Parallel Numerics, WT 2012/2013

6 Domain Decomposition

Contents

1 Introduction
- 1.1 Computer Science Aspects
- 1.2 Numerical Problems
- 1.3 Graphs
- 1.4 Loop Manipulations

2 Elementary Linear Algebra Problems
- 2.1 BLAS: Basic Linear Algebra Subroutines
- 2.2 Matrix-Vector Operations
- 2.3 Matrix-Matrix-Product

3 Linear Systems of Equations with Dense Matrices
- 3.1 Gaussian Elimination
- 3.2 Parallelization
- 3.3 QR-Decomposition with Householder matrices

4 Sparse Matrices
- 4.1 General Properties, Storage
- 4.2 Sparse Matrices and Graphs
- 4.3 Reordering
- 4.4 Gaussian Elimination for Sparse Matrices

5 Iterative Methods for Sparse Matrices
- 5.1 Stationary Methods
- 5.2 Nonstationary Methods
- 5.3 Preconditioning

6 Domain Decomposition
- 6.1 Overlapping Domain Decomposition
- 6.2 Non-overlapping Domain Decomposition
- 6.3 Schur Complements

Concept of Domain Decomposition

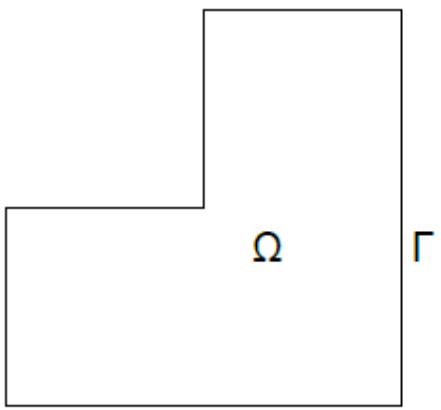
Consider elliptic PDE on region Ω with boundary Γ, e.g. with Dirichlet boundary conditions.

Example:

$$\Delta u = u_{xx} + u_{yy} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y) \in \Omega$$

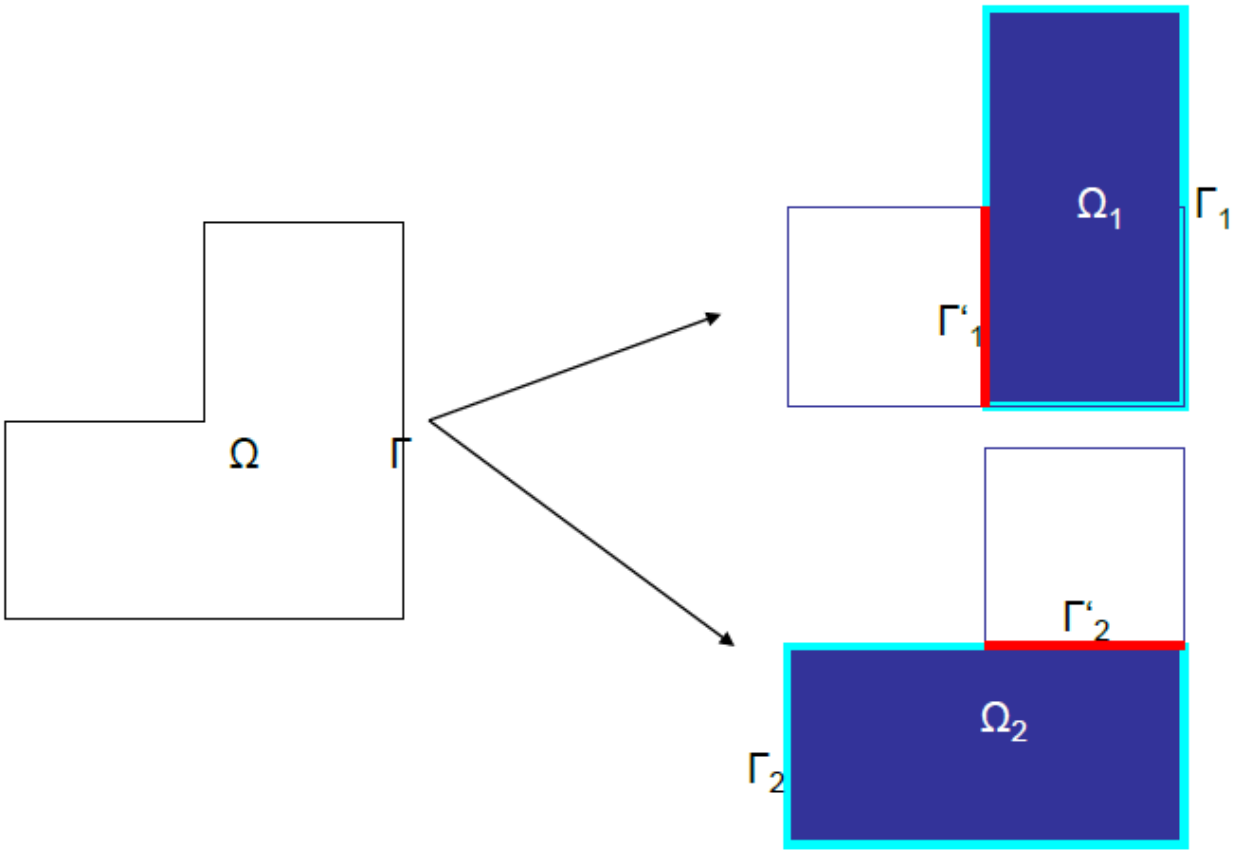
$$u(x, y)|_\Gamma = g(x, y) \in \Gamma$$

Parallel solution?



6.1. Overlapping Domain Decomposition

Partition region Ω in two regions Ω_1 and Ω_2, with new boundaries Γ_1 and Γ_2 which are partially given by old boundary Γ and some unknown parts Γ_1' and Γ_2' :

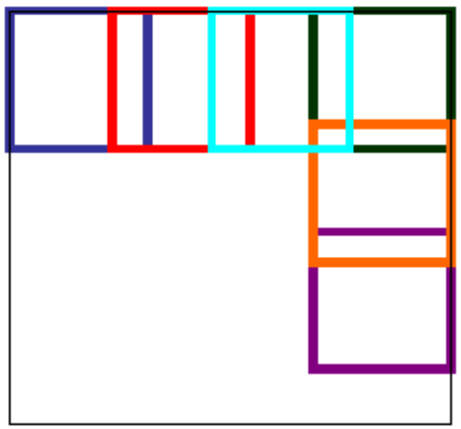


Overlapping Domain Decomposition (cont.)

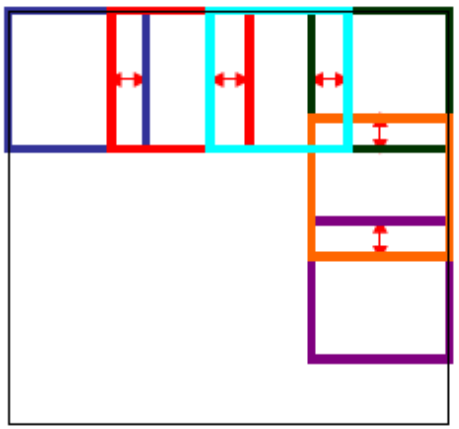
- We discretize and solve given PDE on Ω_1 with boundary Γ_1, but we need the values of $u(x, y)$ on new artificial boundary Γ_1'.
- First, we assume any values for Γ_1', e.g. $u(x, y) = 0$.
- Then we solve the linear system relative to region Ω_1.
- The same can be done in parallel for Ω_2.
- Values of resulting solution of 1) on Γ_2' can be used to compute solution in the next step for region Ω_2 with boundary Γ_2.
- In same way we use the resulting solution of 2) on Γ_1', to compute the solution for region Ω_1 with boundary Γ_1.
- So we generate solutions on partial regions which provide us with approximate values for unknown boundary values of the other partial solution.
- The sequence of solutions converges in each region against solution on Ω.

Overlapping Domain Decompos. (cont. 2)

1. Solve in parallel the PDE on all small subdomains with certain boundary cond.
2. Exchange boundary values



1st step



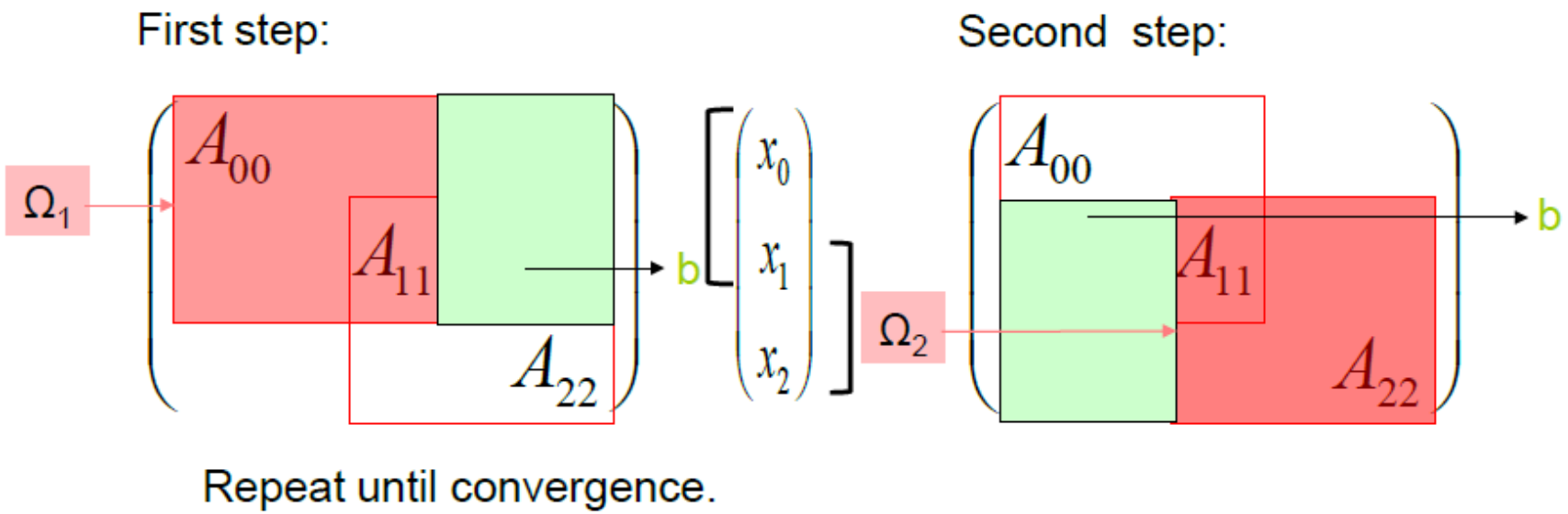
2nd step

3. Repeat until convergence.

For getting better initial values for interior boundary nodes: solve the whole problem on a coarse mesh and interpolate.

Overlapping Domain Decompos. (cont. 3)

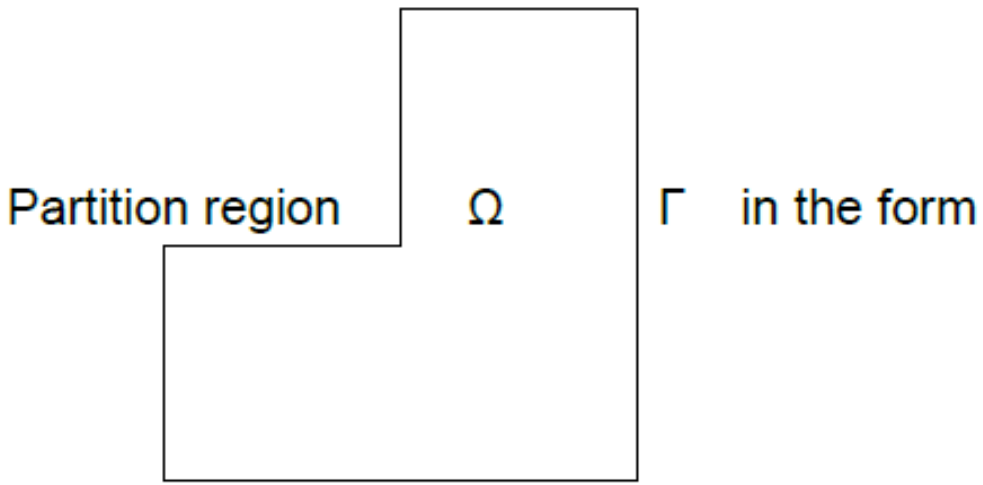
Matrix representation of overlapping domain decomposition:

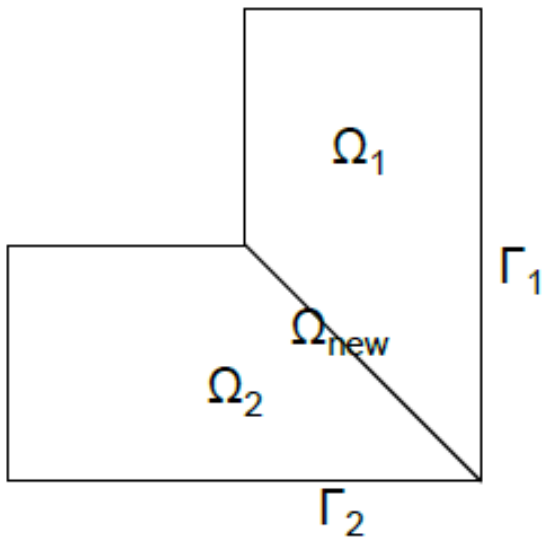


Repeat until convergence.

Green parts are related to the other domain and we asssume to know the related components in the vector *x* of unknowns. They are moved to the right-hand side *b*.

6.2. Non-overlapping Domain Decomposition





Discretization of the original problem with numbering of the unknowns relative to the partitioning given by Ω_1 and Ω_2 leads to a linear system with a matrix in dissection form.

Non-overlapping DD (cont. 2)

Poisson equation on domain Ω

$$\begin{aligned} -\Delta u &= f && \text{in} \quad \Omega, \\ u &= 0 && \text{on} \quad \partial\Omega \end{aligned}$$

is equivalent to

$$\begin{aligned} -\Delta u_1 &= f && \text{in} \quad \Omega_1, \\ u_1 &= 0 && \text{on} \quad \partial\Omega_1 \setminus \Gamma, \end{aligned}$$

$$\begin{aligned} u_1 &= u_2 && \text{on} \quad \Gamma, \\ \frac{\partial u_1}{\partial n_1} &= -\frac{\partial u_2}{\partial n_2} && \text{on} \quad \Gamma, \end{aligned}$$

$$\begin{aligned} -\Delta u_2 &= f && \text{in} \quad \Omega_2, \\ u_2 &= 0 && \text{on} \quad \partial\Omega_2 \setminus \Gamma. \end{aligned}$$

Non-overlapping DD (cont. 3)

In matrix-vector notation $Au = f$ can be written as

$$A = \begin{pmatrix} A_{I,I}^{(1)} & 0 & A_{I,\Gamma}^{(1)} \\ 0 & A_{I,I}^{(2)} & A_{I,\Gamma}^{(2)} \\ A_{\Gamma,I}^{(1)} & A_{\Gamma,I}^{(2)} & A_{\Gamma,\Gamma} \end{pmatrix}, u = \begin{pmatrix} u_I^{(1)} \\ u_I^{(2)} \\ u_\Gamma \end{pmatrix}, f = \begin{pmatrix} f_I^{(1)} \\ f_I^{(2)} \\ f_\Gamma \end{pmatrix}, \qquad (1)$$

where the degrees of freedom are partitioned into those internal to Ω_1, and to Ω_2, and those of the interior of Γ.

On next slide we formulate problem (1) in a more general notation.

Non-overlapping DD (cont. 4)

- A_3 is the so called interface matrix:

$$f = Au = \begin{pmatrix} A_1 & 0 & F_1 \\ 0 & A_2 & F_2 \\ G_1 & G_2 & A_3 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix}$$

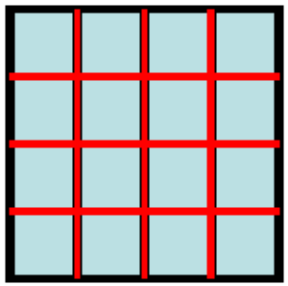
  Better reduce the original problem to two partial subproblems and one interface Schur complement system.

- We can solve $Au = f$ iteratively with PCG and preconditioner:

$$\begin{pmatrix} A_1^{-1} & 0 & 0 \\ 0 & A_2^{-1} & 0 \\ 0 & 0 & M \end{pmatrix}$$

  Here, for M we can use the identity or an approximate inverse for the Schur complement.

Non-overlapping DD (cont. 5)



Leads to 16 block matrices on the diagonal $A_1, \ldots, A_{16}$ and Schur complement S.

$$A = \begin{pmatrix} A_1 & & & & F_1 \\ & \ddots & & & \vdots \\ & & A_{16} & & F_{16} \\ G_1 & \cdots & G_{16} & & A_{17} \end{pmatrix}$$

$$\left(S = A_{17} - G_1 A_1^{-1} F_1 - \ldots - G_{16} A_{16}^{-1} F_{16} \right)$$

- Solve small problems e.g. with multigrid in parallel.
- Overlapping is easy to parallelize, but slow convergence
- Non-overlapping is harder to parallelize, more influence on convergence in S.

6.3. Schur Complements

Write matrix A from (1) in block factorized form $A = LR$

$$L = \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ A^{(1)}_{\Gamma,I} A^{(1)^{-1}}_{I,I} & A^{(2)}_{\Gamma,I} A^{(2)^{-1}}_{I,I} & I \end{pmatrix}, \qquad R = \begin{pmatrix} A^{(1)}_{I,I} & 0 & A^{(1)}_{I,\Gamma} \\ 0 & A^{(2)}_{I,I} & A^{(2)}_{I,\Gamma} \\ 0 & 0 & S \end{pmatrix}$$

leading to the resulting linear system

$$\begin{pmatrix} A^{(1)}_{I,I} & 0 & A^{(1)}_{I,\Gamma} \\ 0 & A^{(2)}_{I,I} & A^{(2)}_{I,\Gamma} \\ 0 & 0 & S \end{pmatrix} u = \begin{pmatrix} f^{(1)}_I \\ f^{(2)}_I \\ b_\Gamma \end{pmatrix},$$

where

$$S = A_{\Gamma,\Gamma} - A^{(1)}_{\Gamma,I} A^{(1)^{-1}}_{I,I} A^{(1)}_{I,\Gamma} - A^{(2)}_{\Gamma,I} A^{(2)^{-1}}_{I,I} A^{(2)}_{I,\Gamma}$$

being the Schur complement relative to the unknowns Γ.

Schur Complements (cont.)

Recalling that

$$f^{(i)} = \begin{pmatrix} f_I^{(i)} \\ f_\Gamma^{(i)} \end{pmatrix}, \qquad A^{(i)} = \begin{pmatrix} A_{I,I}^{(i)} & A_{I,\Gamma}^{(i)} \\ A_{\Gamma,I}^{(i)} & A_{\Gamma,\Gamma}^{(i)} \end{pmatrix}, \qquad i = 1, 2$$

the **local** Schur complement defined by

$$S^{(i)} := A_{\Gamma,\Gamma}^{(i)} - A_{\Gamma,I}^{(i)} {A_{I,I}^{(i)}}^{-1} A_{I,\Gamma}^{(i)},$$

we find the Schur complement system for u_Γ to be

$$S u_\Gamma = b_\Gamma,$$

with

$$\begin{aligned} S &= S^{(1)} + S^{(2)}, \text{ and} \\ b_\Gamma &= (f_\Gamma^{(1)} - A_{\Gamma,I}^{(1)} {A_{I,I}^{(1)}}^{-1} f_I^{(1)}) + (f_\Gamma^{(2)} - A_{\Gamma,I}^{(2)} {A_{I,I}^{(2)}}^{-1} f_I^{(2)}) =: b_\Gamma^{(1)} + b_\Gamma^{(2)}. \end{aligned}$$

Once u_Γ is found, the internal components can be found by

$$u_I^{(i)} = {A_{I,I}^{(i)}}^{-1} (f_I^{(i)} - A_{I,\Gamma}^{(i)} u_\Gamma).$$

Direct Derivation of the Schur Complement

$$\begin{pmatrix} A_1 & 0 & F_1 \\ 0 & A_2 & F_2 \\ G_1 & G_2 & A_3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$\begin{aligned} \Rightarrow \quad & A_1 x_1 + F_1 x_3 = b_1 \\ & A_2 x_2 + F_2 x_3 = b_2 \\ & G_1 x_1 + G_2 x_2 + A_3 x_3 = b_3 \end{aligned}$$

$$\begin{aligned} \Rightarrow \quad & x_1 = A_1^{-1} b_1 - A_1^{-1} F_1 x_3 \quad \text{and} \\ & x_2 = A_2^{-1} b_2 - A_2^{-1} F_2 x_3 \end{aligned}$$

$$\begin{aligned} \Rightarrow \quad & (G_1 A_1^{-1} b_1 - G_1 A_1^{-1} F_1 x_3) + (G_2 A_2^{-1} b_2 - G_2 A_2^{-1} F_2 x_3) + A_3 x_3 = b_3 \\ \Rightarrow \quad & (A_3 - G_1 A_1^{-1} F_1 - G_2 A_2^{-1} F_2) x_3 = b_3 - G_1 A_1^{-1} b_1 - G_2 A_2^{-1} b_2 \\ \Rightarrow \quad & S x_3 = \widetilde{b}_3 \end{aligned}$$

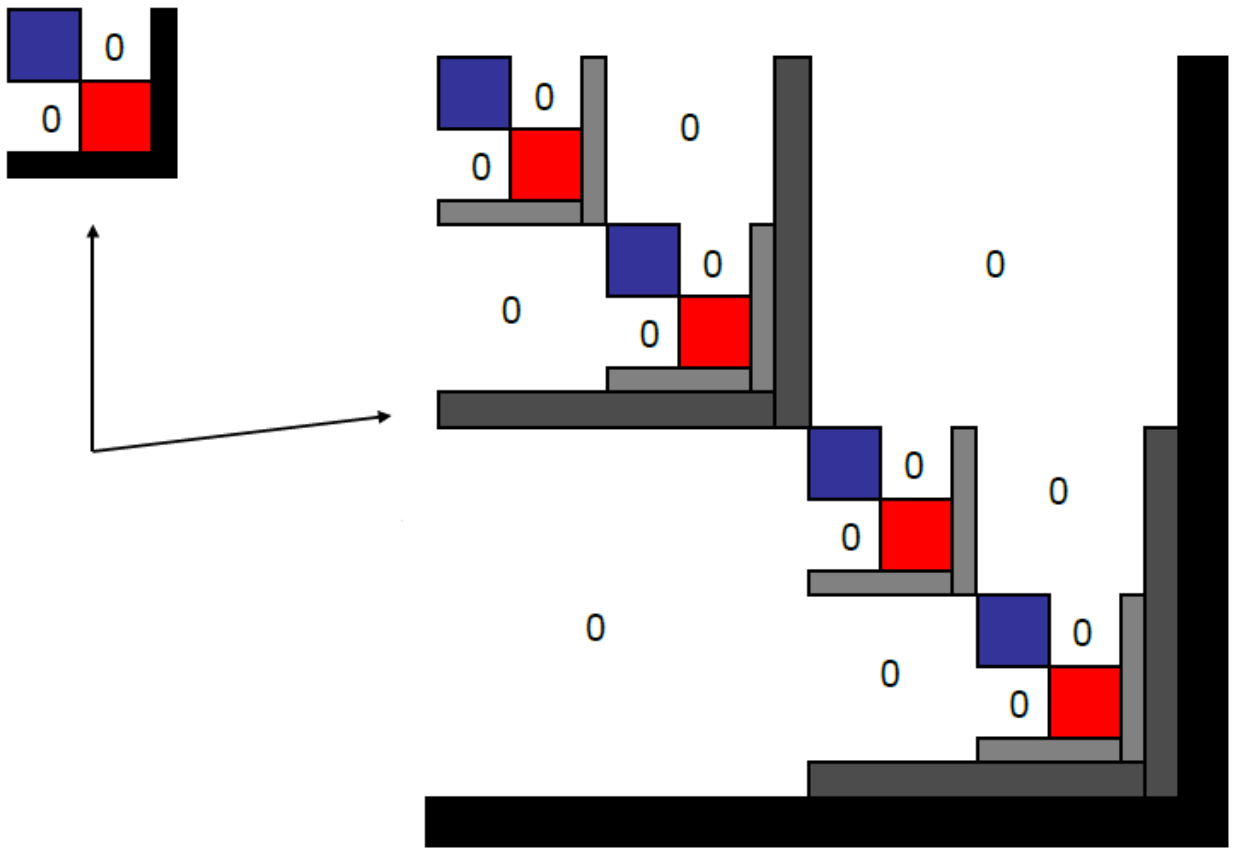
(Parallel) Algorithm to solve $Ax = b$ based on Schur Complement

1. Compute S by using inv(A_1) and inv(A_2)
2. Solve $Sx_3 = \widetilde{b}_3$
3. Compute x_1 and x_2 by using inv(A_1) and inv(A_2)

- The explicit computation of S can be avoided by solving the linear system in S iteratively, e.g. Jacobi, PCG, ...
- Then we need only a part of S and in every iteration step we have to compute $S *$ intermediate vector.
- To achieve fast convergence, a preconditioner (approximation) for S has to be used!
- Precondition Schur complement e.g. with MSPAI

Recursive Form of Non-overlapping DD

$\leftrightarrow$ Nested (recursive) dissection:



Domain Decomposition: Outlook

- Approach can be generalized to
  - non-conforming discretizations (mortar methods/Lagrange multipliers/FETI)
  - time-dependent systems
  - ...
- Literature:
  - A. Toselli, O. Widlund: *Domain Decomposition Methods—Algorithms and Theory*, Springer, 2004
  - A. Quarteroni, A. Valli: *Domain Decomposition Methods for Partial Differential Equations*, Oxford Science Publications, 1999