Universität Stuttgart
Institut für parallele und verteilte Systeme
Prof. Dr. rer. nat. Miriam Mehl
Dipl.-Ing. Florian Lindner

SS 2016

Parallel Numerics

Exercise 2: Gaussian Elimination

1 Gaussian Elimination

To calculate the solution of a linear equation system

$$Ax = b$$

with a non-singular matrix $A = (a_{i,j}) \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$ the Gauss Elimination algorithm can be applied. The algorithm decomposes the matrix A in a lower triangular matrix L and an upper triangular matrix U: $A = L\,U$ (cp. lecture notes).

A procedure for calculating the upper triangular matrix $U =: A^{(n)}$ from the matrix A can be given as follows:

i) Define $A^{(0)} = (a_{i,j}^{(0)}) := A$

ii) Calculate for $k = 1, \dots, n-1$ the values $l_{i,k}$, $i = k+1, \dots, n$ and the matrices $A^{(k+1)} = (a_{i,j}^{(k+1)})$ with

$$l_{i,k} := a_{i,k}^{(k)} / a_{k,k}^{(k)}$$
$$a_{i,j}^{(k+1)} := \begin{cases} a_{i,j}^{(k)} - l_{i,k} a_{k,j}^{(k)} & \text{for } i \in \{k+1, \dots, n\},\ j \in \{k, \dots, n\} \\ a_{i,j}^{(k)} & \text{otherwise} \end{cases}$$

An implementation of this algorithm could have the following form ("kij-loop-arrangement"):

```
for k = 1 to n − 1
   for i = k + 1 to n
      l_{i,k} := a_{i,k}/a_{k,k}
      for j = k to n
         a_{i,j} := a_{i,j} − l_{i,k} a_{k,j}
```

2 A Simple Example

i) Given is

$$A = \begin{pmatrix} 2 & 2 & 1 \\ 4 & 2 & 3 \\ 2 & 2 & 2 \end{pmatrix}$$

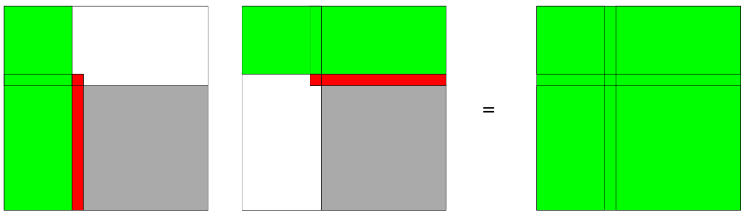
Compute the Gaussian Elimination $A = L \cdot U$ with the algorithm given above.

ii) Compute the Gaussian Elimination using a pivot search (move the entry with the largest absolute value to the pivot position).

iii) Why is pivot search needed? What is the additional computational cost?

3 Parallel Gaussian Elimination

In the lecture, we have defined three steps of the calculation of $L_{2,2}$, $U_{2,2}$, $U_{2,3}$, and $L_{3,2}$ (blocks marked red in the illustration) in the block-wise LU-decomposition



$$\begin{pmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} & U_{13} \\ 0 & U_{22} & U_{23} \\ 0 & 0 & U_{33} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} =$$

$$= \begin{pmatrix} L_{11}U_{11} & L_{11}U_{12} & L_{11}U_{13} \\ L_{21}U_{11} & L_{21}U_{12} + L_{22}U_{22} & L_{21}U_{13} + L_{22}U_{23} \\ L_{31}U_{11} & L_{31}U_{12} + L_{32}U_{22} & * \end{pmatrix}$$

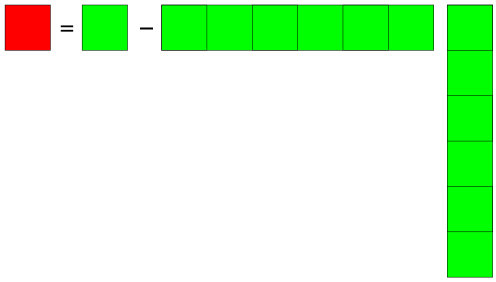
with $A \in \mathbb{R}^{N \times N}$, $L_{22}, U_{22} \in \mathbb{R}^{m \times m}$, $L_{11}, U_{11} \in \mathbb{R}^{km \times km}$. I.e., we are executing the $k+1$st step of the block-wise elimination algorithm, k "stripes" of width m have already been eliminated.

Develop a distributed memory parallel algorithm including computational and communication steps for all substeps of the block-wise elimination step using

- decompositions of all matrices into $m \times m$ blocks,
- BLAS level-3 routines and small LU-decompositions for operations involving these sub-blocks,
- the number of processors and the maximal amount of storage per process prescribed at the end of the substep descriptions below.

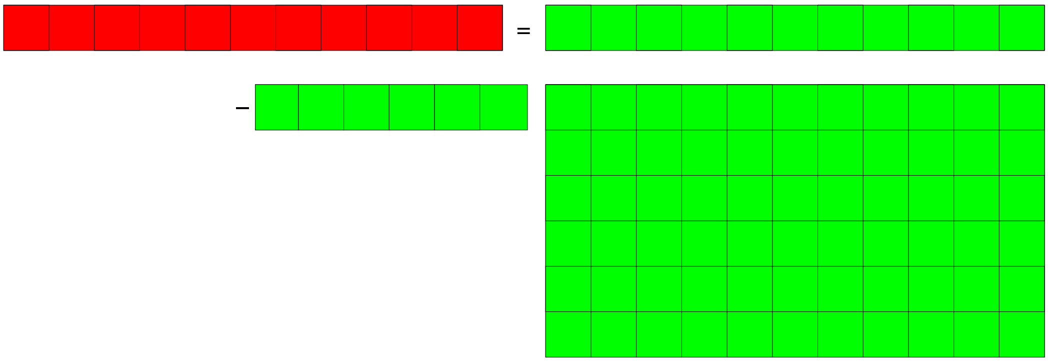
i) Update involved blocks of A using $A \to A - LU$:

$$A_{22} = A_{22} - L_{21}U_{12}$$



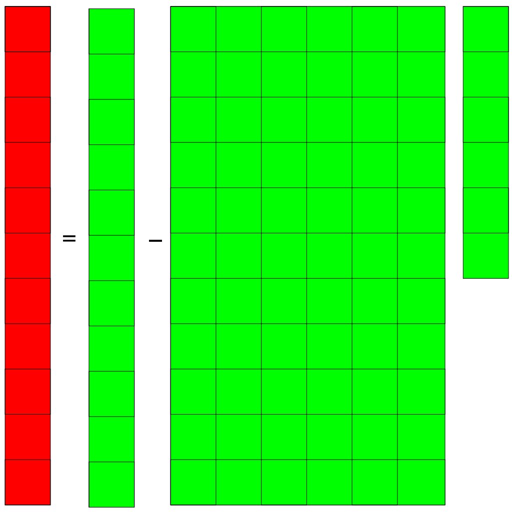
Use k processes and up to three small $m \times m$ matrices per processes plus one additional $m \times m$ matrix in the master processes for your parallel algorithm.

$$A_{23} = A_{23} - L_{21}U_{13}$$



Use $N/m - k$ processes and storage for up to $k+1$ small $m \times m$ matrices per processes plus a minimal number of additional $m \times m$ matrix in the master processes for your parallel algorithm.

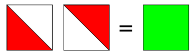
$$A_{32} - L_{31}U_{12}$$



Use $N/m - k$ processes and storage for up to $k+1$ small $m \times m$ matrices per processes plus a minimal number of additional $m \times m$ matrix in the master processes for your parallel algorithm.

ii) Compute the small block-LU-decomposition:

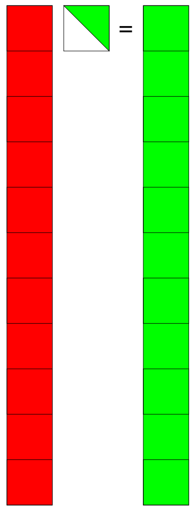
$$L_{22}U_{22} = A_{22}$$



Given the restriction to matric operations with $m \times m$ blocks, this step can not be further parallelized.

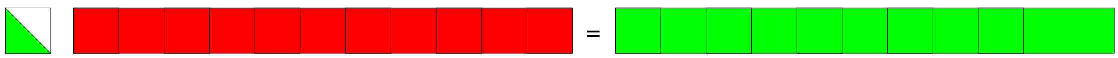
iii) Solve a collection of small independent triangular systems:

$$L_{32}U_{22} = A_{32}$$



Use $N/m - k$ processes and storage for up to three small $m \times m$ matrices per processes for your parallel algorithm.

$$L_{22}U_{23} = A_{23}$$



Use $N/m - k$ processes and storage for up to three small $m \times m$ matrices per processes for your parallel algorithm.