[16] V. Pan and J. Reif. Efficient parallel solution of linear systems. In *Proc. 17th Annual ACM Symposium on Theory of Computing*, pages 143–152, 1985.

[17] F. P. Preparata. Inverting a Vandermonde matrix in minimum parallel time. *Inform. Process. Lett.*, 38:291–294, 1991.

[18] A. H. Sameh and D. J. Kuck. On stable parallel linear systems solvers. *J. Assoc. Comput. Mach.*, 25:81–91, 1978.

[19] J. F. Traub, G. W. Wasilkowski, and H. Wozniakowski. *Information-Based Complexity.* Academic Press, San Diego, Ca, 1988.

[20] J. von zur Gathen. Parallel linear algebra. In J. Reif, editor, *Synthesis of Parallel Algorithms.* to appear.

[21] J. von zur Gathen. Parallel arithmetic computations: a survey. In *Lecture notes in Computer Science*, volume 233, pages 93–122. Springer-Verlag, New-York, 1986.

References

[1] K. E. Atkinson. *An Introduction to Numerical Analysis*. Wiley, New York, 1978.

[2] S. J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Inform. Process. Lett.*, 18:147–150, 1984.

[3] L. Blum, M. Shub, and S. Smale. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bull. Amer. Math. Soc. (N.S.)*, 21:1–46, 1989.

[4] A. Borodin, J. von zur Gathen, and J. Hopcroft. Fast parallel matrix and GCD computations. *Inform. and Control*, 52:241–256, 1982.

[5] A.K. Chandra. Maximal parallelism in matrix multiplication. Technical Report RC-6193, IBM Res., 1976.

[6] B. Codenotti. Parallel solution of linear systems by repeated squaring. *Applied Math. Letters*, 3:19–20, 1990.

[7] B. Codenotti and F. Flandoli. A monte carlo method for the parallel solution of linear systems. *Journal of Complexity*, 5:107–117, 1989.

[8] B. Codenotti and M. Leoncini. *Parallel Complexity of Linear System Solution.* World Scientific Pu. Co., Singapore, 1991.

[9] S. A. Cook. A taxonomy of problems with fast parallel algorithms. *Inform. and Control*, 64:2–22, 1985.

[10] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progression. In *Proc. 19th Annual ACM Symposium on Theory of Computing*, pages 1–6, Berkeley, CA, 1987. Springer-Verlag.

[11] M. Cosnard and Y. Robert. Complexity of parallel QR factorization. *J. Assoc. Comput. Mach.*, 33:712–723, 1986.

[12] P. Crescenzi and A. Panconesi. Completeness in approximation classes. *Inform. and Control*, 93:241–262, 1991.

[13] L. Csanky. Fast parallel matrix inversion algorithms. *SIAM J. Comput.*, 5:618–623, 1976.

[14] G. H. Golub and C. F. van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, Md, 1983.

[15] K. Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. In *Proc. 18th Annual ACM Symposium on Theory of Computing*, pages 338–339, 1986.

It is plain that

$$\sum_{i=0}^{n} ||L^i||_\infty = 1 + \sum_{1\le i\le n} \sum_{1\le j\le n+1} f_{ij}. \tag{12}$$

We now consider the symmetric square matrix $P'_n = (p'_{ij})$ obtained from P_n by adding, as the last row of P'_n, the first row of L^{n+1}. For instance

$$P'_4 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 3 & 6 & 10 & 15 \\ 1 & 4 & 10 & 20 & 35 \\ 1 & 5 & 15 & 35 & 70 \end{pmatrix}.$$

It then holds that

$$p'_{ij} = \binom{i+j-2}{\max\{i,j\}-1}.$$

Let S_n and S'_n denote the sums of the elements of P_n and P'_n, respectively. Then it holds that

$$\begin{aligned} S_n &= S'_n - \sum_{j=1}^{n+1} \binom{n+1+j-2}{n} \\ &= S'_n - \sum_{j=0}^{n} \binom{n+j}{j} \\ &= S'_n - \binom{2n+1}{n+1} \\ &= S'_n - \binom{2n+1}{n}, \end{aligned} \tag{13}$$

and that

$$S'_n = 2S_n - 1. \tag{14}$$

Combining (??) and (??) and solving for S_n gives

$$S_n = \binom{2n+1}{n} + 1.$$

Finally, using (??) we obtain

$$\sum_{i=0}^{n} ||L^i||_\infty = S_n - 1 = \binom{2n+1}{n}.$$

■

Among the p norms, we report in the following the most popular ones.

$$\|\mathbf{x}\|_1 = \sum_i |x_i|,$$
$$\|\mathbf{x}\|_2 = \sqrt{\sum_i |x_i|^2},$$
$$\|\mathbf{x}\|_\infty = \max_i |x_i|.$$

Using (**??**) with the definition of p norm we obtain

$$\|A\|_1 = \max_j \sum_i |a_{ij}|,$$
$$\|A\|_2 = \sqrt{\rho(A^H A)},$$
$$\|A\|_\infty = \max_i \sum_j |a_{ij}|.$$

For any matrix $A \in \mathbf{C}^{n\mathbf{x}n}$ the following relations hold

$$\frac{1}{\sqrt{n}}\|A\|_{1,\infty} \leq \|A\|_2 \leq \sqrt{n}\,\|A\|_{1,\infty},$$

where $\|\cdot\|_{1,\infty}$ denotes either $\|\cdot\|_1$ or $\|\cdot\|_\infty$.

B Appendix

Lemma 11 *Let B be the matrix defined in Proposition* **??**. *Then* $\|B^{-1}\| = \binom{2n+1}{n}$.

Proof Clearly, all the matrices L^k are still Toeplitz with positive integer entries, so that $||L^k||_\infty$ coincides with the sum of the elements of the first row of L^k. Moreover, let l_{ij}^k denote the ijth element of L^k; then it is easy to see that, for $k > 1$,

$$l_{1j}^k = \sum_{r=1}^{j} l_{1r}^{k-1} = l_{1,j-1}^k + l_{1j}^{k-1}.$$

We now consider the matrix $P_n = (f_{ij})$ such that $f_{ij} = l_{1j}^i$. For instance

$$P_4 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 3 & 6 & 10 & 15 \\ 1 & 4 & 10 & 20 & 35 \end{pmatrix}.$$

- *rounding error*, ϵ_a, generated by roundoffs in machine operations. It depends on the fact that a machine function $\tilde{\varphi}$ is computed instead of φ. Note that even the machine counterpart of simple operations do not satisfy some simple arithmetic property, e.g. the associative law. Hence ϵ_a depends on the *algorithm* used to compute φ; in other words two different algorithms for the same function can be affected by (strongly) different roundoff errors. We have

$$\epsilon_a = \tilde{\varphi}(\tilde{\mathbf{x}}) - \varphi(\tilde{\mathbf{x}}), \qquad \epsilon_{r,a} = \frac{\tilde{\varphi}(\tilde{\mathbf{x}}) - \varphi(\tilde{\mathbf{x}})}{\varphi(\tilde{\mathbf{x}})}.$$

The two following formulas give the definition of *total absolute error* and *total relative error*, and state the fundamental relationship between the total error and the propagated, mathematical and rounding errors, where $\doteq$ denotes the equality up to a first order analysis.

$$\begin{aligned} \epsilon_{tot} &= \tilde{\varphi}(\tilde{\mathbf{x}}) - f(\mathbf{x}) \doteq \epsilon_p + \epsilon_t + \epsilon_a, \\ \epsilon_{r,tot} &= \frac{\tilde{\varphi}(\tilde{\mathbf{x}}) - f(\mathbf{x})}{f(\mathbf{x})} \doteq \epsilon_{r,p} + \epsilon_{r,t} + \epsilon_{r,a}. \end{aligned}$$

Vector and matrix norms. [**?**] For any fixed vector norm $\|\cdot\|$ we define its associated *operator norm of matrices*, such that for any matrix A it holds

$$\|A\| = \max_{\mathbf{x}\neq 0} \frac{\|A\mathbf{x}\|}{\|\mathbf{x}\|}. \tag{10}$$

For any matrix $A \in \mathbf{C}^{n\times n}$ the following relations hold

$$\rho(A) \leq \|A\| \tag{11}$$

Furthermore we define a function $\mu(A)$, known as the condition number, such that

$$\mu(A) = \begin{cases} \|A\| \cdot \|A^{-1}\| & \text{if } A \text{ is nonsingular,} \\ \infty & \text{otherwise.} \end{cases}$$

In general, we write $\|A\|$ and $\mu(A)$ in relations that hold for any operator matrix norm.

Let us consider the class of vector norms, known as p *norms* or *Hölder norms*:

$$||\mathbf{x}||_p = \left(\sum_{k=1}^{n} |x_k|^p \right)^{\frac{1}{p}}.$$

Analogously, let us consider the associated matrix norms, $\|A\|_p$, and the associated condition number $\mu_p(A)$.

Definition 9 *Given three integers: the* machine base $\beta \geq 2$, *the* integer precision $i \geq 0$, *the* fractional precision $f \geq 0$, *with* $i + f \geq 1$, *we define the set* $\mathcal{F}_{\beta,i,f}$ *of* fixed point numbers *as*

$$\mathcal{F}_{\beta,i,f} = \{f | f = \pm d_1 d_2 \cdots d_i . d_{i+1} d_{i+2} \cdots d_{i+f}, \quad 0 \leq d_j < \beta\} . \qquad \blacksquare$$

In general, when we represent a real number x with either a fixed or a floating point number $\tilde{x}$ we make an error. In order to estimate such an error two quantities are defined, namely the *absolute error* ϵ and the *relative error* ϵ_r, as follows

$$\epsilon = \tilde{x} - x, \qquad\qquad \epsilon_r = \frac{\tilde{x} - x}{x}.$$

It is easy to see the bind that ties absolute error with fixed point numbers. In fact, given $\mathcal{F}_{\beta,i,f}$ and $x \in \mathbf{R}$ such that $\min \mathcal{F}_{\beta,i,f} \leq x \leq \max \mathcal{F}_{\beta,i,f}$, let us associate the real number x to the nearest fixed point number $\tilde{x} \in \mathcal{F}_{\beta,i,f}$. It holds that

$$|\epsilon| = |\tilde{x} - x| \leq \frac{1}{2}\beta^{-f}.$$

Hence the absolute error is independent of the value of x. Analogously, if we associate to the real number $x \neq 0$ the nearest floating point number $\tilde{x} \in \mathcal{F}_{\beta,t,L,U}$, it holds

$$|\epsilon_r| = \left|\frac{\tilde{x} - x}{x}\right| \leq \frac{1}{2}\beta^{1-t}.$$

Since the upper bound to the absolute (relative) error is constant when fixed point (floating point) arithmetic is used, then it is natural to evaluate the error of a computation performed on fixed point numbers using the absolute (relative) error measure.
Errors in function computation. [**?**] The result of the computation of a function $f(\mathbf{x})$ on a machine can be affected by several kind of error. They can be summarized as follows:

- *propagated* or *induced error*, ϵ_p, due to the propagation (and often amplification) of the initial representation error on $\mathbf{x}$. It can be shown that ϵ_p only depends on the function $f(\mathbf{x})$, not on the algorithm used to compute it. In particular it holds that

$$\begin{aligned} \epsilon_p &= f(\tilde{\mathbf{x}}) - f(\mathbf{x}) \doteq \sum_i (\tilde{x}_i - x_i) x_i \frac{\partial f(\mathbf{x})}{\partial x_i}, \quad \text{(abs. prop. error)}, \\ \epsilon_{r,p} &= \frac{f(\tilde{\mathbf{x}}) - f(\mathbf{x})}{f(\mathbf{x})} \doteq \sum_i \frac{\tilde{x}_i - x_i}{x_i} \frac{x_i}{f(\mathbf{x})} \frac{\partial f(\mathbf{x})}{\partial x_i}; \quad \text{(rel. prop. error)} \end{aligned}$$

- *mathematical* or *truncation error*, ϵ_t, that occurs when the function $f(\mathbf{x})$ is replaced by an approximating function $\varphi(\mathbf{x})$. We have

$$\epsilon_t = \varphi(\mathbf{x}) - f(\mathbf{x}), \qquad\qquad \epsilon_{r,t} = \frac{\varphi(\mathbf{x}) - f(\mathbf{x})}{f(\mathbf{x})}.$$

By Cramer's rule we have $x_i = d_i/d_0$, where $d_i = \det(A_i)$, $i = 1, \ldots, n$. We assume that, on input A_i, the oracle for *DET* returns a result $\tilde{d}_i$ affected by a relative error ϵ_i, $i = 1, \ldots, n$. Moreover, we define $\epsilon = \max_i |\epsilon_i|$. We then have

$$\left|\frac{d_i}{d_0} - \frac{\tilde{d}_i}{\tilde{d}_0}\right| = \left|\frac{d_i}{d_0} - \frac{\tilde{d}_i(1+\epsilon_i)}{\tilde{d}_0(1+\epsilon_0)}\right| = \left|\frac{d_i(\epsilon_0 - \epsilon_i)}{d_0(1+\epsilon_0)}\right| \leq \frac{2\epsilon}{(1-\epsilon)}\left|\frac{d_i}{d_0}\right|.$$

If $\epsilon << 1$, we have

$$\frac{\epsilon}{1-\epsilon} = \epsilon(1 + \epsilon + \epsilon^2 + \ldots) \doteq \epsilon,$$

and thus

$$\left|\frac{d_i}{d_0} - \frac{\tilde{d}_i}{\tilde{d}_0}\right| \doteq 2\epsilon\left|\frac{d_i}{d_0}\right|.$$

It is then easy to see that, for any p norm,

$$\|\mathbf{x} - \tilde{\mathbf{x}}\| \doteq 2\epsilon\|\mathbf{x}\|,$$

from which the thesis readily follows. ∎

Last result extends in a trivial way also to the computation of the inverse of a matrix.

Proposition 10 *The algorithm* $A(\text{MATINV}^{\text{DET}})$ *based on Cramer's rule is a T-reduction with respect to the relative error.* ∎

A Appendix

We recall some definitions and properties related to arithmetic and accuracy issues.

Number representation. Since a computing machine is finite it can only represent a finite subset of the real numbers. Of particular interest are the sets of *floating point numbers* and *fixed point numbers*. Once some characterizing parameters are fixed, these two subsets of $\mathbf{R}$ are finite and well suited for hardware representation and manipulation. We have the following definitions.

Definition 8 *Given four integers: the* machine base $\beta \geq 2$, *the* precision $t \geq 1$, *the* underflow limit L, *and the* overflow limit U, *with* $U \geq L$, *we define the set* $\mathcal{F}_{\beta,t,L,U}$ *of* floating point numbers *as*

$$\mathcal{F}_{\beta,t,L,U} = \{0\} \cup \{f | f = \pm 0.d_1 \cdots d_t \times \beta^e,\ 0 \leq d_j < \beta,\ d_1 \neq 0,\ L \leq e \leq U\}.$$

The condition $d_1 \neq 0$ *is called* normalization *and makes the representation unique, but it also creates the need of a separate representation for 0.* ∎

can be proved (see Appendix B) that $\|B^{-1}\|_\infty = \binom{2n+1}{n}$, we have

$$\|B^{-1} - \tilde{B}^{-1}\|_\infty \leq \delta \|B^{-1}\|_\infty = \delta \binom{2n+1}{n} = e.$$

Let us further assume that an error of absolute value e is located in the diagonal entries of the matrix C, that is $\tilde{c}_{jj} = c_{jj} + e = 1 + e$, $j = 1, \ldots, n+1$. It is easy to see that, up to a first order analysis, for the first row of $\tilde{C}_n^{-1}$ we have

$$\tilde{c}_{1j}^{-1} \doteq c_{1j}^{-1} + (-1)^j \binom{2n}{j-1} e = (-1)^n \left[\binom{2n}{j-1} e - \binom{n}{j-1} \right], \tag{8}$$

assuming that the last inversion does not introduce further errors. Since $\binom{2n}{j-1} \geq \binom{n}{j-1}$, for $j = 1, \ldots, n+1$, from (**??**) it follows that:

$$|c_{1j}^{-1} - \tilde{c}_{1j}^{-1}| = \binom{2n}{j-1} e \geq \binom{n}{j-1} e = |c_{1j}^{-1}| e. \tag{9}$$

If we want to compute the characteristic polynomial of I_n, i.e. the first row of C^{-1}, up to a relative error ϵ, we must have:

$$\|C^{-1} - \tilde{C}^{-1}\|_\infty \leq \epsilon \|C^{-1}\|_\infty.$$

Since, from (**??**), it follows that

$$e\|C^{-1}\|_\infty \leq \|C^{-1} - \tilde{C}^{-1}\|_\infty,$$

we have $e\|C^{-1}\|_\infty \leq \epsilon \|C^{-1}\|_\infty$, and hence $e \leq \epsilon$. Finally, from $e = \binom{2n+1}{n}\delta > 2^n \delta$, it follows $\delta < \epsilon 2^{-n}$. This means that, in order to obtain a global relative error ϵ, we must call an inversion oracle with accuracy at least $\epsilon 2^{-n}$, so that the time bound of the oracle, hence of $A(CHARPOLY^{MATINV})$, is $\Theta(\log^2 n)$.

A similar result holds using $\|\cdot\|_1$ or $\|\cdot\|_2$ instead of $\|\cdot\|_\infty$. In fact $\|B^{-1}\|_1 = \|B^{-1}\|_\infty = \binom{2n+1}{n}$ and $\|B^{-1}\|_2 \geq \|B^{-1}\|_\infty/(n+1)$. The last inequality follows from the fact that $\|A\|_2 \geq \|A\|_\infty/\sqrt{n}$, for any $n \times n$ matrix A. Therefore, repeating the proof for $\|\cdot\|_2$ we obtain

$$\delta \leq \binom{2n+1}{n} \frac{\epsilon}{n+1} < \epsilon 2^{-n+1} \qquad \blacksquare$$

Proposition 9 *The algorithm $A(NONSINGEQ^{DET})$ based on Cramer's rule is a T-reduction with respect to the relative error.*

Proof Consider the $n \times n$ linear system $A\mathbf{x} = \mathbf{b}$. Define $A_0 = A$ and, for $i = 1, \ldots, n$,

$$A_i = (A_1 | \ldots | A_{i-1} | \mathbf{b} | A_{i+1} | \ldots | A_n)\,.$$

Therefore, to compute $\mathbf{x}$ up to a relative error ϵ, the inversion oracle must be asked to produce a result with error bounded by $2^{-n}\epsilon$, but this means a time bound $\Theta(\log^2 n)$. ∎

Since for every matrix A, of size n, it holds that $\|A\|_1 \leq \sqrt{n}\|A\|_\infty$, and that $\|A\|_2 \leq \sqrt{n}\|A\|_\infty$, the result stated above is also valid for $\|\cdot\|_1$ and $\|\cdot\|_2$.

The next reduction we study is the algorithm $A(CHARPOLY^{MATINV})$. As usual, we assume that, given an input matrix A, the output $\tilde{A}^{-1}$ produced by the matrix inversion oracle satisfies $\|A^{-1} - \tilde{A}^{-1}\| \leq \epsilon\|A^{-1}\|$, $\epsilon > 0$.

Proposition 8 *With respect to the relative error the time bound of the algorithm* $A(\mathrm{CHARPOLY}^{\mathrm{MATINV}})$ *is* $\Theta(\log^2 n)$ *in the worst case.*

Proof We show an example, based on the adversary argument, for which, in order to compute the characteristic polynomial up to a relative error ϵ, the results returned by the matrix inversion oracle cannot be affected by errors greater than $2^{-n}\epsilon$.

Let us apply the algorithm $A(CHARPOLY^{MATINV})$ of Section 3 to the $n \times n$ identity matrix I_n. The first n calls to the inversion oracle are used to obtain the powers of the submatrices $A_r = I_{n-r+1}$, $r = 1, \ldots, n$. The next call asks for the inverse of the following $(n+1)^2 \times (n+1)^2$ matrix B:

$$B = \begin{pmatrix} I & -L & & & \\ & I & -L & & \\ & & \ddots & \ddots & \\ & & & I & -L \\ & & & & I \end{pmatrix},$$

where L is an $(n+1) \times (n+1)$ upper triangular matrix with all the entries equal to 1. We have

$$B^{-1} = \begin{pmatrix} I & L & L^2 & \ldots & L^n \\ & I & \ddots & \ddots & \vdots \\ & & \ddots & \ddots & L^2 \\ & & & I & L \\ & & & & I \end{pmatrix}.$$

The last step of the algorithm consists now of the inversion of the upper rightmost $(n+1) \times (n+1)$ block of the matrix B^{-1}, say $C = L^n$. The entries of the first row of C^{-1} are, in reverse order, the coefficients of the characteristic polynomial of I_n, that is

$$(\lambda - 1)^n = \lambda^n - \binom{n}{n-1}\lambda^{n-1} + \binom{n}{n-2}\lambda^{n-2} - \cdots + (-1)^n.$$

Let us now assume, without loss of generality, that the first n inversions give the exact results, while the computation of B^{-1} is affected by a relative error δ. Since it

2. Ask the oracle for the inverse of A' and return the first column of A'^{-1}.

The ∞ norm of A' is bounded by the quantity $\|EA\| \leq \|E\|\,\|A\|$, where

$$E = \begin{pmatrix} 1 & & & \\ -b_2 & 1 & & \\ \vdots & & \ddots & \\ -b_n & & & 1 \end{pmatrix},$$

is the matrix which describes the step of Gaussian elimination. It is easy to see that $\|E\| \leq 2$ and $\|E^{-1}\| \leq 2$. Thus $\|A'^{-1} - \tilde{A}'^{-1}\| \leq \epsilon\|A'^{-1}\| \leq 2\epsilon\|A^{-1}\|$, where $\tilde{A}'^{-1}$ is the actual result returned by the *MATINV* oracle. Moreover, it holds that $\|\mathbf{x} - \tilde{\mathbf{x}}\| \leq \|A'^{-1} - \tilde{A}'^{-1}\|$. Overall, we have

$$\|\mathbf{x} - \tilde{\mathbf{x}}\| \leq 2\epsilon\|A^{-1}\| \leq 2\epsilon\mu(A)\|\mathbf{x}\|, \tag{7}$$

where $\mu(A) = \|A\|\,\|A^{-1}\|$ is the condition number of the matrix A. Note that, the relation $\|A^{-1}\| \leq \mu(A)\|\mathbf{x}\|$ holds because $\|\mathbf{b}\| = 1$. In fact, $\|A\|\,\|\mathbf{x}\| \geq \|A\mathbf{x}\| = \|\mathbf{b}\| = 1$, so that

$$\|\mathbf{x}\| \geq \frac{1}{\|A\|} = \frac{\|A^{-1}\|}{\|A\|\,\|A^{-1}\|} = \frac{\|A^{-1}\|}{\mu(A)}.$$

It follows from (**??**) that, for the algorithm to produce an error bounded by ϵ, it is "sufficient" to ask the matrix inversion oracle for an accuracy $\epsilon/(2\mu(A))$. However, unless A is very well-conditioned (i.e. condition number independent of n), the resulting time bound grows asymptotically faster than $\log n$. If A is ill-conditioned, i.e. $\mu(A) = \Theta(2^n)$, the time bound is $\Theta(\log^2 n)$.

From (**??**) we cannot immediately conclude that there will be actual problem instances with the bad behaviour outlined above. However, the following proposition shows that the algorithm $A(NONSINGEQ^{MATINV})$ is not a T-reduction.

Proposition 7 *The algorithm* $A(\mathrm{NONSINGEQ}^{\mathrm{MATINV}})$ *requires* $\Theta(\log^2 n)$ *parallel time in the worst case.*

Proof Let us consider the $n \times n$ linear system $A\mathbf{x} = \mathbf{b}$, with $\mathbf{b} = \mathbf{e}_1$ and $A = L^n$, where L is the $n \times n$ upper triangular matrix whose entries are all 1s. Since $\mathbf{b} = \mathbf{e}_1$, we can avoid step 1 of the algorithm and compute immediately A^{-1}, from which $\mathbf{x} = \mathbf{e}_1$. It can be show that $\|A^{-1}\|_\infty = 2^n - 1$, hence

$$\|A^{-1} - \tilde{A}^{-1}\|_\infty \leq \epsilon\|A^{-1}\|_\infty \approx 2^n\epsilon.$$

Now, the adversary can choose the distribution of errors in a way that at least one element of the first column of $\tilde{A}^{-1}$ is affected by an absolute error $2^n\epsilon$. From the equation $\|\mathbf{x}\| = \|\mathbf{e}_1\| = 1$ we have:

$$\|\mathbf{x} - \tilde{\mathbf{x}}\| = 2^n\epsilon = 2^n\epsilon\|\mathbf{x}\|.$$

we have

$$\begin{aligned}
\|\Delta\mathbf{a}\|_\infty &\leq \frac{1}{n+1}\|F^H\|_\infty \|\Delta\mathbf{y}\|_\infty \leq \epsilon, \\
\|\Delta\mathbf{a}\|_1 &\leq \frac{1}{n+1}\|F^H\|_1 \|\Delta\mathbf{y}\|_1 \leq (n+1)\epsilon, \\
\|\Delta\mathbf{a}\|_2 &\leq \frac{1}{n+1}\|F^H\|_2 \|\Delta\mathbf{y}\|_1 \leq \frac{1}{\sqrt{n+1}}\epsilon.
\end{aligned}$$

Reasoning as in the case of relative errors, we can conclude that the reduction is a T-reduction if either the 2 or ∞ norm is adopted. Using the 1 norm the time bound is $\Theta(\log n \log\log n)$ in the worst case. ∎

What Propositions **??** and **??** say is that, when a higher accuracy in the numerical results is paid according to our cost model, the known reductions do not allow to draw the conclusion that *DET* and *CHARPOLY* have the same parallel asymptotic complexity. However, Proposition **??** proves that, when the relative error measure is adopted (i.e. when the floating point number representation is used), *DET* is at least as general as *CHARPOLY*.

Proposition 6 *If the 1 norm is adopted, then* MATINV *is T-reducible to* NONSINGEQ *with respect to both absolute and relative error.*

Proof Let $\tilde{\mathbf{x}}_i$ denote the result produced by the oracle for *NONSINGEQ* when the input is the pair $(A, \mathbf{e}_i)$, with $\|\mathbf{x}_i - \tilde{\mathbf{x}}_i\|_1 \leq \epsilon\|\mathbf{x}_i\|_1$, $i = 1, \ldots, n$. The vectors $\tilde{\mathbf{x}}_i$ can be obtained by n parallel calls to oracles. Let $\tilde{A}^{-1} = (\tilde{\mathbf{x}}_1|\tilde{\mathbf{x}}_2|\ldots|\tilde{\mathbf{x}}_n)$. Then, we have

$$\begin{aligned}
\|A^{-1} - \tilde{A}^{-1}\|_1 &= \|(\mathbf{x}_1 - \tilde{\mathbf{x}}_1|\mathbf{x}_2 - \tilde{\mathbf{x}}_2|\ldots|\mathbf{x}_n - \tilde{\mathbf{x}}_n)\|_1 \\
&= \max_i \|\mathbf{x}_i - \tilde{\mathbf{x}}_i\|_1 \\
&\leq \epsilon \max_i \|\mathbf{x}_i\|_1 \\
&= \epsilon\|A^{-1}\|_1.
\end{aligned}$$

The error bound is not preserved if the 2 norm or the ∞ norm is used. ∎

It is interesting to see that the known reductions from *NONSINGEQ* to *MATINV* in the opposite direction do not appear to be T-reductions. The trivial algorithm asks the oracle for A^{-1}, then computes $\mathbf{x} = A^{-1}\mathbf{b}$. Since the matrix-vector multiplication may produce a linear loss of accuracy, we consider the following alternative M-reduction.

Algorithm $A(NONSINGEQ^{MATINV})$. Let $A\mathbf{x} = \mathbf{b}$ be the $n \times n$ linear system to be solved, and assume, without loss of generality, that $|b_1| = \max\{|b_i| : i = 1, \ldots, n\} = 1$. Note that the condition $|b_1| = 1$ can be easily satisfied by scaling the coefficients of the first equation.

1. In constant time, perform one step of Gaussian elimination to obtain the equivalent system $A'\mathbf{x} = \mathbf{e}_1$.

Now, the coefficients of

$$\det(\lambda I - A) = a_0 + a_1\lambda + \ldots + a_n\lambda^n$$

can be computed by solving the system

$$\begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega & \omega^2 & \ldots & \omega^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega^n & \omega^{2n} & \ldots & \omega^{n^2} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n. \end{pmatrix} \tag{6}$$

Our goal is to study the error induced in the vector solution $\mathbf{a}$ of (**??**) when the right-hand side vector $\mathbf{y}$ is substituted with $\tilde{\mathbf{y}}$. From the general theory (see, e.g., [**?**]) it is known that, if F denotes the coefficient matrix in (**??**), then

$$\frac{\|\mathbf{a} - \tilde{\mathbf{a}}\|}{\|\mathbf{a}\|} = \frac{\|\Delta\mathbf{a}\|}{\|\mathbf{a}\|} \leq \mu(F)\frac{\|\Delta\mathbf{y}\|}{\|\mathbf{y}\|},$$

where $\Delta\mathbf{y} = \mathbf{y} - \tilde{\mathbf{y}}$ and $\mu(F)$ is the conditioning of F with respect to the chosen norm. It is also known that $\mu_2(F) = 1$, and hence

$$\frac{\|\mathbf{a} - \tilde{\mathbf{a}}\|_2}{\|\mathbf{a}\|_2} \leq \frac{\|\mathbf{y} - \tilde{\mathbf{y}}\|_2}{\|\mathbf{y}\|_2} \leq \epsilon.$$

On the other hand, since $\mu_1(F) = \mu_\infty(F) = n + 1$, the best bound we can obtain for the 1 and ∞ norm is

$$\frac{\|\mathbf{a} - \tilde{\mathbf{a}}\|_{1,\infty}}{\|\mathbf{a}\|_{1,\infty}} \leq (n+1)\frac{\|\mathbf{y} - \tilde{\mathbf{y}}\|_{1,\infty}}{\|\mathbf{y}\|_{1,\infty}} \leq (n+1)\epsilon.$$

We conclude that, if the 2 norm is used, it is sufficient to ask the oracles for accuracy ϵ (which is independent of n by hypothesis). The time bound of the oracles if then $\Theta(\log n)$ and from an algorithm by Preparata [**?**] it follows that the reduction is indeed a T-reduction. On the other hand, if either the 1 norm or the ∞ norm is adopted, the results produced by the oracle must be of a factor $\epsilon/(n+1)$ close to exact results, and this gives a time bound $\Theta(\log n \log\log n)$. This completes the proof of part (1). For what concerns the absolute error, it holds that

$$\|\Delta\mathbf{a}\| \leq \|F^{-1}\|\,\|\Delta\mathbf{y}\| = \frac{1}{n+1}\|F^H\|\,\|\Delta\mathbf{y}\|,$$

where F^H is the conjugate transpose matrix of F. Therefore, using the bounds previously obtained for $\Delta\mathbf{y}$, and the equalities

$$\|F^H\|_2 = \frac{1}{\sqrt{n+1}}\|F^H\|_{1,\infty} = \sqrt{n+1},$$

Proposition 5

1. *The algorithm $A(\text{CHARPOLY}^{\text{DET}})$ is a T-reduction with respect to the relative error measure if the 2 norm is used.*

2. *With respect to the absolute error measure, $A(\text{CHARPOLY}^{\text{DET}})$ is a T-reduction if either the ∞ or the 2 norm is adopted.*

Proof Let $y_k = \det(\omega^k I - A)$, and let $\tilde{y}_k$ be the actual value returned by the oracle for *DET* when the input is the matrix $\omega^k I - A$, $k = 0, \ldots, n$. We assume that

$$\frac{|y_k - \tilde{y}_k|}{|y_k|} \leq \epsilon, \qquad k = 0, \ldots, n. \tag{5}$$

We first prove that

$$\frac{\|\mathbf{y} - \tilde{\mathbf{y}}\|}{\|\mathbf{y}\|} \leq \epsilon,$$

where $\tilde{\mathbf{y}} = [\tilde{y}_0, \ldots, \tilde{y}_n]$, and $\|\cdot\|$ is one of the 1, 2, or ∞ norms. For the ∞ norm, we have

$$\frac{\|\mathbf{y} - \tilde{\mathbf{y}}\|}{\|\mathbf{y}\|} = \frac{\max\limits_{0 \leq k \leq n} |y_k - \tilde{y}_k|}{\max\limits_{0 \leq k \leq n} |y_k|} = \frac{|y_i - \tilde{y}_i|}{|y_j|} = \frac{|y_i - \tilde{y}_i|}{|y_i|}\frac{|y_i|}{|y_j|} \leq \frac{|y_i - \tilde{y}_i|}{|y_i|} \leq \epsilon.$$

For what concerns the 2 norm, it follows from (**??**) that

$$\frac{(y_k - \tilde{y}_k)^2}{y_k^2} \leq \epsilon^2, \qquad k = 0, \ldots, n.$$

Hence, assuming $y_k \neq 1$, $k = 0, \ldots, n$,

$$\begin{aligned}
\frac{\|\mathbf{y} - \tilde{\mathbf{y}}\|_2}{\|\mathbf{y}\|_2} &= \left((1/\sum_{k=0}^{n} y_k^2) \sum_{k=0}^{n} (y_k - \tilde{y}_k)^2 \right)^{\frac{1}{2}} \\
&= \left((1/\sum_{k=0}^{n} y_k^2) \sum_{k=0}^{n} \frac{(y_k - \tilde{y}_k)^2}{y_k^2} y_k^2 \right)^{\frac{1}{2}} \\
&\leq \left((1/\sum_{k=0}^{n} y_k^2) \sum_{k=0}^{n} \epsilon^2 y_k^2 \right)^{\frac{1}{2}} \\
&= \epsilon.
\end{aligned}$$

A similar proof can be exhibited for the 1 norm.
For the absolute error measure, the following relations can be easily proved:

$$\begin{aligned}
&\|\mathbf{y} - \tilde{\mathbf{y}}\|_1 \leq (n+1)\epsilon, \\
&\|\mathbf{y} - \tilde{\mathbf{y}}\|_2 \leq \sqrt{n+1}\epsilon, \\
&\|\mathbf{y} - \tilde{\mathbf{y}}\|_\infty \leq \epsilon.
\end{aligned}$$

a solution affected by the error ϵ. To be a legal reduction, $A(\Pi^{\Pi'})$ must solve Π in parallel time $O(\log n)$ independently of the actual distributions of the errors in the output of Π' (assuming that the latter is not a single number). We then show that an adversary can choose a particular distribution of errors in a way that the cost of $A(\Pi^{\Pi'})$ satisfies (**??**).

Our first result is that even the behaviour of the simpliest reduction considered, i.e. $DET \preceq_T CHARPOLY$, is different under the relative and absolute error measures.

Proposition 4

1. *The algorithm* $A(\mathrm{DET}^{\mathrm{CHARPOLY}})$ *is a T-reduction with respect to the absolute error measure.*

2. *With respect to the relative error,* $A(\mathrm{DET}^{\mathrm{CHARPOLY}})$ *solves the determinant problem in parallel time* $\Theta(\log^2 n)$ *in the worst case.*

Proof Let us consider an $n \times n$ matrix A. The characteristic polynomial of A is defined as $\det(\lambda I - A) = c_n\lambda^n + c_{n-1}\lambda^{n-1} + \cdots + c_0$ and it can be represented by the vector $\mathbf{c} = (c_0, c_1, \ldots, c_n)$, with $c_n = 1$.

1. The first part is trivial since, for any p norm (see Appendix A), $\|\mathbf{c} - \tilde{\mathbf{c}}\|_p$ cannot be less than the absolute error on any component.

2. We want to compute an approximation $\widetilde{\det}(A)$ to $\det(A)$ with relative error bounded by a fixed $\epsilon > 0$, i.e.

$$|\det(A) - \widetilde{\det}(A)| \leq \epsilon |\det(A)|, \tag{3}$$

   We use an oracle that computes the characteristic polynomial with relative error bounded by $\delta > 0$. Since $\det(A) = (-1)^n c_0$, the time bound of the algorithm $A(DET^{CHARPOLY})$ is given by the time required to compute $(-1)^n$, which is $O(\log n)$, plus the time taken by the oracle to compute the characteristic polynomial, which is $\Theta(\log n \log\log \frac{1}{\delta})$. The point is that δ cannot be chosen independently of n. In fact, there are matrices for which $|c_0| = 2^{-n}\|\mathbf{c}\|$ (one example is the identity matrix), and hence the adversary might choose a distribution of the errors in $\|\tilde{\mathbf{c}}\|$ in such a way that

$$|c_0 - \tilde{c}_0| = \|\mathbf{c} - \tilde{\mathbf{c}}\| = \delta\|\mathbf{c}\| = \delta 2^n |c_0|. \tag{4}$$

   Combining (**??**) and (**??**) yields the condition $\delta = \epsilon 2^{-n}$. Hence the time bound of the algorithm is $\Theta(\log^2 n)$ in the worst case. ∎

Quite surprisingly, the reduction in the opposite direction, which is certainly more involved, exhibits a better global behaviour with respect to the induced error.

4. $\{DET, CHARPOLY\} \preceq_T \{NONSINGEQ, MATINV\}$. This is by far the most "difficult" reduction [?]. The algorithm is given below; it computes $\det(xI - A)$ using an oracle for *MATINV*.

Algorithm $A(CHARPOLY^{MATINV})$.

1. Compute the powers A_r^2 through A_r^n, for $r = 1, \ldots, n$.

2. Let $a_i^{(r)}$ denote the bottom right element of the matrix A_r^i, and let

$$b_r(x) = 1 + a_1^{(r)}x + \ldots + a_n^{(r)}x^n, \ r = 1, \ldots, n.$$

   Compute

$$b(x) = \prod_{1 \le r \le n} b_r(x) \ (\mathrm{mod}\, x^{n+1}).$$

3. Compute the modular inverse of $b(x)$, i.e. $d(x) = b(x) \ (\mathrm{mod}\, x^{n+1})$.

4. Return the coefficients of $d(x)$ in reverse order, that is $\det(xI - A) = d_0x^n + d_1x^{n-1} + \cdots + d_n$. ∎

It can be proved that the above algorithm needs to call the matrix inversion oracle only. In fact, the following lemma holds.

Lemma 3 ([?])

1. POWERS $\preceq_M$ MATINV,

2. POLYINV $\preceq_M$ MATINV *and* POLYPROD $\preceq$ ITEPROD,

3. ITEPROD $\preceq_M$ POWERS. ∎

4 Approximating circuits and reductions for matrix computations

We now study the reductions discussed in Section 3 assuming that the oracles return accurate but not exact solutions. For any reduction we consider, we either prove that it is a T-reduction (M-reduction), or show that in the worst case its time cost is $f(n)$, with

$$\lim_{n \to \infty} \frac{\log n}{f(n)} = 0. \tag{2}$$

To prove the latter case we proceed as follows. Consider the reduction $A(\Pi^{\Pi'})$ among the matrix problems Π and Π', and assume that the oracle which solves Π' returns

Part (??) of Proposition ?? was first proved in [?]. Various amendments and improvements appeared in [?, ?]. For what concerns part (??), the updated reference is [?].

Definition 6 Det *is the class of problems that T-reduce to* DET. ∎

Definition 7 *A problem* Π *is said to be* T-complete *(*M-complete*) for* Det *if* DET $\preceq_T$ Π *(*DET $\preceq_M$ Π*).* ∎

Proposition 1 establishes that the problems defined above are complete for *Det*. This fact is of great theoretical interest, since, even in absence of full knowledge on the complexity of the problems at hand, it makes a precise statement on their relative difficulty. From a more practical viewpoint, it suggests that a reasonable criterion guiding the design of parallel machines (say special purpose machines for scientific computing) could consist of dedicating hardware/software resources to solve instances of one complete problem, e.g., *DET*, so that these resources can be used in many other computations (see, e.g., [?]).

3 Reductions among linear algebra problems

In this section we recall some of the known reductions among the problems in the class *Det* defined above.

1. *DET* $\equiv_T$ *CHARPOLY*. In one direction the reduction is "almost" an M-reduction. In fact, given an $n \times n$ matrix A, $\det(A)$ is $(-1)^n$ times the coefficient of the constant term of $\det(\lambda I - A)$. Conversely, the coefficients of $\det(\lambda I - A)$ can be computed by evaluating $\det(\alpha_i I - A)$ at $n+1$ distinct points α_i and then interpolating in $O(\log n)$ parallel time (see, e.g., [?]).

2. *NONSINGEQ* $\equiv_T$ *MATINV*. Both reductions are far obvious[3]. To solve a system $A\mathbf{x} = \mathbf{b}$ first invert A then compute, in parallel time $\lceil \log n \rceil + 1$, the product $A^{-1}\mathbf{b}$. To invert A solve, in parallel, the n linear systems $A\mathbf{x}_i = \mathbf{e}_i$, where $\mathbf{e}_i$ is the ith column of the $n \times n$ identity matrix, $i = 1, \ldots n$. $\mathbf{x}_i$ is the ith column of A^{-1}.

3. $\{NONSINGEQ, MATINV\} \preceq_T \{DET, CHARPOLY\}$. The reduction from either problem in the set $\{MATINV, NONSINGEQ\}$ to *DET* (and thus to *CHARPOLY*) are provided by the Cramer's rule.

[3]Actually, using different algorithms, the two problems at hand can be shown to be M-equivalent.

In the presence of approximation, the depth of a circuit can be affected by the required accuracy ϵ[2]. Therefore, the definition of efficient circuit must take into account the value ϵ.

Definition 3 *A family of $\epsilon AACs$ is said to solve a problem* Π *in* polylog parallel time *if, for any $n \geq 1$ and for every instance $I \in \Pi_n$, the depth $D(n)$ of the n-th member of the family is polylogarithmic in both n and* $\log \frac{1}{\epsilon}$. ∎

We now introduce two different notions of reducibility to be used in the rest of the paper. We call them T and M reducibility, since they correspond to the well-known Turing (or Cook) reducibility and many-one (or Karp) reducibility, respectively. As a general observation, a reduction from a problem Π to a problem Π' is an algorithm (e.g. a circuit family) which solves Π using "black boxes" (e.g. oracle gates) for solving instances of Π'. We indicate such algorithms with $A(\Pi^{\Pi'})$. Various types of reductions can then be defined by restricting either the resources available to $A(\Pi^{\Pi'})$ or the number of times the oracle can be invoked.

Definition 4 *A problem* Π T- reduces *to the problem* Π' *(written* $\Pi \preceq_T \Pi'$*) if and only if, for any fixed (i.e. independent of n) $\epsilon > 0$, there exists a family of $\epsilon AACs$ that solves* Π *with accuracy ϵ, using oracle gates for* Π'*, in depth $D(n) = O(\log n)$. If both* $\Pi \preceq_T \Pi'$ *and* $\Pi' \preceq_T \Pi$ *hold, then* Π *and* Π' *are said to be* T-equivalent*, written* $\Pi \equiv_T \Pi'$. ∎

Definition 5 *An algorithm $A(\Pi^{\Pi'})$ is said to be an* M-reduction *from* Π *to* Π' *if and only if it is a T-reduction with the following further restrictions: the oracle for* Π' *is called only once, and the result of $A(\Pi^{\Pi'})$ can be read off the result returned by the oracle.If*Π *M-reduces to* Π' *and* Π' *M-reduces to* Π*, then* Π *and* Π' *are M-equivalent. M-reducibility and M-equivalence are denoted by $\preceq_M$ and $\equiv_M$, resp.* ∎

If infinite precision at finite cost is available, then the following result holds, as already pointed out in the Introduction. Note that, if one takes $\epsilon = 0$ in Definition **??** then the notion of T-reduction coincides with that of NC^1 reduction.

Proposition 2 ([?])

1. DET, CHARPOLY, MATINV, *and* NONSINGEQ *can be solved in parallel time* $O(\log^2(\text{input size}))$ *with a polynomial amount of work (operation count).*

2. *All the four problems,as well as many others, are T-equivalent.* ∎

[2]In principle, since a circuit has fixed structure, this would imply that different circuits must be used even for solving instances with the same size, if the required accuracy is different. However, in our problems the higher cost in circuit depth can always be charged to the oracles gates, which are the only place in which computations are not performed exactly.

Besides the definition given above, there are other characteristics of this model that we want to point out.

1. The circuits of a family are allowed to make use of "oracle gates" to solve instances of a (sub)problem $\Pi'(S', Z', R')$. Each time a solution to an instance I' of Π'_k is needed, an oracle gate can be used with $S'(k)$ inputs and $Z'(k)$ outputs. On input I' the oracle returns an ϵ-approximation of an exact solution to I'. Depth and size of an oracle gate affect depth and size of the circuit. More precisely we assume that the depth D_O of an oracle computing an ϵ-approximation for $I' \in \Pi'_k$ is given by

$$D_O = \Theta(\log S'(k) \log\log(1/\epsilon)).$$

   On the other hand, we assume that the size of oracle gates is always 1. Our choice of the function D_O, and in particular the presence of the factor $\log\log\frac{1}{\epsilon}$, is justified by the fact that $\log\frac{1}{\epsilon}$ is approximately the number of bits required to represent the solution with accuracy ϵ, and by the further assumption that the word length of the oracle is a polynomial in $\log\frac{1}{\epsilon}$.

2. Any gate which is not an oracle gate performs one of the four arithmetic operations over the ground field F, and is supposed not to introduce rounding errors.

3. To any gate g we associate a rational function, namely the function computed by the portion of the circuit above, and including, g. Suppose that the inputs X and Y of an arithmetic gate g are the outputs of two gates computing the rational functions $f(I)$ and $h(I)$, where I is the input instance. Then, for the output Z produced by g it does hold

$$Z = f(I) \; op \; h(I), \tag{1}$$

   where *op* is one of the four arithmetic operations over F. Even though the operation (**??**) is performed exactly (i.e. without introducing rounding errors), nonetheless it is sensitive to the errors by which the inputs X and Y can be affected. It is known from numerical analysis that, if X and Y are affected by relative errors ϵ_X and ϵ_Y, respectively, then the computed value $\tilde{Z}$ is affected (up to a first order analysis) by the error

$$\epsilon_Z = \frac{Z - \tilde{Z}}{Z} \doteq \frac{X}{Z}\frac{\partial Z}{\partial X}\epsilon_X + \frac{Y}{Z}\frac{\partial Z}{\partial Y}\epsilon_Y.$$

   Since the only "operations" that introduce errors are the oracle calls, the analysis that we perform on approximating arithmetic circuits is just directed to the evaluation of the error propagation from the oracle onto the whole algorithm. This gives precise indications on the extent to which a given reduction is numerically viable.

A computational problem Π can be defined as a collection of subsets, indexed by natural numbers, i.e. $\Pi = \Pi_1 \cup \Pi_2 \cup \ldots$ The elements of Π_n are called the *instances* of Π of *size* n. Each instance is defined by specific pieces of data.

Definition 1 *Let* $S, Z : \mathbf{N} \to \mathbf{N}$ *and, for any* $n \geq 1$, *let* $R_n \subseteq F^{S(n)} \times F^{Z(n)}$. *An* instance I *of a problem* Π *over* F *consists of an ordered set of* $S(n)$ *values,* $I = [x_1, \ldots, x_{S(n)}]$, *such that* $x_i \in F$, $i = 1, \ldots, S(n)$. *We are asked to compute an ordered set* $J = [y_1, \ldots, y_{Z(n)}]$ *of elements of* F *satisfying* $R_n(I, J)$. ∎

In the rest of the paper we regard ordered sets of k elements of F as points of F^k. The positive integer n is the size of the instance I (i.e. $I \in \Pi_n$), while the vector $J = [y_1, \ldots, y_{Z(n)}]^T$ is called an *exact solution* to I. We observe that a computational problem Π is completely specified by giving the functions S and Z and the family of relations $R =< R_n >_{n \in \mathbf{N}}$, i.e. $\Pi = \Pi(S, Z, R)$.

In this paper we consider the following matrix problems. *DET* (computing determinants), *MATINV* (computing the inverse matrix), *NONSINGEQ* (solving linear systems), *CHARPOLY* (computing the characteristic polynomial), *POWERS* (computing matrix powers), *ITEPROD* (computing the product of n matrices of order n), *POLYINV* (computing the modular inverse of a polynomial), and *POLYPROD* (computing the product of polynomials).

We now introduce a computation model based on the well-known arithmetic circuits. We assume the reader familiar with the definitions of arithmetic circuit and of arithmetic circuit family (see, e.g., [**?**]). The "natural" complexity measures for arithmetic circuits are *depth* and *size*. The depth D of an arithmetic circuit is the length of the longest directed path from inputs to outputs, while the size T is the overall number of gates. Depth is also called *parallel time*, while size gives the *operation count* (or sequential time).

In what follows, the symbols S, Z, S', and Z' denote polynomially bounded functions from $\mathbf{N}$ to $\mathbf{N}$.

Definition 2 *Let* ϵ *be a fixed positive real, and let* $\Pi = \Pi(S, Z, R)$ *be a computational problem. A family of arithmetic circuits* $\alpha = \{\alpha_n\}_{n \in \mathbf{N}}$, *with* $S(n)$ *inputs and* $Z(n)$ *outputs, is an* ϵ-approximating arithmetic circuit family *(or simply* ϵAAC*) for* Π *if and only if, for any* n *and for any instance* $I \in \Pi_n$, *the output* $\alpha_n(I)$ *produced by the* n*-th circuit* (α_n) *of the family on input* I *satisfies*

$$\|\alpha_n(I) - J\| \leq \gamma\epsilon,$$

where J *is such that* $R(I, J)$ *holds,* $\|\cdot\|$ *is a norm, and*

$$\gamma = \begin{cases} 1 & \text{if absolute error is used,} \\ \|J\| & \text{if relative error is used.} \end{cases}$$

The value $\alpha_n(I)$ *is called an* ϵ-approximation *to* J. ∎

following table shows which reductions continue to hold in an approximate parallel computing environment. Note, however, that while a positive result does imply that the reducibility among the problems at hand is indeed preserved, negative results only concern the reductions that we know of. Our approach is similar, in spirit, to that

$<_\epsilon$	*NonSingEq*	*CharPoly*	*MatInv*	*Det*
NonSingEq	•		no	yes
CharPol		•	no	yes
MatInv	yes		•	yes
Det		no		•

Table 1: Reductions w.r.t. the relative error.

of information-based complexity (see [**?**]), in the sense that in both cases we regard certain information (e.g., the results returned by oracles) as partial and priced, and consider the cost of obtaining it as part of the cost of solving the whole problem. On the other hand, the approach of classical computational complexity assumes that information is both exact and free.

From another viewpoint, our work has can be also compared with the attempt, undertaken by Blum, Shub, and Smale [**?**], of developing a complexity theory over the real numbers (i.e. a complexity theory of continuous problems) which borrows concepts and tools from classical complexity theory.

Finally, we should mention the efforts directed towards the development of a theory of approximation for hard optimization problems (see [**?**]). Though in a different problem area (we address the issue of parallel complexity of "feasible" problems), certain aspects concerning, e.g., the definition of a notion of reduction that preserves the approximation, are at the very heart of both these studies.

The paper is organized as follows. In Section 2 we present the computation model adopted and define some related formal notions. In Section 3 we review the complexity class *Det* and recall the reductions that make certain problems in *Det* equivalent with respect to parallel complexity. Section 4 is the heart of the paper, where we investigate the way in which the presence of errors affect the relative difficulty of problems. Appendix A presents some basic notions from numerical analysis and, particularly, numerical linear algebra; appendix B contains the proof of a Lemma needed in Proposition 8.

Throughout the paper we let F stand for some field with characteristic 0, i.e. a field containing the rational field as a subset. For definiteness, we can think of F as being the real field.

2 Approximate algorithms and complexity

In this section we introduce the formal framework within which we present our results.

1 Introduction

In this paper we speculate on the *arithmetic NC* theory, of fast and feasible (i.e. polylogarithmic time and polynomial work) parallel algorithms for numerical problems, say problems defined over the real numbers [?, ?, ?, ?, ?, ?, ?]. In spite of a few contributions (see, e.g., [?]), this theory appears not to have fully considered, in the statements about the complexity of problems, the influence of such issues as the numerical accuracy. Our goal is just to give both qualitative and quantitative indications about the role played by the required numerical accuracy with respect to the complexity of problems.

In past studies, the relationships between complexity and numerical accuracy have been explored according to at least two viewpoints. From one side, many works on the bit (or boolean) complexity of numerical problems investigated the cost required to solve a given problem as a function of the accuracy. It is now a fact that the numerical accuracy has implications on the arithmetic (word length) to be used, hence on the boolean cost. On the other hand, it is well-known from numerical analysis that the cost of an iterative method depends on specific features of the instances, such as the spectral radius of the iteration matrix (e.g., in the case of a method for solving a linear system), or the absolute value of the derivative of the iteration function (e.g., in the case of a method for solving nonlinear equations). In turn, these features are strictly related to the numerical conditioning of the particular problem instance.

In this paper we explore the issues outlined above according to a different perspective. Our main concern is the relative complexity of problems. As a field of investigation we consider many matrix computations. It has been shown that

Proposition 1 ([?, ?]) *Computing the determinant, the characteristic polynomial, and the inverse of a matrix, and solving a system of linear equations are equivalent problems under NC^1 reduction.* ∎

We maintain that the proofs of equivalence only reflect the combinatorial shape of the problems at hand. If one also considers the numerical aspects that characterize a problem, certain subtleties come into play that affect the complexity in a nontrivial way. In particular, after having introduced a model of computation suitable to take approximation issues into account, we address the following question: "Given that there exists an algorithm which solves the problem A with prescribed accuracy ϵ in parallel time t, does this imply that the problem B, known to be reducible to A whenever infinite precision is available, is also solvable with accuracy ϵ in parallel time t?". In particular, for the NC^1 reductions mentioned above, the parallel time bound is $\Theta(\log n)$[1]. We show that, within our computation model, many known reductions do not hold. For instance, we prove that even the trivial reduction that allows to compute the determinant of a matrix given a black box which returns its characteristic polynomial requires $\Theta(\log^2(\text{input size}))$ time in the worst case. The

[1]All the logarithms in this paper are to the base 2.

Oracle Computations in Parallel Numerical Linear Algebra *

Bruno Codenotti † Mauro Leoncini ‡ Giovanni Resta §

TR-91-060

May 1992

Abstract

In this paper we address the notion of reducibility among linear algebra problems within a parallel computing environment. We prove that, though many such problems have been shown to be NC^1-equivalent, when approximation is taken into account new questions about their relative complexity come up. We introduce a computation model, based on arithmetic circuits, and define a new notion of reducibility that allows to investigate, in an approximate setting, the behaviour of the known reductions. Within this framework, the computation of the determinant appears more general than, e.g., matrix inversion, and it is set as an open problem whether the former can be reduced to the latter under both time and accuracy constraints.

*This work was partially supported by the Italian National Research Council, unde r the " Progetto Finalizzato Sistemi Informatici e Calcolo Parallelo, Sottoproge tto 2". Part of this work was done while the first author was at IEI-CNR, Pisa (Italy).

†International Computer Science Institute, Berkeley, CA 94704. email: brunoc@icsi.berkeley.edu.

‡Dipartimento di Informatica, Pisa (Italy)

§IEI-CNR, Pisa (Italy)