mu(b)}.$$

Hence

$$c = fl(fl(a) \pm fl(b)) = a\delta^{\mu(a)}\Delta \pm b\delta^{\mu(b)}\delta.$$

By definition

$$\sigma_m(c) = \max(\mu(a), \mu(b)), \quad s_m(c) = \mu(a) + \mu(b).$$

Hence the lemma is true.

If there is a common factor, say x, between a and b, then

$$a = x\tilde{a}, \quad b = x\tilde{b}, \quad \tilde{a}, \tilde{b} \neq 1,$$

and one might choose to compute c as

ORIGINAL PAGE IS
OF POOR QUALITY

$$c = fl(x(\tilde{a} \pm \tilde{b}))$$

once $fl(x), fl(\tilde{x}), fl(\tilde{b})$ are computed. Now by (2.1)

$$\begin{aligned} c &= fl(x) fl(fl(\tilde{a}) \pm fl(\tilde{b}))\delta \\ &= x\delta^{\mu(x)}(\tilde{a}\delta^{\mu(\tilde{a})} \pm \tilde{b}\delta^{\mu(\tilde{b})})\delta\Delta \\ &= x\tilde{a}\Delta\delta^{\mu(x)+\mu(\tilde{a})+1} \pm x\tilde{b}\Delta\delta^{\mu(x)+\mu(\tilde{b})+1} \end{aligned}$$

Hence by definition

$$\begin{aligned} \sigma_m(c) &= \max(\mu(x) + \mu(\tilde{a}) + 1, \mu(x) + \mu(\tilde{b}) + 1) \\ &= \max(\mu(a), \ \mu(b)) \end{aligned}$$

and

$$\begin{aligned} s_m(c) &= \mu(x) + \mu(\tilde{a}) + 1 + \mu(x) + \mu(\tilde{b}) + 1 \\ &= \mu(a) + \mu(b). \qquad \text{Q.E.D.} \end{aligned}$$

By repeated application of Lemma 2.4 to the evaluation of (2.2) we can easily establish the following theorem:

Theorem 2.1 The computed z of (2.2) is such that

$$\sigma_m(z) = \max_{1 \le j \le \lambda(z)} \mu(z_j),$$
$$s_m(z) = \sum_{j=1}^{\lambda(z)} \mu(z_j).$$

In other words, Theorem 2.1 states that the multiplicative error complexities are invariant to the algorithms used to evaluate z. Henceforth we shall only look at the additive error complixities in the evaluation of different algorithms for the computation of the type of (2.2). This is equivalent to having exact multiplication operations possible for the computation of (2.2). We need the following theorem:

Theorem 2.2 If in (2.2) $\lambda(z) = 2^k$ and $2^k - 1$ additions are used to evaluate z, then

$$\sigma_a(z) \geq k, \quad s_a(z) \geq k2^k.$$

Proof The computation of z in (2.2) is equivalent to the construction of a binary tree with 2^k leaves at the top and $2^k - 1$ interior nodes of additions with z the output of the bottom root node. In such case then $\sigma_a(z)$ is the height of the tree and $s_a(z)$ is the sum of the lengths of all the paths from the leaf nodes to the root node. Q.E.D.

An important type of computation of (2.2) is the evaluation of the inner product given as

(2.7) $$z = fl\left(\sum_{i=1}^{k} x_i y_i\right)$$

we need to specify the order in which the additions are executed. We discuss several strategies.

If the products are added recursively in parallel by divide-and- conquer, then the strategy is called left-heavy if

$$z = fl(z_1 + z_2)$$

where

$$z_1 = fl\left(\sum_{i=1}^{\lceil k/2 \rceil} x_i y_i\right), \quad z_2 = fl\left(\sum_{i=\lceil k/2 \rceil + 1}^{k} x_i y_i\right)$$

Similarly the strategy is called right-heavy if

$$z = fl(z_3 + z_4), \quad z_3 = fl\left(\sum_{i=1}^{\lfloor k/2 \rfloor} x_i y_i\right) \quad z_4 = fl\left(\sum_{i=\lfloor k/2 \rfloor + 1}^{k} x_i y_i\right).$$

If the inner product is summed up in sequential order, then we have the common strategies of left-to-right or right-to-left.

We now establish the following theorem:

Theorem 2.3 Assuming exact multiplications are possible in evaluating the x_i, y_i and xy_i of (2.7) and

$$\begin{aligned} \sigma_a(x_1) > \sigma_a(x_2) > \dots > \sigma_a(x_{k-1}) = \sigma_a(x_k) \geq 0, \\ \sigma_a(y_1) = \sigma_a(y_2) = \dots = \sigma_a(y_{k-1}) = \sigma_a(y_k) \geq 0, \end{aligned} \tag{2.8}$$

then the computed z of (2.7) is such that

$$\sigma_a(z) = \sigma_a(x_1) + \sigma_a(y_1) + w$$

where

$$w = \begin{cases} \lceil \log k \rceil & \text{if the strategy is left-heavy,} \\ \lfloor \log k \rfloor & \text{if the strategy is right-heavy,} \\ k-1 & \text{if the strategy is left-to-right,} \\ 1 & \text{if the strategy is right-to-left.} \end{cases}$$

Proof We first consider the last two cases. If the strategy is left-to-right , then we can easily obtain

$$z = x_1 y_1 \Delta^{k-1} + x_2 y_2 \Delta^{k-1} + \dots + x_k y_k \Delta^1$$

Hence we have

$$\begin{aligned} \sigma_a(z) &= \max(\sigma_a(x_1) + \sigma_a(y_1) + k - 1,\ \sigma_a(x_2) + \sigma_a(y_2) + k - 1,\ \dots\ , \sigma_a(x_k) + \sigma_a(y_k) + 1) \\ &= \sigma_a(x_1) + \sigma_a(y_1) + k - 1 \end{aligned}$$

and the theorem is true.

If the strategy is right-to-left, then we have

$$z = x_1 y_1 \Delta^1 + \ \dots \ + x_{k-1} y_{k-1} \Delta^{k-1} + x_k y_k \Delta^{k-1}$$

By (2.8) we have

$$\sigma_a(x_1) \geq \sigma_a(x_2) + 1 \geq \dots \geq \sigma_a(x_{k-1}) + k - 2 = \sigma_a(x_k) + k - 2$$

Hence

$$\sigma_a(x_1) + \sigma_a(y_1) + 1 \geq \sigma_a(x_2) + \sigma_a(y_2) + 2 \geq \dots \geq \sigma_a(x_{k-1}) + \sigma_a(y_{k-1}) + k - 1 = \sigma_a(x_{k-1}) + \sigma_a(y_{k-1}) + k - 1$$

And indeed

$$\sigma_a(z) = \sigma_a(x_1) + \sigma_a(y_1) + 1$$

For the parallel strategies, we prove by induction. For $k = 1$ the theorem is trivial. Assume it is true for $k - 1$ expressed as

$$k - 1 = \beta_j \beta_{j-1} \dots \beta_1 \beta_0, \ \beta_j = 1$$

in binary form. For k then if the strategy is left-heavy, we have

$$z = fl\left(\sum_{i=1}^{\lceil k/2 \rceil} x_i y_i + \sum_{i = \lceil k/2 \rceil + 1}^{k} x_i y_i \right)$$

By assumption

$$\begin{aligned} \sigma_a(z) &= 1 + \max(\sigma_L, \ \sigma_R) \\ &= \sigma_a(x_1) + \sigma_a(y_1) + 1 + \lceil \log \lceil k/2 \rceil \rceil, \end{aligned}$$

where

$$\sigma_L = \sigma_a(x_1) + \sigma_a(y_1) + \lceil \log \lceil k/2 \rceil \rceil$$

$$\sigma_R = \sigma_a(x_{\lceil k/2 \rceil + 1}) + \sigma_a(y_{\lceil k/2 \rceil + 1}) + \lceil \log(k - \lceil k/2 \rceil) \rceil$$

Since

$$k - 1 = \beta_j \beta_{j-1} \cdots \beta_1 \beta_0$$

Hence

$$k = \beta_j \cdots \beta_1 \beta_0 + 1$$

And we have

$$2^j < k \leq 2^{j+1}$$

$$2^{j-1} < \lceil k/2 \rceil \leq 2^j$$

$$j - 1 < \lceil \log \lceil k/2 \rceil \rceil \leq j$$

Therefore

$$1 + \lceil \log \lceil k/2 \rceil \rceil = j + 1 = \lceil \log k \rceil$$

So the theorem is true. Similar reasoning can be used to show the truth of the theorem for the right-heavy strategy. Q.E.D.

3. Error Complexity Analysis

Given a unit-diagonal lower triangular system

(3.1) $$Ax = b$$

where

$$A = \begin{bmatrix} 1 & & & \\ a_{21} & 1 & & \\ \vdots & & \ddots & \\ a_{n1} & \cdots & a_{n,n-1} & 1 \end{bmatrix}, \; b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}, \; x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix},$$

then the exact solution x_i can be expressed as

$$(3.2) \qquad x_i = (-1)^{i-1} \det \begin{bmatrix} b_1 & 1 & & & \\ b_2 & a_{21} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \vdots & \vdots & & \ddots & \ddots \\ b_i & a_{i1} & \cdots & a_{i,i-1} & 1 \end{bmatrix}$$

Thus the evaluation of x_i is equivalent to the evaluation of the determinant of an i by i lower Hessenberg matrix with unity super-diagonal elements. We assume the given A and b are error free original data with $\lambda(a_{ij}) = \lambda(b_i) = 1$ and $\mu(a_{ij}) = \mu(b_i) = 0$. Denoting by t_i the generic computation of such a matrix, then it is easily shown (by expanding the first row of the above determinant) that

$$(3.3) \qquad t_i = fl(t_{i-1} + wt_{i-1}), \ \lambda(w) = 1, \ \mu(w) = 0$$

where w is error-free. It is obvious that

$$\lambda(t_i) = 2\lambda(t_{i-1}), \ \lambda(t_1) = 1,$$

hence

$$(3.4) \qquad \lambda(t_i) = 2^{i-1}.$$

Furthermore we have the following lemma

<u>Lemma 3.1</u> The computation of t_i requires at most $2^{i-1} - 1$ additive operations.

<u>Proof</u> Denoting a_i as the number of additive operations needed to compute t_i ; it is obvious from (3.3) that

$$a_i = 2a_{i-1} + 1, \quad a_1 = 0.$$

The solution is given as

$$a_i = 2^{i-1} - 1.$$

This proves the lemma. Q.E.D.

Lemma 3.1 together with (3.4) and Theorem 2.2 gives us the following theorem:

Theorem 3.1 The computation of x_i requires at most a total of $2^{i-1} - 1$ additive operations with

$$\sigma_a(x_i) \geq i - 1,$$
$$s_a(x_i) \geq (i-1)2^{i-1}.$$

The solution of (3.1) can also be expressed as the following:

(3.5) $$x = M_{n-1}M_{n-2}...M_2M_1b$$

where

$$M_i = I - a_ie_i^T,$$

$$a_i = [0(1{:}i), a_{i+1,i}, \dots, a_{ni}]^T$$

and e_i is the i-th column of the identity matrix I. Note in the above expression we use $a(b{:}c)$ to denote a sub-vector of identical component a placed in the b-th to c-th positions of a larger vector.

The usual sequential algorithm can then be expressed as

```
$x^{(0)} = b$
for $i = 1$ to $n - 1$ do
    $x^{(i)} = fl(M_ix^{(i-1)})$
$x = x^{(n-1)}$
```

or more specifically,

(3.6)

```
for $j = 1$ to $n$ do
  $x_j^{(0)} = b_j$
for $i = 1$ to $n$ do
  $x_i = x_i^{(i-1)}$
  for $j = 1$ to $i$ do
    $x_j^{(i)} = x_j^{(i-1)}$
  for $j = i + 1$ to $n$ do
    $x_j^{(i)} = fl(x_j^{(i-1)} - a_{ji}x_i^{(i-1)})$
```

We have the following lemma:

Lemma 3.2 The computation of $x^{(i)}$ is equivalent to the computation of

$$t^{(i)} \equiv [t_1\ t_2 \dots t_i\ t_{i+1}(i+1{:}n)\ ]^T$$

Proof We only need to notice that the inner loop computation of (3.6) is essentially of the type $fl(t_i + t_1 t_i)$ which is t_{i+1} by definition. Q.E.D.

Applying Lemma 2.1 (i) to the inner loop of (3.6) and assuming exact multiplications, we have for $j > i$

$$\sigma_a(x_j^{(i)}) = 1 + \max(\sigma_a(x_j^{(i-1)}), \sigma_a(x_i))$$
$$s_a(x_j^{(i)}) = s_a(x_j^{(i-1)}) + s_a(x_i) + \lambda(x_j^{(i)}) \qquad \lambda(x_1) = 1,\ \ s_a(x_1) = \sigma_a(x_1) = 0$$
$$\lambda_a(x_j^{(i)}) = \lambda(x_j^{(i-1)}) + \lambda(x_i)$$

The solutions to the above equations are given in the following theorem:

Theorem 3.2 The sequential algorithm of (3.6) produces results such that for $j > i$

$$\sigma_a(x_j^{(i)}) = i,\ \ s_a(x_j^{(i)}) = i2^i,\ \ \lambda(x_j^{(i)}) = 2^i,$$

$$\sigma_a(x_i) = i - 1,\ \ s_a(x_i) = (i-1)2^{i-1},\ \lambda(x_i) = 2^{i-1}.$$

Comparing the results in the above theorem to those in Theorem 3.1, we conclude that the sequential algorithm is optimal in terms of having the minimal maximum and cumulative error complexities.

We now turn our attention to the parallel algorithm as proposed by Sameh and Brent[1]. The algorithm for $n = 2^v$ is given as follows:

(3.7)

```
for $i = 1$ to $2^v - 1$ do
    $M_i^{(0)} = M_i$
$b^{(0)} = b$
for $j = 0$ to $v - 1$ do
    for $k = 2^{v-j-1} - 1$ downto 1 do
        $M_k^{(j+1)} = fl(M_{2k+1}^{(j)} \times M_{2k}^{(j)})$
    $b^{(j+1)} = fl(M_1^{(j)} b^{(j)})$
$x = b^{(v)}$
```

First the λ matrix of $M_k^{(0)}$ can easily be obtained as:

$$\lambda(M_k^{(0)}) = I + \lambda(a_k)e_k^T$$

where

$$\lambda(a_k) = [0(1{:}k),\ 1(k+1{:}n)]^T.$$

Then we have the following theorem:

<u>Theorem 3.3</u>

$$\lambda(M_k^{(j+1)}) = \lambda(M_{2k+1}^{(j)})\lambda(M_{2k}^{(j)}) = \begin{bmatrix} I_p & & \\ & \lambda(L^{(j+1)}) & \\ & \lambda(R_k^{(j+1)}) & I_q \end{bmatrix},$$

$$\lambda(b^{(j+1)}) = \lambda(M_1^{(j)})\lambda(b^{(j)}) = [1\ 2^1\ 2^2 \ldots 2^{2^{j+1}-1}(2^{j+1}{:}n)]^T$$

where

$$p = k2^{j+1} - 1, \quad q = 2^{\nu} - p - 2^{j+1}$$

$$\lambda(L^{(j+1)}) = \begin{bmatrix} 1 & & & \\ 2^0 & 1 & & \\ \vdots & \ddots & \ddots & \\ 2^{2^{j+1}-2} & \cdots & 2^0 & 1 \end{bmatrix}, \quad \lambda(R_k^{(j+1)}) \equiv \begin{bmatrix} \lambda(U_k^{(j+1)}) \\ \lambda(V_k^{(j+1)}) \end{bmatrix} = \left.\begin{bmatrix} 2^{2^{j+1}-1} & 2^{2^{j+1}-2} & \cdots & 2^0 \\ \vdots & \vdots & & \vdots \\ 2^{2^{j+1}-1} & 2^{2^{j+1}-2} & \cdots & 2^0 \end{bmatrix}\right\} \text{q rows}$$

and $\lambda(U_k^{(j+1)})$ and $\lambda(V_k^{(j+1)})$ are the first 2^{j+1} and the last $q - 2^{j+1}$ rows of $\lambda(R_k^{(j+1)})$, respectively.

Proof See Appendix I.

If we define $\sigma(A)$ as a matrix whose (i,j)-th component is $\sigma(a_{ij})$ then we have the following theorem:

Theorem 3.4 $\sigma(M_k^{(j+1)})$ is of the same structure as that of $\lambda(M_k^{(j+1)})$. Furthermore let $\sigma_{gh} \equiv$ the (g,h)-th element of $\sigma(M_k^{(j+1)})$, then we have

$$\sigma_{gh}\begin{Bmatrix} > 0 \text{ if } g - h \geq 2 \\ = 0 \text{ otherwise} \end{Bmatrix} \quad p+1 \leq h \leq p + 2^{j+1}, \quad p+1 \leq g \leq 2^{\nu} - p$$

$$\sigma_{g,p+1} > \sigma_{g,p+2} > \ \ldots\ > \sigma_{g,p+2^{j+1}}$$

$$\sigma_{p+1,h} < \sigma_{p+2,h} < \ \ldots\ < \sigma_{p+2^{j+1},h} = \ \ldots\ = \sigma_{2^{\nu}-p,h}$$

$$\sigma_{p+1,p} = \sigma_a(x_1) < \sigma_{p+2,p} = \sigma_a(x_2) < \ \ldots\ < \sigma_{p+2^{j+1},p} = \ \ldots\ = \sigma_{2^{\nu}-p,p} = \sigma_a(x_{2^{j+1}})$$

Proof See Appendix II.

With the general property established in Theorem 3.4, we have the following theorem for $\sigma(x)$:

Theorem 3.5 If in (3.7) the inner products are evaluated using either the left-heavy or the right-heavy strategy, then the computed x is such that

$$\sigma_a(x_i) \le 1.5(i-1)$$

Proof At the (j + 1)-th step let us denote by m_{gh} the (g,h)th component of $M_1^{(j)}$. Now by construction

$$x_{2^j+i} = b^{(j)}_{2^j+i}, \quad 1 \le i \le 2^j$$

To calculate

$$b^{(j+1)} = fl(M_1^{(j)} b^{(j)})$$

we have

$$b^{(j+1)}_{2^j+i} = fl\left(\sum_{k=2^j}^{2^j+i-1} m_{2^j+i,k} b^{(j)}_k + b_{2^j+i} \right), \quad 1 \le i \le 2^j$$

For $i = 2^j$ we have

$$x_{2^{j+1}} = b^{(j+1)}_{2^{j+1}} = fl\left(\sum_{k=2^j}^{2^{j+1}-1} m_{2^{j+1},k} b^{(j)}_k + b^{(j)}_{2^{j+1}} \right)$$

where the summation (inner product) is evaluated by either the left-heavy or the right-heavy parallel strategy. Now by assumption

$$\sigma_a(b^{(j)}_k) = \sigma_a(x_{2^j}), \quad 2^j + 1 \le k \le 2^{j+1}$$

Also

$$\sigma_a(m_{2^{j+1},k}) > \sigma_a(m_{2^{j+1},k+1}), \quad 2^j + 1 \le k \le 2^{j+1} - 2$$

$$\sigma_a(m_{2^{j+1},2^j+1}) = \sigma_a(x_{2^j})$$

Hence by Theorem 2.3 we have

$$\sigma_a(b_{2^{j+1}}^{(j+1)}) = \sigma_a(x_{2^{j+1}}) = \sigma_a(m_{2^{j+1},2^j+1}) + \sigma_a(x_{2^j}) + j = 2\sigma_a(x_{2^j}) + j, \quad \sigma_a(x_1) = 0.$$

The solution to the above equation is given as

$$\sigma_a(x_{2^{j+1}}) = 3(2^j) - j - 2, \quad 0 \le j+1 \le v$$

For general i we also have by Theorem 2.3 that

$$\sigma_a(x_{2^j+i}) = \sigma_a(x_i) + \sigma_a(x_{2^j}) + |\log(i+1)|$$

where $|.|$ is used to denote either $\lceil . \rceil$ or $\lfloor . \rfloor$.

Now if

$$i = \beta_r \beta_{r-1} \cdots \beta_0$$

$$\beta_r = 1, \quad \beta_f \in \{0,1\}, \quad 0 \le f \le r-1$$

then $\sigma_a(x_i)$ can further be expressed as

$$\begin{aligned}
\sigma_a(x_i) &= \beta_r \sigma_a(x_{2^r}) + \sigma_a(x_{i-\beta_r 2^r}) + \beta_r |\log(i+1-\beta_r 2^r)| \\
&= \beta_r \sigma_a(x_{2^r}) + \beta_{r-1}\sigma_a(x_{2^{r-1}}) + \sigma_a(x_{i-\beta_r 2^r - \beta_{r-1}2^{r-1}}) \\
&\qquad + \beta_r |\log(i+1-\beta_r 2^r)| + \beta_{r-1} |\log(i+1-\beta_r 2^r - \beta_{r-1}2^{r-1})| \\
&= \ldots \le \sum_{f=1}^{r} \beta_f \sigma_a(x_{2^f}) + \sigma_a(x_{\beta_0 2^0}) \\
&\qquad + \beta_r |\log 2^r| + \beta_{r-1} |\log 2^{r-1}| + \ldots + \beta_1 |\log 2^1| \\
&\le \sum_{f=1}^{r} \beta_f (3(2^{f-1}) - 1) = \sum_{f=1}^{r} \beta_f (2^f + 2^{f-1} - 1) \le \frac{3}{2}(i-1) \quad \textbf{Q.E.D.}
\end{aligned}$$

If the inner products are evaluated in a sequential manner, then similar reasoning can be used to establish the following theorem:

<u>Theorem 3.6</u> If in (3.7) the inner products are evaluated in sequence, then the computed x is such that

(*i*) if the strategy is left-to-right, then

$$\sigma_a(x_i) \leq \lceil \log i \rceil (2^{\lceil \log i \rceil} - 1)$$

(*ii*) if the strategy is right-to-left, then

$$\sigma_a(x_i) = i - 1$$

$$\sigma_{gh} = \begin{cases} g - h - 1 \text{ if } g - h \geq 2 \\ 0 \text{ otherwise} \end{cases} \quad \begin{matrix} p + 1 \leq h \leq p + 2^j, \\ p + 1 \leq g \leq 2^\nu - p \end{matrix}$$

where σ_{gh} is used to denote the (g,h)-th element of $\sigma_a(M_k^{(j)})$ and $p = k2^{j+1} - 1$.

We can now summarize the results as follows:

$$(3.8) \qquad \sigma_a(x_i) \begin{cases} = i - 1 \textbf{ If the strategy is right-to-left or the algorithm of (3.6) is used,} \\ \leq 1.5(i - 1) \textbf{ if the strategy is left-heavy or right-heavy,} \\ \leq \lceil \log i \rceil (2^{\lceil \log i \rceil} - 1) \textbf{ if the strategy is left-to-right.} \end{cases}$$

The cumulative error complexities can then be bounded using (3.8) as follows:

$$(3.9) \quad s_a(x_i) \begin{cases} = (i - 1)2^{i-1} \textbf{ If the strategy is right-to-left or the algorithm of (3.6) is used,} \\ \leq 1.5(i - 1)2^{i-1} \textbf{ if the strategy is left-heavy or right-heavy,} \\ \leq 2^{i-1} \lceil \log i \rceil (2^{\lceil \log i \rceil} - 1) \textbf{ if the strategy is left-to-right.} \end{cases}$$

We conclude that the parallel algorithm (3.7) is as accurate as the sequential algorithm (3.6) if the parallel inner products are evaluated using the strategy of right-to-left. For other strategies we can easily obtain from (3.8) and (3.9) that

$$\frac{\sigma_a(x_i) \textbf{ resulting from (3.7)}}{\sigma_a(x_i) \textbf{ resulting from (3.6)}} \leq \begin{cases} 1.5 \textbf{ if the strategy is left-heavy or right-heavy,} \\ \lceil \log i \rceil \textbf{ if the strategy is left-to-right.} \end{cases}$$

$$\frac{s_a(x_i) \textbf{ resulting from (3.7)}}{s_a(x_i) \textbf{ resulting from (3.6)}} \leq \begin{cases} 1.5 \textbf{ if the strategy is left-heavy or right-heavy,} \\ \lceil \log i \rceil \textbf{ if the strategy is left-to-right.} \end{cases}$$

Hence in all cases the parallel algorithm is essentially 'equivalent' to the usual sequential algorithm in terms of our error complexity measures.

4. Numerical Experiments and Conclusion

In the first experiment a 64 by 64 lower triangular system satisfying

$$x_{i+1} = 4x_i - x_{i-1} + 1, \quad x_1 = 1, \quad x_2 = 5$$

is solved in Pascal shortreal using an IBM 370 machine. The unit round-off is 16^{-5}. If we denote by $e_{seq}(x_i)$ and $e_{par}(x_i)$, respectively, the absolute error of x_i produced by the sequential and parallel algorithm, then a selected sample of errors is shown below:

k	$e_{seq}(x_{4k})$	$e_{par}(x_{4k})$
1	0	0
2	0	0
3	0	0
4	3.68E02	8.80E02
5	2.26E05	2.91E05
6	8.52E07	5.16E07
7	2.15E10	6.99E09
8	4.50E12	1.48E12
9	1.02E15	3.02E14
10	2.40E17	5.97E16
11	5.10E19	1.07E19
12	1.08E22	2.58E21
13	2.29E24	4.78E23
14	4.75E26	1.07E26
15	1.04E29	2.47E28
16	2.38E31	4.80E30

For the second experiment a set of 100 random lower triangular matrices A with unit diagonal elements and 100 vectors b are generated such that

$$0 \geq a_{ij} \geq -1, \quad 0 \leq b_i \leq 1, \quad 1 \leq i,j \leq 64, \quad i-j \geq 1.$$

The systems are solved using an IBM PC with an 8087 coprocessor. The unit round-off is 2^{-23}. The cumulative absolute error of all x_i produced by the sequential and parallel algorithm are represented by $ce_{seq}(x_i)$ and $ce_{par}(x_i)$, respectively. A selected set of errors is given below:

k	$ce_{seq}(x_{4k})$	$ce_{par}(x_{4k})$
1	$5.16E-6$	$3.88E-6$
2	$3.52E-5$	$3.16E-5$
3	$1.83E-4$	$1.91E-4$
4	$9.85E-4$	$1.02E-3$
5	$5.76E-3$	$5.37E-3$
6	$3.29E-2$	$2.75E-2$
7	$1.75E-1$	$1.74E-1$
8	$8.45E-1$	$7.42E-1$
9	$4.29E00$	$3.47E00$
10	$1.79E01$	$1.85E01$
11	$9.43E01$	$9.10E01$
12	$5.69E02$	$5.12E02$
13	$3.12E03$	$2.36E03$
14	$1.57E04$	$1.67E04$
15	$8.90E04$	$6.61E04$
16	$4.61E05$	$4.67E05$

We see from the above tables that the numerical results produced by the parallel algorithm are as accurate as those produced by the usual sequential algorithm. In the first experiment the parallel results can even be classified as slightly 'better' than the sequential ones.

References

[1] A.H. Sameh and R.P. Brent, Solving triangular systems on a parallel computer, SIAM J. Numer. Anal., 14(1977), pp.1101-1113.

[2] J.H. Wilkinson, Rounding Errors in Algebraic Processes, Prentice-Hall, Englewood Cliffs, NJ, 1963.

[3] V.B. Aggarwal and J.W. Burgmeier, A round-off error model with applications to arithmetic expressions, SIAM J. Computing, 8(1979), pp.60-72.

[4] N.K. Tsao, Error complexity analysis of algorithms for matrix multiplication and matrix chain product, IEEE Trans. Computers, C-30(1981), pp.758-771.

[5] N.K. Tsao, A simple approach to the error analysis of division-free numerical algorithms, IEEE Trans. Computers, C-32(1983), pp.343-351.

Appendix I. Proof of Theorem 3.3

To show the validity of

$$(A.1) \qquad \lambda(M_k^{(j+1)}) = \lambda(M_{2k+1}^{(j)})\lambda(M_{2k}^{(j)}),$$

we first show that for general matrix multiplication

$$C = fl(A \times B), \quad A \in R^{m \times n}, \quad B \in R^{n \times r}, \quad C \in R^{m \times r},$$

we have

$$\lambda(C) = \lambda(A)\lambda(B).$$

By definition

$$c_{ij} = fl\left(\sum_{k=1}^{n} a_{ik}b_{kj}\right),$$

then we have, by repeated application of Lemma 2.1,

$$\lambda(c_{ij}) = \sum_{k=1}^{n} \lambda(a_{ik})\lambda(b_{kj}).$$

So we have

$$\lambda(C) = \lambda(A)\lambda(B).$$

The validity of (A.1) can then be shown by direct substitution of the results for $\lambda(M_{2k+1}^{(j)})$ and $\lambda(M_{2k}^{(j)})$ and $\lambda(M_k^{(j+1)})$ into (A.1). Q.E.D.

Appendix II. Proof of Theorem 3.4

First of all , the nontrivial part of $\sigma_a(M_k^{(j+1)})$ is of the same structure as that of $\lambda(M_k^{(j+1)})$. Furthermore, the diagonals and subdiagonals of $\lambda(M_k^{(j+1)})$ consist of only one basic term each. Hence no additive operations are involved. And we have

$$\sigma_{gh}\begin{cases} >0 \ \textit{if } g-h\geq 2, \\ =0 \ \textit{otherwise.}\end{cases} \quad p+1\leq h\leq p+2^{j+1}, \quad p+1\leq g\leq 2^{v}-p.$$

Let us assume that the rest of the theorem is true for $M_{2k+1}^{(j)}$ and $M_{2k}^{(j)}$. Now

$$M_k^{(j+1)} = fl(M_{2k+1}^{(j)}M_{2k}^{(j)})$$
$$= \begin{bmatrix} I_p & & \\ & L_k^{(j+1)} & \\ & R_k^{(j+1)} & I_q \end{bmatrix}, \quad R_k^{(j+1)} = \begin{bmatrix} U_k^{(j+1)} \\ V_k^{(j+1)} \end{bmatrix}$$

where

$$L_k^{(j+1)} = \begin{bmatrix} L_{2k}^{(j)} & \\ fl(L_{2k+1}^{(j)}U_{2k}^{(j)}) & L_{2k+1}^{(j)} \end{bmatrix},$$

$$R_k^{(j+1)} = [fl(R_{2k+1}^{(j)}U_{2k}^{(j)} + V_{2k}^{(j)}) \quad R_{2k+1}^{(j)}].$$

The submatrices $L_{2k}^{(j)}$, $L_{2k+1}^{(j)}$ in $L_k^{(j-1)}$ and $R_{2k+1}^{(j)}$ in $R_k^{(j+1)}$ will retain the same properties as stated in the theorem. Let

$$X = fl(L_{2k+1}^{(j)}U_{2k}^{(j)}), \quad Y = fl(R_{2k+1}^{(j)}U_{2k}^{(j)} + V_{2k}^{(j)}),$$

and

$$L = L_{2k+1}^{(j)}, \quad R = R_{2k+1}^{(j)}, \quad U = U_{2k}^{(j)}, \quad V = V_{2k}^{(j)}.$$

Then

ORIGINAL PAGE IS OF POOR QUALITY

$$x_{mn} = fl\left(\sum_{k=1}^{m} l_{mk}u_{kn}\right), \quad 1 \le m,n \le 2^j$$

with

$$\sigma_a(l_{m1}) > \sigma_a(l_{m2}) > \ \dots \ > \sigma_a(l_{mn}),$$

$$\sigma_a(u_{1n}) = \sigma_a(u_{2n}) = \ \dots \ = \sigma_a(u_{mn}) \equiv \sigma_n > \sigma_{n+1}.$$

Hence by Theorem 2.3 we have

$$\begin{aligned}\sigma_a(x_{mn}) &= \sigma_a(l_{m1}) + \sigma_n + w(m)\\ &< \sigma_a(l_{m+1,1}) + \sigma_n + w(m+1) = \sigma_a(x_{m+1,n})\end{aligned}$$

where

$$w(m) = \begin{cases} \lceil \log m \rceil & \text{if the strategy is left-heavy,}\\ \lfloor \log m \rfloor & \text{if the strategy is right-heavy,}\\ k-1 & \text{if the strategy is left-to-right,}\\ 1 & \text{if the strategy is right-to-left.}\end{cases}$$

Also

$$\sigma_a(x_{mn}) > \sigma_a(l_{m1}) + \sigma_{n+1} + w(m) = \sigma_a(x_{m,n+1})$$

by assumption. Hence the ordered property for the σ's in $L_k^{(j+1)}$ is preserved. A similar argument can also be used to show the same property is true for the matrix $R_k^{(j+1)}$. Q.E.D.

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No. NASA TM-101384 ICOMP-88-19	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle On the Accuracy of Solving Triangular Systems in Parallel		5. Report Date November 1988
		6. Performing Organization Code
7. Author(s) Nai-kuan Tsao		8. Performing Organization Report No. E-4439
		10. Work Unit No. 505-62-21
9. Performing Organization Name and Address National Aeronautics and Space Administration Lewis Research Center Cleveland, Ohio 44135-3191		11. Contract or Grant No.
		13. Type of Report and Period Covered Technical Memorandum
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546-0001		14. Sponsoring Agency Code

15. Supplementary Notes

Nai-kuan Tsao, Wayne State University, Detroit, Michigan 48202. This work was supported in part by a NASA-ASEE 1988 Summer Faculty Fellowship and in part by the Institute for Computational Mechanics in Propulsion, NASA Lewis Research Center (work funded under Space Act Agreement C99066G).

16. Abstract

An error complexity analysis of two algorithms for solving a unit-diagonal triangular system is given. The results show that the usual sequential algorithm is optimal in terms of having the minimal maximum and cumulative error complexity measures. The parallel algorithm described by Sameh and Brent is shown to be essentially equivalent to the optimal sequential one. Some numerical experiments are also included.

17. Key Words (Suggested by Author(s)) Triangular system; Parallel algorithm; Error complexity; Sequential algorithm; Round-off error; Floating-point computation	18. Distribution Statement Unclassified – Unlimited Subject Category 64

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No of pages	22. Price*
Unclassified	Unclassified	26	A03

*For sale by the National Technical Information Service, Springfield, Virginia 22161