Optimum Parallel Computations with Band Matrices

Victor Pan

TR-93-061

September 1993

Optimum Parallel Computations with Band Matrices

Victor Pan†
Department of Mathematics and Computer Science
Lehman College, City University of New York
Bronx, NY 10468
VPAN@ LCVAX.BITNET
and
International Computer Science Institute
1947 Center St.
Berkeley, CA 94704

Isdor Sobze† and Antoine Atinkpahoun†
Department of Computer Science
Graduate School and University Center
City University of New York
33 West 42 Street, New York, NY 10036

Summary. We devise optimum parallel algorithms for solving a band linear system of equations and for computing the determinant of a band matrix, substantially improving the previous record computational complexity estimates of [E]. All our algorithms are in NC or RNC and processor efficient; almost all of them reach the optimum bound on the *potential work* (the product of time and processor bounds). Moreover, these algorithms are in NC^1 or RNC^1 if the bandwidth is a constant.

Keywords. Banded linear systems, banded matrices, determinant, parallel algorithms, parallel computational complexity.

1991 Mathematics Subject Classification. 68Q25, 65Y05, 65F40, 65F05.

† Supported by NSF Grant CRR 9020690 and PSC CUNY Award 662478.

1. Introduction. Solving band linear systems of equations is among the most frequent operations in practice of scientific and engineering computations, and has long been a subject of intensive research (see [GL] for a survey and further bibliography). A $k \times h$ matrix is called *banded* if k and h largely exceed its *bandwidth*, which is defined in terms of its *lower bandwidth* and its *upper bandwidth* as follows: A matrix $A = (a_{ij})$ has lower (respectively, upper) bandwidth $m_- = m_-(A)$ (respectively, $m_+ = m_+(A)$) if m_- (respectively, m_+) is the minimum nonnegative integer such that $a_{ij} = 0$ for $i > j + m_-$ (respectively, $j > i + m_+$). The sum $m = m_+ + m_- = m(A)$ is called the bandwidth of A. If $a_{ij} \neq 0$ as long as $i = j + m_-(A)$ (respectively, $j = i + m_+(A)$), then the matrix A has a lower (respectively, upper) *edge*.

Our main focus in this paper is the computational problem denoted $LIN \cdot SOLVE = L \cdot S(n,m)$: Given a nonsingular $n \times n$ matrix A, with bandwidth m, and a vector $\overrightarrow{b}$ of dimension n, compute the solution $\overrightarrow{x} = A^{-1}\overrightarrow{b}$ to the linear system $A\overrightarrow{x} = \overrightarrow{b}$.

The nonsingularity assumption of $LIN \cdot SOLVE$ can be verified by solving the following two problems (also of independent interest):

$DET = D(n,m)$: Given an $n \times n$ matrix A, with bandwidth m, compute its determinant, $\det A$, or alternatively, if the computation is in the complex field or in one of its subfields,

$|DET|^2 = |D(n,m)|^2$: Given an $n \times n$ matrix A, with bandwidth m, compute $|\det A|^2$.

Frequently, one has to solve several linear systems $A\overrightarrow{x} = \overrightarrow{b}(i)$ with the same nonsingular banded matrix A and several vectors $\overrightarrow{b}(i)$.

We will solve this problem in two stages:

1°. $PREPROCESS = P(n,m)$: given a nonsingular $n \times n$ matrix A with bandwidth m, compute a set $\mathcal{G}$ of parameters, implicitly defining the inverse matrix A^{-1}(various choices of the set $\mathcal{G}$ will be specified in sections 3 and 7).

2° . $BACK \cdot SOLVE = B(n,m)$: given a nonsingular $n \times n$ matrix A with bandwidth m, a vector $\overrightarrow{b}(i)$, and the set $\mathcal{G}$ of parameters output by $PREPROCESS$ $P(n,m)$, compute the vector $A^{-1}\overrightarrow{b}(i)$.

Besides the solution of $LIN \cdot SOLVE$, DET, $|DET|^2$, $PREPROCESS$ and $BACK \cdot SOLVE$, we will present improved algorithms for two special cases of $LIN \cdot SOLVE$, $PREPROCESS$ and $BACK \cdot SOLVE$: In one case, denoted $LIN \cdot SOLVE^*$, $PREPROCESS^*$ and $BACK \cdot SOLVE^*$, the input matrix A has at least one edge; in another case, denoted $LIN \cdot SOLVE^{**}$, $PREPROCESS^{**}$ and $BACK \cdot SOLVE^{**}$, the input matrix A has both (lower and upper) edges. These requirements hold for a large class of band matrices; in particular, they typically hold for the matrices encountered in applications to $PDE's$ and $ODE's$ (see [A] and [LP]).

We will state all the complexity estimates in the form $(t_A, p_A) = O(t,p)$, which shows that a computational problem $A = A(n,m)$ can be solved in time $t = t(n,m)$ using $p = p(n,m)$ processors, both t and p estimated within constant factors, under the arithmetic $PRAM$ models of parallel computing [KR], [PP].

We will assume a variant of Brent's scheduling principle [KR], [PP], hereafter referred to as B-principle and expressed by the implication: $O(t, sp)$ implies $O(st, p)$ for any $s \geq 1$.

We recall the known estimates (from [BP], [P], [CW], [P91], [P92], [KP], [KP,a]) for the parallel complexity of the computational problems $M(n,q,r)$ and $I(q)$, of $n \times q$ by $q \times r$ matrix multiplication and of $q \times q$ matrix inversion, respectively:

$$(t_{M(n,q,r)}, p_{M(n,q,r)}) = O(\log q, nqrh^{\omega-3}), \tag{1.1}$$

$$(t_{I(q)}, p_{I(q)}) = O\big(\phi(\mathbf{F}), p_{M(q,q,q)}\big), \tag{1.2}$$

provided that $h = \min\{n,q,r\}$, $2 \leq \omega < 2.376$, and that the computation is performed over the field of constants $\mathbf{F}$; $\phi(\mathbf{F}) \leq \log^2 q$ if $\mathbf{F}$ has characteristic 0, $\phi(\mathbf{F}) \leq \log^4 q$ for any $\mathbf{F}$. The bound (1.2) has been obtained in [P91], [P92], [KP], [KP,a], by using Las Vegas randomized algorithms.

The most recent results on the complexity of $LIN \cdot SOLVE$ and DET are due to [E]:

$$(t_{D(n,m)}, p_{D(n,m)}) = O\big(T^*(n), P^*(n,m)\big),$$

$$(t_{L \cdot S(n,m)}, p_{L \cdot S(n,m)}) = O\big(T^*(n), P^*(n,m)\big),$$

where

$$T^*(n) = (\log^3 n)\psi(n) t_{I(n)},$$

$$P^*(n,m) = (\frac{n}{m}) p_{I(m)} \log^{O(1)} n,$$

$$\psi(n) = \begin{cases} 1, & \text{if } |\mathbf{F}| \geq n \ ; \\ (1 + \log\log_{|\mathbf{F}|} n)\log_{|\mathbf{F}|} n, & \text{otherwise.} \end{cases}$$

In this paper, we substantially improve these estimates of [E], to the Las-Vegas randomized bounds

$$O\Big(\Big(\log(\frac{n}{m})\Big) t_{I(m)}, \frac{(\frac{n}{m})}{\log(\frac{n}{m})} p_{I(m)}\Big) \tag{1.3}$$

for DET and for $LIN \cdot SOLVE$ (over any field of constants), which means that our parallel algorithms are in RNC (and even in RNC^1 if m is a constant) and are *optimum* according to the definition of [KR], since their *potential work* (that is, the product of the associated time and processor bounds) does not exceed the record sequential time bounds for DET and $LIN \cdot SOLVE$.

Moreover, if we solve more than an order of m linear systems $A\overrightarrow{x} = \overrightarrow{b}(i)$ with the same $n \times n$ matrix A having bandwidth m, we obtain a further improvement: We solve $PREPROCESS$ at the same randomized Las-Vegas cost (1.3), and we give a solution of $BACK \cdot SOLVE$ at an optimum deterministic cost

$$O\Big((\log n)(\log m), \frac{mn}{(\log n)(\log m)}\Big). \tag{1.4}$$

To obtain these improvements, we applied several techniques distinct from the ones of [E]. In particular, we introduced special preprocessing based on a 2×2 block factorization of the banded input matrix A (thus avoiding a more complicated 3×3 approach of [E]) and on utilizing some auxiliary matrices, such as $I_F(n,m)$, $I_L(n,m)$ defined in section 2, which helped us to reveal and to exploit the sparse structure of the input matrix A and its diagonal blocks. In addition, we achieved the required nonsingularity of the auxiliary block matrices at a substantially lower computational cost than that would have been required by the techniques of [E], since the symmetrization and the randomization technique of [Sc] and [Z] enable us to avoid the auxiliary computation of matrix ranks.

Under some mild additional assumptions, we obtain stronger results:

a) We <u>deterministically</u> obtain (1.3) for $LIN \cdot SOLVE$ and $|DET|^2$ over the fields of characteristic 0.

b) For the problems $LIN \cdot SOLVE^*$ and $LIN \cdot SOLVE^{**}$, we use a completely distinct approach, based on the reduction of these problems to the block bidiagonal linear systems, and we exploit some structural properties of the inverse of a block bidiagonal matrix (compare [BP]). This alternative way enables us to obtain the improved deterministic computational cost bounds:

$$O\Big((\log(\frac{n}{m}))(\log m) + t_{I(m)}, (\frac{n}{m})p_{I(m)} \frac{t_{I(m)}}{t_{I(m)} + (\log(\frac{n}{m}))(\log m)}\Big), \tag{1.5}$$

for $LIN \cdot SOLVE^{**}$, and

$$O\Big((\log(\frac{n}{m}))(\log m) + t_{I(m)}, (\frac{n}{m})p_{I(m)} \log(\frac{n}{m}) \frac{t_{I(m)}}{t_{I(m)} + (\log(\frac{n}{m}))(\log m)}\Big), \tag{1.6}$$

for $LIN \cdot SOLVE^*$. The same estimates (1.5) and (1.6) apply to the problems $LIN \cdot SOLVE^*$ and $LIN \cdot SOLVE$, respectively, when the input matrix is block bidiagonal (which includes the banded triangular case).

c) Furthermore, regarding the solution of several linear systems $A\overrightarrow{x} = \overrightarrow{b}(i)$ with the same $n \times n$ matrix A that has bandwidth m and has both lower and upper edges (respectively, has a lower edge), we show how to solve the problems $PREPROCESS^{**}$, $PREPROCESS^*$, $BACK \cdot SOLVE^{**}$ and $BACK \cdot SOLVE^*$ at a deterministic cost bounded by (1.5), (1.6),

$$O(\log n, \frac{mn}{\log n}), \tag{1.7}$$

and

$$O(\log n, \frac{mn}{\log m}), \tag{1.8}$$

respectively. We also deduce the same deterministic bounds (1.5)-(1.8), for the problems $PREPROCESS^*$, $PREPROCESS$, $BACK \cdot SOLVE^*$ and $BACK \cdot SOLVE$, respectively, in the case where the input matrix A is block bidiagonal (banded triangular).

Let us summarize: By using various new techniques, we substantially improved the algorithms of [E]. In particular, unlike [E], we reached processor optimality and, if m is a constant, the RNC^1 time bound $O(\log n)$. Moreover, we obtained improved and deterministic complexity estimates for $LIN \cdot SOLVE$ when the input matrix is block bidiagonal (banded triangular) and, with a general input, for $LIN \cdot SOLVE^{**}$ and $LIN \cdot SOLVE^{*}$, and we improved the solution of several linear systems with the same band matrix, by using preprocessing algorithms.

We organize our presentation in the following order: After some preliminary results in section 2, we preprocess a strongly nonsingular input matrix A in section 3. We use the results of this preprocessing for the solution of $BACK \cdot SOLVE$ in section 4 and DET in section 5. In section 6, we relax the assumption about the strong nonsingularity of the input matrix A. In sections 7 and 8 we treat the block bidiagonal case and solve $PREPROCESS^{**}$, $BACK \cdot SOLVE^{**}$, $PREPROCESS^{*}$ and $BACK \cdot SOLVE^{*}$. A proof of an auxiliary result from [E] is given in appendix A. Appendix B includes Figures 1 and 2, which show the output of preprocessing for a nonsingular block bidiagonal matrix.

2. Definitions and auxiliary results. Hereafter, 0 denotes the null matrices, I_k the $k \times k$ identity matrix, $I_F(k,p)$ $(k \geq p)$ the $k \times k$ matrix $\begin{pmatrix} I_p & 0 \\ 0 & 0 \end{pmatrix}$, $I_L(k,p)$ $(k \geq p)$ the $k \times k$ matrix $\begin{pmatrix} 0 & 0 \\ 0 & I_p \end{pmatrix}$, $diag(U,V)$ the matrix $\begin{pmatrix} U & 0 \\ 0 & V \end{pmatrix}$, W^T and W^H the transpose and the Hermitian transpose of a matrix W, respectively.

Proposition 2.1. Let $p, r, s, r_1, s_1, r_2, s_2$ be seven positive integers such that $r = r_1 + r_2$, $s = s_1 + s_2$, $p \leq r$, $p \leq s$; B be a $p \times p$ matrix, $W = \begin{pmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{pmatrix}$ be an $r \times s$ matrix, where W_{11} is an $r_1 \times s_1$ matrix; then

$$I_{F(r,r_1)}W = \begin{pmatrix} W_{11} & W_{12} \\ 0 & 0 \end{pmatrix}, \quad I_{L(r,r_2)}W = \begin{pmatrix} 0 & 0 \\ W_{21} & W_{22} \end{pmatrix},$$

$$WI_{F(s,s_1)} = \begin{pmatrix} W_{11} & 0 \\ W_{21} & 0 \end{pmatrix}, \quad WI_{L(s,s_2)} = \begin{pmatrix} 0 & W_{12} \\ 0 & W_{22} \end{pmatrix},$$

$$I_{F(r,r_1)}WI_{F(s,s_1)} = \begin{pmatrix} W_{11} & 0 \\ 0 & 0 \end{pmatrix}, \quad I_{F(r,r_1)}WI_{L(s,s_2)} = \begin{pmatrix} 0 & W_{12} \\ 0 & 0 \end{pmatrix},$$

$$I_{L(r,r_2)}WI_{F(s,s_1)} = \begin{pmatrix} 0 & 0 \\ W_{21} & 0 \end{pmatrix}, \quad I_{L(r,r_2)}WI_{F(s,s_2)} = \begin{pmatrix} 0 & 0 \\ 0 & W_{22} \end{pmatrix}.$$

Moreover, if the matrices $(\,B \quad 0\,)$ and $(\,0 \quad B\,)$ (respectively, $\begin{pmatrix} B \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ B \end{pmatrix}$) have size $p \times r$ (respectively, $s \times p$), then

$$(\,B \quad 0\,)W = (\,B \quad 0\,)I_{F(r,p)}W, \quad (\,0 \quad B\,)W = (\,0 \quad B\,)I_{L(r,p)}W,$$

$$W\begin{pmatrix} B \\ 0 \end{pmatrix} = WI_{F(s,p)}\begin{pmatrix} B \\ 0 \end{pmatrix}, \quad W\begin{pmatrix} 0 \\ B \end{pmatrix} = WI_{L(s,p)}\begin{pmatrix} 0 \\ B \end{pmatrix}.$$

Definition 2.1. A $k \times k$ submatrix of a matrix W formed by rows and columns $i_1, \ldots, i_k$ of W for any k-tuple $(i_1, \ldots, i_k)$ is called *principal*. A matrix is *strongly nonsingular* if all its principal submatrices are nonsingular.

Proposition 2.2 [GL, p.140]. In the field of complex numbers (and in any of its subfields), $W^H W$ and $(W^H W)^{-1}$ are strongly nonsingular matrices for any nonsingular matrix W.

Proposition 2.3. $m(W^H W) \leq 2m(W)$; $\quad m(W_k) \leq m(W)$ for any principal submatrix W_k of W.

3. Preprocessing for the solution of a banded linear system (using a 2×2 block decomposition of the input matrix).

Hereafter, A denotes an $n \times n$ banded matrix having bandwidth m; A_{11}, A_{12}, A_{21} and A_{22} denote the blocks in the 2×2 block representation

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

where the matrices A_{11} and A_{22} have sizes $n_1 \times n_1$ and $n_2 \times n_2$, respectively, with $n_1 = \lceil n/2 \rceil$, $n_2 = n - n_1$. Until section 6, we assume that A is strongly singular, so that A and A^{-1} have the factorizations

$$A = \begin{pmatrix} I_{n_1} & 0 \\ A_{21}A_{11}^{-1} & I_{n_2} \end{pmatrix} \begin{pmatrix} A_{11} & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I_{n_1} & A_{11}^{-1}A_{12} \\ 0 & I_{n_2} \end{pmatrix}, \tag{3.1}$$

$$A^{-1} = \begin{pmatrix} I_{n_1} & -A_{11}^{-1}A_{12} \\ 0 & I_{n_2} \end{pmatrix} \begin{pmatrix} A_{11}^{-1} & 0 \\ 0 & S^{-1} \end{pmatrix} \begin{pmatrix} I_{n_1} & 0 \\ -A_{21}A_{11}^{-1} & I_{n_2} \end{pmatrix}, \tag{3.2}$$

$$S = A_{22}Y, \quad Y = I_{n_2} - A_{22}^{-1}A_{21}A_{11}^{-1}A_{12}, \tag{3.3}$$

where S is called Schur complement of A_{11} in A.

Note that nonsingularity of Y follows from nonsingularity of the matrices A, A_{11} and A_{22}.

Proposition 3.1. If the first m and the last m columns of the matrices A_{11}^{-1} and A_{22}^{-1} are known, then the matrix Y can be computed at a cost bounded by

$$O(t_{M(n/2,m,m)}, p_{M(n/2,m,m)}), \tag{3.4a}$$

and the matrix Y^{-1} can be computed at a cost bounded by

$$O(t_{I(m)} + t_{M(n/2,m,m)}, \frac{t_{I(m)}p_{I(m)} + p_{M(n/2,m,m)}t_{M(n/2,m,m)}}{t_{I(m)} + t_{M(n/2,m,m)}}). \tag{3.4b}$$

Proof. Observe that the matrix A_{12} has the form $\begin{pmatrix} 0 & 0 \\ 0 & L \end{pmatrix}$ and A_{21} has the form $\begin{pmatrix} 0 & U \\ 0 & 0 \end{pmatrix}$, where L and U are $m \times m$ matrices, and apply proposition 2.1 to deduce that

$$A_{12} = I_L(n_1, m)A_{12} = A_{12}I_F(n_2, m) = I_L(n_1, m)A_{12}I_F(n_2, m), \tag{3.5a}$$

$$A_{21} = I_F(n_2, m)A_{21} = A_{21}I_L(n_1, m) = I_F(n_2, m)A_{21}I_L(n_1, m). \tag{3.5b}$$

Then combine (3.5a), (3.5b) and (3.3) to deduce that $Y = I_{n_2} - A_{22}^{-1}I_L(n_1, m)A_{21}I_F(n_2, m)A_{11}^{-1}I_L(n_1, m)A_{12}$. Therefore, the cost of computing Y is bounded by (3.4), and Y is a block triangular matrix of the form

$$Y = \begin{pmatrix} Y_{11} & 0 \\ Y_{21} & I_{n_2 - m} \end{pmatrix}, \tag{3.6}$$

where Y_{11} is an $m \times m$ matrix. It follows that the block Y_{11} is nonsingular and that

$$Y^{-1} = \begin{pmatrix} Y_{11}^{-1} & 0 \\ -Y_{21}Y_{11}^{-1} & I_{n_2-m} \end{pmatrix}. \tag{3.7}$$

This immediately implies the bound (3.5) on the cost of the computation of Y^{-1}. (Note that only the first m columns of Y and of Y^{-1} need be computed). ■

Definition 3.1. We define *preprocessing-2* of matrix A (based on 2×2 block decompositions (3.1), (3.2) and (3.3)) as the computation the triples of matrices $(A^{-1}I_F(n,m)$, $A^{-1}I_L(n,m)$, $Y^{-1})$.

Theorem 3.1. The complexity of preprocessing-2 is bounded by (1.3).

Proof. The first m and the last m columns of A^{-1} are given by the first m and the last m columns of the matrices $A^{-1}I_F(n,m)$ and $A^{-1}I_L(n,m)$, respectively. Expand (3.2) and then apply (3.5a) and (3.5b) to deduce that

$$\begin{aligned} A^{-1}I_F(n,m) &= \begin{pmatrix} A_{11}^{-1}I_F(n_1,m) + A_{11}^{-1}A_{12}Y^{-1}A_{22}^{-1}A_{21}A_{11}^{-1}I_F(n_1,m) & 0 \\ -Y^{-1}A_{22}^{-1}A_{21}A_{11}^{-1}I_F(n_1,m) & 0 \end{pmatrix} \\ &= \begin{pmatrix} A_{11}^{-1}I_F(n_1,m) + A_{11}^{-1}I_L(n_1,m)A_{12}Y^{-1}A_{22}^{-1}I_F(n_2,m)A_{21}A_{11}^{-1}I_F(n_1,m) & 0 \\ -Y^{-1}A_{22}^{-1}I_F(n_2,m)A_{21}A_{11}^{-1}I_F(n_1,m) & 0 \end{pmatrix}; \end{aligned} \tag{3.8}$$

$$\begin{aligned} A^{-1}I_L(n,m) &= \begin{pmatrix} 0 & -A_{11}^{-1}A_{12}Y^{-1}A_{22}^{-1}I_L(n_2,m) \\ 0 & Y^{-1}A_{22}^{-1}I_L(n_2,m) \end{pmatrix} \\ &= \begin{pmatrix} 0 & -A_{11}^{-1}I_L(n_1,m)A_{12}Y^{-1}A_{22}^{-1}I_L(n_2,m) \\ 0 & Y^{-1}A_{22}^{-1}I_L(n_2,m) \end{pmatrix}. \end{aligned} \tag{3.9}$$

Let $t_1(n,m)$ denote the number of parallel steps and let $p_1(n,m)$ denote the number of processors needed for the computation of all the triples $(A^{-1}I_F(n,m), A^{-1}I_L(n,m), Y^{-1})$ and, recursively, $((A_{11}^{-1}I_F(n_1,m)$, $A_{11}^{-1}I_L(n_1,m)$, $Y_{11}^{-1})$, $((A_{22}^{-1}I_F(n_2,m)$, $A_{22}^{-1}I_L(n_2,m)$, $Y_{22}^{-1})$, ... etc. Clearly, this computation includes preprocessing-2. Combining (3.8) and (3.9) with proposition 3.1 leads to the following estimates, where c_1 and c_2 are two constants:

$$t_1(n,m) \le \begin{cases} t_{I(m)}, & n < 2m \\ t_1(n/2,m) + c_1 t_{I(m)} + c_2 t_{M(n/2,m,m)}, & \text{otherwise,} \end{cases}$$

$$p_1(n,m) \le \begin{cases} p_{I(m)}, & n < 2m \\ \max(2p_1(n/2,m),\ p_{I(m)},\ p_{M(n/2,m,m)}), & \text{otherwise.} \end{cases}$$

Recursive application of the latter estimates and the B-principle yields the desired complexity bound (1.3).■

Remark 3.1. The computation of each triple at each step involves only the results obtained at the previous recursive step. Therefore, the current results can override the previous results, so that the storage complexity for the entire computation is bounded by $O(mn)$.

4. Solving a preprocessed band linear system.

Theorem 4.1. If the matrices $A^{-1}_{i_1 i_1 \ldots i_k i_k} I_F(n_{i_1 \ldots i_k}, m)$, $A^{-1}_{i_1 i_1 \ldots i_k i_k} I_L(n_{i_1 \ldots i_k}, m)$, and $Y^{-1}_{i_1 \ldots i_k}$ $(k \geq 0)$ of preprocessing-2 have been precomputed, then the complexity of the subsequent computation of $A^{-1}\overrightarrow{b}$, for a given vector $\overrightarrow{b}$, is bounded by (1.4).

Proof. Let $\overrightarrow{b} = \begin{pmatrix} \overrightarrow{b_1} \\ \overrightarrow{b_2} \end{pmatrix}$ where $\overrightarrow{b_1}$ and $\overrightarrow{b_2}$ are two vectors of dimensions n_1 and n_2, respectively. Expand (3.2), multiply by $\overrightarrow{b}$ and apply (3.5a) and (3.5b) to deduce

$$A^{-1}\overrightarrow{b} = \begin{pmatrix} A_{11}^{-1}\overrightarrow{b_1} + A_{11}^{-1}A_{12}Y^{-1}A_{22}^{-1}A_{21}A_{11}^{-1}\overrightarrow{b_1} - A_{11}^{-1}A_{12}Y^{-1}A_{22}^{-1}\overrightarrow{b_2} \\ -Y^{-1}A_{22}^{-1}A_{21}A_{11}^{-1}\overrightarrow{b_1} + Y^{-1}A_{22}^{-1}\overrightarrow{b_2} \end{pmatrix}$$

$$= \begin{pmatrix} A_{11}^{-1}\overrightarrow{b_1} + A_{11}^{-1}I_L(n_1,m)A_{12}Y^{-1}A_{22}^{-1}I_F(n_2,m)A_{21}A_{11}^{-1}\overrightarrow{b_1} - A_{11}^{-1}I_L(n_1,m)A_{12}Y^{-1}A_{22}^{-1}\overrightarrow{b_2} \\ -Y^{-1}A_{22}^{-1}I_F(n_2,m)A_{21}A_{11}^{-1}\overrightarrow{b_1} + Y^{-1}A_{22}^{-1}\overrightarrow{b_2} \end{pmatrix}.$$

Due to preprocessing , the matrices Y^{-1}, $A_{11}^{-1}I_L(n_1,m)$ and $A_{22}^{-1}I_F(n_2,m)$ are available. Therefore, given the results of preprocessing-2, $A_{11}^{-1}\overrightarrow{b_1}$ and $A_{22}^{-1}\overrightarrow{b_2}$, the computation of $A^{-1}\overrightarrow{b}$ only requires a constant number of multiplications of matrices of sizes at most $m \times q$ by vectors of dimensions at most q, where $q \leq \lceil n/2 \rceil$.

Let $t_2(n,m)$ denote the number of parallel steps and $p_2(n,m)$ the number of processors needed for the computation of $A^{-1}\overrightarrow{b}$, assuming that preprocessing-2 has been performed. The above argument leads us to the following estimates for the pair (t_2, p_2), where c_6 is a constant:

$$t_2(n,m) \leq \begin{cases} t_{M(m,m,1)}, & n \leq 2m \\ t_2(n/2,m) + c_6 t_{M(n/2,m,1)}, & \text{otherwise,} \end{cases}$$

$$p_2(n,m) \leq \begin{cases} p_{M(m,m,1)}, & n \leq 2m \\ \max(2p_2(n/2,m),\ p_{M(n/2,m,1)}), & \text{otherwise.} \end{cases}$$

Recursive application of the latter relations and the B-principle yields the desired complexity bound (1.4).■

Remark 4.1. We can easily exploit Remark 3.1 while computing the solution vector $A^{-1}\overrightarrow{b}$, to keep the storage complexity for this computation bounded by $O(mn)$.

5. Solving *DET*.

Theorem 5.1. Under the assumptions of theorem 4.1, the complexity of *DET* is bounded by (1.3).

Proof. Deduce from (3.1) that

$$\det A = \det A_{11} \det A_{22} \det Y. \tag{5.1}$$

Due to (3.6), $\det Y$=$\det Y_{11}$. Moreover, each of the matrices A_{11} and A_{22} has bandwidth at most m. Therefore, solving *DET* is reduced to two problems of half size each, at the cost of computing the determinant of an $m \times m$ matrix.

Let $t_3(n,m)$ denote the number of parallel steps and $p_3(n,m)$ the number of processors needed for solving DET, given the precomputed matrices. The above argument leads to the following complexity estimates for the pair $(t_3(n,m), p_3(n,m))$:

$$t_3(n,m) \le \begin{cases} t_{I(m)}, & n \le 2m \\ t_3(n/2,m) + t_{I(m)}, & \text{otherwise,} \end{cases}$$

$$p_3(n,m) \le \begin{cases} p_{I(m)}, & n \le m \\ \max(2p_3(n/2,m),\ p_{I(m)}), & \text{otherwise.} \end{cases}$$

Recursive application of the latter estimates and the B-principle yields the complexity bound (1.3). ■

6. How to relax the strong nonsingularity assumption. Shifting from A and DET to $A^H A$ and $|DET|^2$ (over any subfield of the complex field) enables us to relax the assumption about strong nonsingularity of A, due to propositions 2.2, 2.3 and the equations $A^{-1} = (A^H A)^{-1} A^H$, $\det(A^H A) = |\det A|^2$. More precisely, our algorithm of section 5 (for DET) computes, as a by-product, the determinant of all the matrices $A_{i_1 i_1 \ldots i_k i_k}$, $Y_{i_1 i_1 \ldots i_k i_k}$ defined in the proof of theorem 3.1, unless at least one of these matrices is zero. In the latter case, we just output $\det A = 0$ [this equation follows, due to (5.1) and its recursive extension]. Otherwise, if $\det A \ne 0$, all the matrices $A_{i_1 i_1 \ldots i_k i_k}$, $Y_{i_1 i_1 \ldots i_k i_k}$ are nonsingular and our algorithms of section 3 and 4 solve $PREPROCESS$ and $BACK \cdot SOLVE$.

To relax the strong nonsingularity assumption in the case of <u>any</u> field of constants, we apply an alternative argument. Again, we only need to ensure nonsingularity of the matrices A, Y, A_{11}, Y_{11}, A_{22}, Y_{22}, ..., defined in the proof of theorem 3.1. Furthermore, since the nonsingularity of $Y_{i_1 i_1 \ldots i_k i_k}$ follows from the nonsingularity of $A_{i_1 i_1 \ldots i_k i_k}$, $A_{i_1 i_1 \ldots i_k i_k 11}$ and $A_{i_1 i_1 \ldots i_k i_k 22}$, it is sufficient to ensure nonsingularity of $A_{i_1 i_1 \ldots i_k i_k 11}$ and $A_{i_1 i_1 \ldots i_k i_k 22}$, assuming that $A_{i_1 i_1 \ldots i_k i_k}$ is nonsingular. We will next show how to yield nonsingularity of the two largest diagonal blocks (a similar argument applies to all other diagonal blocks). Towards this goal, we will shift from A to PA, with $P = diag\ (I_{n_1 - m_+(A)}, R, I_{n_2 - m_-(A)})$, where R is a random $m \times m$ matrix. Observe that $m(P) \le 2m$; $\det P$ is readily available, $\det A = \det(PA)/\det P$, and $A^{-1} = (PA)^{-1} P$. Moreover, we immediately verify by inspection that the equations (3.5a), (3.5b) and consequently the proofs of proposition 3.1 and theorem 3.1 remain valid after the transition from the matrix A to the matrix PA, as long as $(PA)_{11}$, $(PA)_{22}$, the two principal submatrices of PA (corresponding to the submatrices A_{11} and A_{22}) are nonsingular, and, therefore, PA and $(PA)^{-1}$ have decompositions of the format (3.1), (3.2) and (3.3). We will show this fact in appendix A, by using a certain assignment of the random entries of R (which turns R into some permutation matrix employed in a more complicated argument of [E]). By the standard argument of [Sc], [Z], the nonsingularity of B_{11} and of B_{22} follows (with a high probability) for a random assignment of the values (from a large set) to the entries of R.

7. Preprocessing and the solution of a block bidiagonal linear system.

Notation. In this section, $A = (A_{ij})$, $i, j = 0, 1, \dots, k-1$, denotes a nonsingular bidiagonal $k \times k$ block matrix with $m \times m$ blocks A_{ij}, such that $A_{ij} = 0$ if $j+1 > i$ or $i > j$; $A_i = A_{ii}$, $i = 0, \dots, k-1$; $B_{j+1} = A_{j,j+1}$, $j = 0, \dots, k-2$; $X = A^{-1} = (X_{ij})$, $i, j = 0, \dots, k-1$, where X_{ij} denotes the $m \times m$ blocks of X. Observe, that

$$X_{ij} = 0 \ \text{ if } \ i > j; \quad X_{ij} = (-1)^{i+j} A_i^{-1} \prod_{h=i+1}^{j} B_h A_h^{-1} \ \text{ if } \ i \le j. \tag{7.1}$$

Preprocessing of a $k \times k$ block bidiagonal matrix A (with nonsingular superdiagonal blocks). If $X_{0,k-1}$ (the northeastern $m \times m$ block of A^{-1}) is nonsingular (which is equivalent to simultaneous nonsingularity of all the blocks $B_1, \dots, B_{k-1}$ of A), then we define preprocessing of A as computing the m first rows of A^{-1}, the m last columns of A^{-1}, and the matrix $X_{0,k}^{-1}$.

Theorem 7.1. The complexity of preprocessing of a $k \times k$ block bidiagonal matrix A with nonsingular block $X_{0,k-1}^{-1}$ is bounded by

$$O((\log m)(\log k) + t_{I(m)},\ k p_{I(m)} \frac{t_{I(m)}}{t_{I(m)} + (\log m)(\log k)}). \tag{7.2}$$

Proof. Perform the following

Algorithm 7.1 (see Figure 1):

1°. Concurrently compute $X_{ii} = A_i^{-1}$, $i = 0, \dots, k-1$, at the cost $O(t_{I(m)}, k p_{I(m)})$.

2°. Combine (7.1) and the parallel prefix algorithm to compute the first block row and the last block column of the block matrix X at the overall cost $O(\log m \log k, \frac{k p_{M(m,m,m)}}{\log k})$.

3°. Compute $X_{0,k-1}^{-1}$, at the cost $O(t_{I(m)}, p_{I(m)})$.

Combining the above estimates and using the B-principle, we deduce the bound (7.2) of theorem 7.1. ■

Solving a preprocessed block bidiagonal linear system (with nonsingular superdiagonal blocks). We next extend Algorithm 7.1 to the evaluation of the solution vector $\overrightarrow{x} = X \overrightarrow{b} = A^{-1} \overrightarrow{b}$ to the linear system $A \overrightarrow{x} = \overrightarrow{b}$, where $\overrightarrow{b} = (\overrightarrow{b}^{(i)})$, $\overrightarrow{b}^{(i)}$ are vectors of dimension m $i = 0, \dots, k-1$.

Algorithm 7.2: Successively compute

1°. $\overrightarrow{v}^{(i)} = X_{0i} \overrightarrow{b}^{(i)}$, $i = 0, \dots, k-1$;

2°. $\overrightarrow{u}^{(i)} = \sum_{h=i}^{k-1} \overrightarrow{v}^{(h)}$, $i = 0, \dots, k-1$ (by applying the parallel prefix algorithm);

3°. $\overrightarrow{x}^{(i)} = X_{i,k-1} X_{0,k-1}^{-1} \overrightarrow{u}^{(i)}$, $i = 0, \dots, k-1$;

4° . Output the vector $\overrightarrow{x} = (\overrightarrow{x}^{(i)})$, $i = 0, \dots, k-1$.

The correctness of this algorithm follows from (7.1). Its parallel complexity is bounded by

$$O(\log(km), \frac{k p_{M(m,m,1)}}{\log(km)}) = O(\log(km), \frac{km^2}{\log(km)}) = O(\log n, \frac{mn}{\log n}), \text{ for } n = km.$$

Remark 7.1. There are various ways to exploit (7.1) in order to effectively compute $X\overrightarrow{b}$ also when only few blocks B_i of A are singular (which implies singularity of $X_{0,k-1}$). For demonstration, assume that only one block B_h is singular. Represent A and $\overrightarrow{b}$ as $A = \begin{pmatrix} C & E \\ 0 & G \end{pmatrix}$, where $E = \begin{pmatrix} 0 & 0 \\ B_h & 0 \end{pmatrix}$, $\overrightarrow{b} = \begin{pmatrix} \overrightarrow{c} \\ \overrightarrow{d} \end{pmatrix}$, so that $X\overrightarrow{b} = \begin{pmatrix} \overrightarrow{u} \\ \overrightarrow{v} \end{pmatrix}$, $\overrightarrow{u} = C^{-1}(\overrightarrow{c} - E\overrightarrow{v})$, $\overrightarrow{v} = G^{-1}\overrightarrow{d}$. This reduces the solution of the linear system $A\overrightarrow{x} = \overrightarrow{b}$ to solving two linear systems with matrices C and G.

Preprocessing of a $k \times k$ block bidiagonal matrix A (general case). Even if all the blocks B_h are singular, we still may extend our solution at the price of increasing by factors of $\log(\frac{n}{m})$ and $\frac{\log n}{\log m}$ the estimated processor bounds for the preprocessing and the subsequent solution of the preprocessed linear system, respectively. Specifically, to extend the preprocessing, we will again apply algorithm 7.1 but will remove its stage 3° and modify its stage 2°. This modification is summarized by the following algorithm, where, for simplicity, we assume that $k = 2^h - 1$ for some integer h:

Algorithm 7.3 (see Figure 2):

1°. Concurrently compute $X_{ii} = A_i^{-1}$, $i = 0, \ldots, k-1$, at the cost $O(t_{I(m)}, kp_{I(m)})$.

2°. Combine (7.1) and the parallel prefix algorithm to compute the blocks X_{ij} of A^{-1}, at the overall cost $O(\log m \log k, kp_{M(m,m,m)})$:

a) for $j = i+1, \ldots, i+q(i)-1$; $i = 1, \ldots, k-1$,

b) for $i = j-1, \ldots, j-q(j)+1$; $j = 1, \ldots, k-1$,

provided that $q(s)$ denotes the largest power of 2 that divides $s+1$.

Solving a preprocessed block bidiagonal linear system (general case). We will extend algorithm 7.3 by the following

Algorithm 7.4. Successively compute

1°. $\overrightarrow{w}^{(i)} = \overrightarrow{b}^{(i)} + A_i \sum_{j=i+1}^{i+q(i)-1} X_{ij}\overrightarrow{b}^{(j)}$, $\quad i = 0, \ldots, k-1$ (by applying the parallel prefix algorithm);

2°. $\overrightarrow{x}^{(i)} = \sum_s^{(i)} X_{i,k-2^s}\overrightarrow{w}^{(k-2^s)}$ where $\sum_s^{(i)}$ denotes the sum in s, s taking the values for which $i_s = 1$ in the following binary representation : $k - i = \sum_{s=0}^{\lfloor \log_2(k-i) \rfloor} i_s 2^s$, $i = 0, \ldots, k-1$.

3°. Output the vector $\overrightarrow{x} = (\overrightarrow{x}^{(i)})$.

The correctness of this algorithm follows from (7.1). Its parallel complexity is bounded by

$O(\log(km), kp_{M(m,m,1)}) = O(\log n, \frac{nm}{\log m})$, for $n = km$.

8. Reduction of $LIN \cdot SOLVE^*$ and $LIN \cdot SOLVE^{**}$ to the block bidiagonal form.

We first note that if m divides n, we may represent any band $n \times n$ upper triangular matrix A as an $\frac{n}{m} \times \frac{n}{m}$ block bidiagonal matrix, with $m \times m$ blocks, where $m = m(A)$ and, moreover, all the superdiagonal blocks are triangular matrices, which are nonsingular if A has an upper edge. Furthermore, we may always shift from

A to $\widehat{A} = diag(A, I_h)$ with h such that m divides $n+h$, $0 \leq h < m$, and then $m(A) = m(\widehat{A})$. It remains to reduce $LIN \cdot SOLVE^*$ to the band triangular case, which we will do next.

Let A, the $n \times n$ input matrix of $LIN \cdot SOLVE^*$, have a lower edge and let p denote $m_-(A) \leq m = m(A)$. Consider the following approach to the solution of the linear system $A\overrightarrow{x} = \overrightarrow{b}$. First define the $(n+p)\times(n+p)$ matrix $B = \begin{pmatrix} V & A \\ 0 & W \end{pmatrix}$, where $V = \begin{pmatrix} I_p \\ 0 \end{pmatrix}$, $W = (0 \quad I_p)$, and consider the auxiliary linear system

$$B\begin{pmatrix} \overrightarrow{0} \\ \overrightarrow{x} \end{pmatrix} = \begin{pmatrix} \overrightarrow{b} \\ \overrightarrow{z} \end{pmatrix}. \tag{8.1}$$

Note that B is a nonsingular triangular matrix [with a bandwidth $m = m(A) \geq p$]. Denote $B^{-1} = \begin{pmatrix} G & H \\ K & L \end{pmatrix}$, $\begin{pmatrix} \overrightarrow{0} \\ \overrightarrow{x} \end{pmatrix} = \begin{pmatrix} G & H \\ K & L \end{pmatrix}\begin{pmatrix} \overrightarrow{b} \\ \overrightarrow{z} \end{pmatrix}$ where H is a $p \times p$ matrix. [Since $m \geq p$, we may apply algorithm 7.1 and evaluate G and H at the cost bounded by (7.2) for $k = \frac{n}{m}$]. Furthermore, from the nonsingularity of A and the factorization

$$B = \begin{pmatrix} V & A \\ 0 & W \end{pmatrix} = \begin{pmatrix} I & 0 \\ WA^{-1} & I \end{pmatrix}\begin{pmatrix} A & 0 \\ 0 & -WA^{-1}V \end{pmatrix}\begin{pmatrix} A^{-1}V & I \\ I & 0 \end{pmatrix}, H = -(WA^{-1}V)^{-1},$$

we deduce the nonsingularity of H.

Now we compute the vector $\overrightarrow{z} = -H^{-1}G\overrightarrow{b}$ and substitute it into (8.1). This computation, to which we will refer as to **algorithm 8.1**, turns (8.1) into the solution of a banded triangular system, $A\overrightarrow{x} = \overrightarrow{b}$. The computational cost of this transition, that is, of computing the vector $\overrightarrow{z}$, is bounded by

$$O(t_{I(p)} + \log n, \quad \frac{p_{I(p)}t_{I(p)} + p_{M(n,p,1)}\log n}{t_{I(p)} + \log n}), \quad p \leq m. \tag{8.2}$$

Combining (8.2) with the estimates of section 7, we arrive at the following result:

Theorem 8.1. A nonsingular $n \times n$ matrix A, having bandwidth $m = m(A)$ and having upper and (respectively, or) lower edges, can be preprocessed at a computational cost bounded by (1.5) [respectively, (1.6)], by means of algorithms 8.1 and 7.1 [respectively, 8.1 and 7.3], and after that, for any fixed vector $\overrightarrow{b}$ of dimension n, the linear system $A\overrightarrow{x} = \overrightarrow{b}$ can be solved, by using algorithm 7.2 (respectively, 7.4) at a computational cost bounded by (1.7) [respectively, (1.8)].

REFERENCES

[A] W.F. Ames, Numerical Methods for Partial Differential Equations, Academic Press, NY, 1977.

[BP] D. Bini and V. Pan, Numerical and Algebraic Computations with Matrices and Polynomials, Birkhauser, Boston, Mass., 1993 (to appear).

[CW] D. Coppersmith, S. Winograd, Matrix Multiplication via Arithmetic Progression, J. of Symbolic Comp., 9,3,251-280, 1990.

[E] W. Eberly, On Efficient Band Matrix Arithmetic, Proc. 33 Ann. IEEE Symp. FOCS, 457-463, 1992.

[GL] G. H. Golub, C.F. Van Loan, Matrix Computations, Johns Hopkins University Press, 1989.

[KP] V.Pan, E. Kaltofen, Processor Efficient Parallel Solution of Linear Systems over an Abstract Field, in Proc. 3 Ann. ACM Symposium on Parallel Algorithms and Architectures, 180-191, 1991.

[KP,a] V. Pan, E. Kaltofen, Processor Efficient Parallel Solution of Linear System II. The Positive Characteristic and Singular Cases, Proc. 33 Ann. IEEE Symp. FOCS, 714-723, 1992.

[KR] R. Karp, V. Ramachandran, Parallel Algorithms for Shared-Memory Machines, in Handbook of Theoretical Computer Science, 869-941, MIT/Elsevier, 1990.

[LP] L. Lapidus, G.F. Pinder, Numerical Solutions of Partial Differential Equations in Science and Engineering, Willey, NY, 1982.

[P] V. Pan, Complexity of Parallel Matrix Computations, Theoret. Comput. Sci., 54, 65-85, 1987.

[PP] V. Pan, F. Preparata, Supereffective Slow-Down of Parallel Computations, Proc. 4 Ann. ACM Symposium on Parallel Algorithms and Architectures, 402-409, 1992.

[P91] V. Pan, "Complexity of Algorithms for Linear Systems of Equations", in Computer Algorithms for Solving Linear Algebraic Equations (The State of the Art), edited by E. Spedicato, NATO ASI Series, Series F: Computer and Systems Sciences, 77-27-56, Springer, Berlin(1991).

[P92] V. Pan, Parametrization of Newton Iteration for Computation with Structured Matrices and Applications, Computers and Mathematics (with Applications), 24,3,61-75 1992.

[Sc] J.T. Schwartz, Fast Probabilistic Algorithms for Verification of Polynomial Identities, J. ACM, 27, 701-717, 1980.

[Z] R.E. Zippel, Probabilistic Algorithms for Sparse Polynomials, Proc. EUROSAM'79, Springer Lecture Notes in Comp. Sci., 72, 216-226, 1979.

Appendix A. How to ensure nonsingularity of the diagonal blocks.

Theorem A.1. Let A be a nonsingular band matrix of size $n \times n$, with bandwidth $m = m(A)$. Let n_1 and n_2 be defined as in section 3. Then there exists a permutation matrix R, of size $m(A) \times m(A)$, such that the matrix $B = PA = diag(I_{n_1-m_+(A)}, R, I_{n_2-m_-(A)})A$ has the block representation of section 3, where B_{11} and B_{22} are nonsingular (B_{ij} denote the blocks of B corresponding to the blocks A_{ij}), whereas the last $n_2 - m$ columns and the first $n_1 - m$ rows of B_{12}, as well as the first $n_1 - m$ columns and the last $n_2 - m$ rows of B_{21}, are all zeros.

Proof. Let A_L denote the $n \times n_1$ matrix $\begin{pmatrix} A_{11} \\ A_{21} \end{pmatrix}$. Without loss of generality, we will assume that $m_+(A) = m_-(A) = k$. Since A is nonsingular, A_L has full (column) rank n_1. The top $n_1 - k$ rows of A_L are linearly independent, since the top $n_1 - k$ rows of A are linearly independent, and since these rows have no nonzero entries outside A_L. The top $n_1 + k$ rows of A_L have full rank, n_1, as well, because A_L has this rank and has no nonzero entries below these rows. It follows that there exists a set of n_1 rows of A_L that includes the first $n_1 - k$ top rows, as well as k of the next $2k = m$ rows, that form a nonsingular $n_1 \times n_1$ matrix. Consequently, there exists a permutation matrix R of size $2k \times 2k$ such that the leading principal $n_1 \times n_1$ submatrix of the matrix $B = \text{diag}(I_{n_1-k}, R, I_{n_2-k})A$ is nonsingular. With no loss of generality, we may assume that all the rows, moved down in the transition from A to B, keep their relative row order. Then $m(B) \leq 2m(A)$, $m(B_{11}) \leq m(A)$, $m(B_{22}) \leq m(A)$. Moreover, the rows that are moved down have no nonzero elements in any column with the number less than $n_1 - m$ and, therefore, the last $n_2 - m$ columns and the first $n_1 - m$ rows of B_{12} will remain filled with zeros as well as the first $n_1 - m$ columns and the last $n_2 - m$ rows of B_{21}. The nonsingularity of B_{22} follows from the nonsingularity of B and of B_{11}. The same argument can be applied to the matrices B_{11}, B_{22} and so on. ■

Appendix B. Figures

Figure 1. Output of preprocessing for a nonsingular 15×15 block bidiagonal matrix A with nonsingular blocks.

```
   0  1  2  3  4  5  6  7  8  9  0  1  1  3  4

0  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■
1     ■                                      ■
2        ■                                   ■
3           ■                                ■
4              ■                             ■
5                 ■                          ■
6                    ■                       ■
7                       ■                    ■
8                          ■                 ■
9                             ■              ■
0                                ■           ■
1                                   ■        ■
2                                      ■     ■
3                                         ■  ■
4                                            ■
```

Figure 2. Output of preprocessing for a nonsingular 31×31 block bidiagonal matrix A (general case).

```
    0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5  6  7  8  9  0

0   ■  ■     ■           ■                       ■
1      ■  ■  ■           ■                       ■
2         ■  ■           ■                       ■
3            ■  ■  ■  ■  ■                       ■
4               ■  ■     ■                       ■
5                  ■  ■  ■                       ■
6                     ■  ■                       ■
7                        ■  ■  ■  ■  ■  ■  ■  ■  ■
8                           ■  ■     ■           ■
9                              ■  ■  ■           ■
0                                 ■  ■           ■
1                                    ■  ■  ■  ■  ■
2                                       ■  ■     ■
3                                          ■  ■  ■
4                                             ■  ■
5                                                ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■  ■
6                                                   ■  ■     ■           ■
7                                                      ■  ■  ■           ■
8                                                         ■  ■           ■
9                                                            ■  ■  ■  ■  ■
0                                                               ■  ■     ■
1                                                                  ■  ■  ■
2                                                                     ■  ■
3                                                                        ■  ■  ■  ■  ■  ■  ■  ■
4                                                                           ■  ■     ■
5                                                                              ■  ■  ■
6                                                                                 ■  ■
7                                                                                    ■  ■  ■  ■
8                                                                                       ■  ■
9                                                                                          ■  ■
0                                                                                             ■
```