A PARALLEL AMG FOR OVERLAPPING AND NON-OVERLAPPING DOMAIN DECOMPOSITION*

GUNDOLF HAASE†

Abstract. There exist several approaches for the parallel solving of huge systems of linear equations resulting from the discretization of 2^{nd} order elliptic pdes. On the other hand, there exists a great demand on Algebraic Multigrid solvers (AMG) which have as input only matrix and right hand side or, as a substitute, the appropriate information per element. In this paper we propose a general parallel AMG algorithm using overlapping or non-overlapping data decompositions.

Key words. Algebraic multigrid, parallel algorithms, domain decomposition, iterative solvers.

AMS subject classifications. 65F10, 65N22, 65Y05.

1. Introduction. Without loss of generality we want to solve a second order pde with homogeneous Dirichlet boundary conditions in a domain $\Omega \subset \mathbb{R}^d$, $d = 2, 3$ such that the weak formulation is represented by

$$\text{Find } u \in \mathbb{X}(\Omega) : \quad a(u,v) \;=\; \langle F, v\rangle \quad \forall v \in \mathbb{X}(\Omega) \tag{1.1}$$

with bilinear form $a(u,v) : \mathbb{X} \times \mathbb{X} \mapsto \mathbb{R}$ and duality product $\langle F, v\rangle : \mathbb{X}^* \times \mathbb{X} \mapsto \mathbb{R}$.

A discretization of the domain Ω results in a f.e. mesh τ_h consisting of elements δ^r and nodes $x^{[j]}$ with $j \in \omega$. The f.e. basis $\Phi = \left\{\varphi^{[j]}(x)\,,\; j \in \omega\right\}$ has local support

$$\overline{\text{supp}\varphi^{[i]}} \;=\; \bigcup_{r:\, x^{[i]} \in \overline{\delta}^{(r)}} \overline{\delta}^{(r)} \tag{1.2}$$

and spans a f.e. space $\mathbb{V} = \text{span}\Phi$. Eq. (1.1) changes to

$$\text{Find } u \in \mathbb{V}(\Omega) \subseteq \mathbb{X} : \quad a(u,v) \;=\; \langle F, v\rangle \quad \forall v \in \mathbb{V}(\Omega) \;. \tag{1.3}$$

Now, the f.e. isomorphism

$$u(x) \;=\; \sum_{j\in\omega} v^{[j]} \cdot \varphi^{[j]}(x) \;=\; \Phi \cdot \underline{v} \;, \tag{1.4}$$

guarantees the equivalence of (1.3) to the system of equations

$$K \cdot \underline{u} \;=\; \underline{f} \tag{1.5}$$

with

$$K^{[i,j]} = a(\varphi^{[j]}, \varphi^{[i]}) \tag{1.6}$$

$$f^{[i]} = \left\langle F, \varphi^{[i]} \right\rangle \qquad \forall i, j \in \omega \;. \tag{1.7}$$

The parallelization strategy extends and generalizes the work done in [3, 4, 5, 6].

*This work has been partially supported by the Austrian Science Fund - "Fonds zur Förderung der wissenschaftlichen Forschung" - within the SFB F013 "Numerical and Symbolic Scientific Computing".

†Institute for Analysis and Computational Mathematics, Johannes Kepler University Linz, A-4040 Linz, Austria, ghaase@numa.uni-linz.ac.at.

2. Definitions and examples.

2.1. Sets of subdomains and nodes.

DEFINITION 2.1 (Element Decomposition). *Let us assume that the domain* $\overline{\Omega} = \bigcup_{\delta^{(r)} \in \tau_h} \overline{\delta}^{(r)}$ *can be expressed as union of* p *subdomains* $\overline{\Omega}_s = \bigcup \overline{\delta}^{(r)}$ *such that* $\overline{\Omega} = \bigcup_{s=1}^{p} \overline{\Omega}_s$ *holds. If* $\Omega_s \cap \Omega_q = \emptyset$, $\forall q \neq s$, $s, q = \overline{1, P}$, *then the decomposition of* Ω *is called non-overlapping. Otherwise it is an overlapping decomposition.*

DEFINITION 2.2. *That set of all subdomains, a node* $x^{[i]}$ *belongs to, is denoted by*

$$\sigma^{[i]} \ := \ \{s \ : \ x^{[i]} \in \overline{\Omega}_s\} \ . \tag{2.1}$$

Setting (2.1) is equivalent to the statement

$$x^{[i]} \in \bigcap_{s \in \sigma^{[i]}} \overline{\Omega}_s \ . \tag{2.2}$$

DEFINITION 2.3 ($\omega(\sigma)$). *All nodes with the same set of subdomains* σ *are collected in the index set*

$$\omega(\sigma) \ := \ \{i \in \omega \ : \ \sigma^{[i]} = \sigma\} \ . \tag{2.3}$$

Further we define the index sets

$$\underline{\omega}(\sigma) := \{i \ : \ \sigma^{[i]} \subseteq \sigma\} \tag{2.4}$$

$$\overline{\omega}(\sigma) := \{i \ : \ \sigma \subseteq \sigma^{[i]}\} \ . \tag{2.5}$$

REMARK 2.1. *If we denote by* $\mathcal{P}(B) = \{C \ : \ C \subseteq B\}$ *the power set of a set* B, *then we can write*

$$\underline{\omega}(\sigma) = \bigcup_{\sigma^{[i]} \in \mathcal{P}(\sigma)} \omega(\sigma^{[i]}) \tag{2.6}$$

$$\overline{\omega}(\sigma) = \bigcup_{\sigma \in \mathcal{P}(\sigma^{[i]})} \omega(\sigma^{[i]}) \ . \tag{2.7}$$

REMARK 2.2. *Obviously,* $\omega(\emptyset) = \emptyset$ *and* $\sigma^{[i]} \neq \sigma^{[j]} \ \Leftrightarrow \ \omega(\sigma^{[i]}) \cap \omega(\sigma^{[j]}) = \emptyset$ *hold. The definitions (2.4) and (2.5) imply directly*

$$i \in \underline{\omega}(\sigma) \Leftrightarrow \sigma^{[i]} \subseteq \sigma \qquad \textit{and}$$
$$i \in \overline{\omega}(\sigma) \Leftrightarrow \sigma^{[i]} \supseteq \sigma \ .$$

DEFINITION 2.4. *The index set* $\omega_s := \overline{\omega}(\{s\})$ *contains all nodes of subdomain* Ω_s. *We denote by* $N_s := |\omega_s|$ *the number of nodes therein.*
The definition above implies

$$\sigma^{[i]} \ = \ \{s \ : \ i \in \omega_s\} \ . \tag{2.8}$$

REMARK 2.3. *Obviously, the statements*

$$i \in \omega_s \qquad \Leftrightarrow \qquad s \in \sigma^{[i]} \qquad \Leftrightarrow \qquad \{s\} \subseteq \sigma^{[i]} \tag{2.9}$$

$$j \notin \omega_s \qquad \Leftrightarrow \qquad s \notin \sigma^{[j]} \qquad \Leftrightarrow \qquad \{s\} \not\subseteq \sigma^{[j]} \tag{2.10}$$

are valid.

2.2. Examples. EXAMPLE 2.1 (Non-overlapping Elements). *As a first example, we divide the square into 4 non-overlapping squares and investigate the appropriate σ and $\omega(\sigma)$ sets. Let us represent all equal sets of subdomains by $\sigma^{[i]}$ with some*

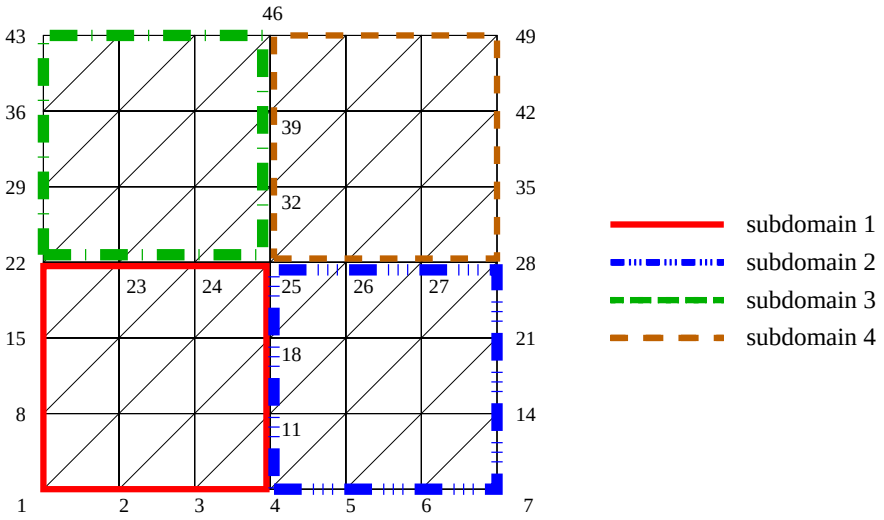

FIGURE 2.1. *Four non-overlapping subdomains with triangulation and row-wise numbering*

fixed node i. Then we have the following adjoint sets :

$$\sigma^{[1]} = \{1\}, \quad \sigma^{[7]} = \{2\}, \quad \sigma^{[43]} = \{3\}, \quad \sigma^{[49]} = \{4\},$$
$$\sigma^{[4]} = \{1,2\}, \quad \sigma^{[22]} = \{1,3\}, \quad \sigma^{[28]} = \{2,4\}, \quad \sigma^{[46]} = \{3,4\},$$
$$\sigma^{[25]} = \{1,2,3,4\}.$$

Some representative index sets are :

$$\begin{aligned}
\omega(\sigma^{[1]}) = \omega(\{1\}) &= \{1,2,3,8,9,10,15,16,17\},\\
\omega(\sigma^{[7]}) = \omega(\{2\}) &= \{5,6,7,12,13,14,19,20,21\},\\
\omega(\sigma^{[4]}) = \omega(\{1,2\}) &= \{4,11,18\},\\
\omega(\sigma^{[22]}) &= \{22,23,24\},\\
\omega(\sigma^{[28]}) &= \{26,27,28\},\\
\omega(\sigma^{[25]}) &= \{25\}.
\end{aligned}$$

Some index sets derived by (2.4) and (2.5) :

$$\begin{aligned}
\underline{\omega}(\sigma^{[1]}) &= \omega(\sigma^{[1]}),\\
\overline{\omega}(\sigma^{[1]}) = \overline{\omega}(\{1\}) &= \omega(\sigma^{[1]}) \cup \omega(\sigma^{[4]}) \cup \omega(\sigma^{[22]}) \cup \omega(\sigma^{[25]})\\
&= \{\overline{1,4}, \overline{8,11}, \overline{15,18}, \overline{22,25}\},\\
\underline{\omega}(\sigma^{[4]}) &= \omega(\sigma^{[4]}) \cup \omega(\{1\}) \cup \omega(\{2\}) &&= \{\overline{1,21}\},\\
\overline{\omega}(\sigma^{[4]}) &= \omega(\sigma^{[4]}) \cup \omega(\sigma^{[25]}) &&= \{4,11,18,25\},\\
\underline{\omega}(\sigma^{[25]}) & &&= \{\overline{1,49}\},\\
\overline{\omega}(\sigma^{[25]}) &= \omega(\sigma^{[25]}) &&= \{25\}.
\end{aligned}$$

EXAMPLE 2.2 (Overlapping Elements). *Here, we divide the square into 4 overlapping squares with an overlap of 4 triangular elements.*

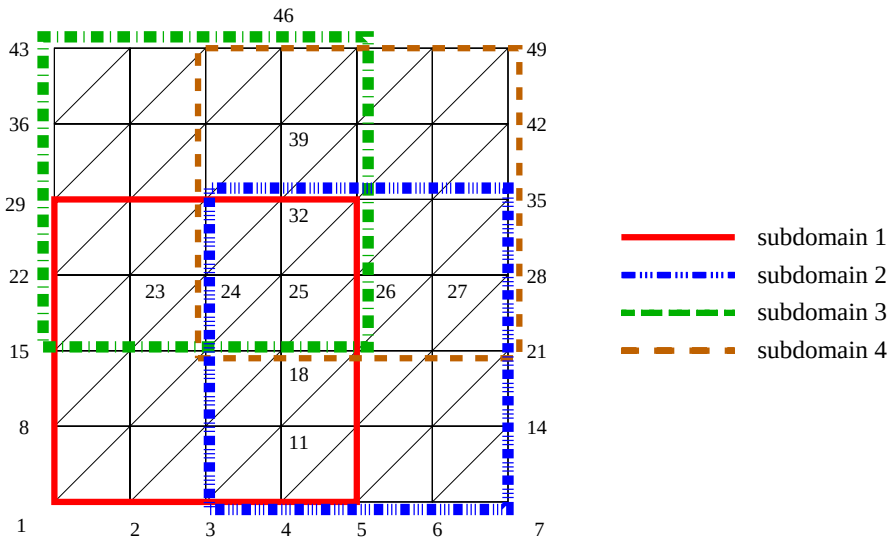

FIGURE 2.2. *Four overlapping subdomains with triangulation and row-wise numbering*

The subdomain sets $\sigma^{[i]}$ *are represented in the same way as in Ex. 2.1 and result in the identical sets. On the other hand, we achieve different index sets, e.g.,*

$$\begin{aligned}
\omega(\sigma^{[1]}) &= \{1,2,8,9\}, & \omega(\sigma^{[7]}) &= \{6,7,13,14\},\\
\omega(\sigma^{[43]}) &= \{36,37,43,44\}, & \omega(\sigma^{[49]}) &= \{41,42,48,49\},\\
\omega(\sigma^{[4]}) &= \{3,4,5,10,11,12\}, & \omega(\sigma^{[22]}) &= \{15,16,22,23,29,30\},\\
\omega(\sigma^{[28]}) &= \{20,21,27,28,34,35\}, & \omega(\sigma^{[46]}) &= \{38,39,40,45,46,47\},\\
\omega(\sigma^{[25]}) &= \{17,18,19,24,25,26,31,32,33\}.
\end{aligned}$$

We achieve the following derived index sets:

$$\begin{aligned}
\underline{\omega}(\sigma^{[1]}) &= \omega(\sigma^{[1]}),\\
\overline{\omega}(\sigma^{[1]}) &= \{\overline{1,5},\overline{8,12},\overline{15,19},\overline{22,25},\overline{29,33}\},\\
\underline{\omega}(\sigma^{[4]}) &= \{\overline{1,14}\},\\
\overline{\omega}(\sigma^{[4]}) &= \{3,4,5,10,11,12,17,18,19,24,25,26,31,32,33\},\\
\underline{\omega}(\sigma^{[25]}) &= \{\overline{1,49}\},\\
\overline{\omega}(\sigma^{[25]}) &= \omega(\sigma^{[25]}).
\end{aligned}$$

2.3. The coincidence matrix.

DEFINITION 2.5 (Coincidence matrix). *The coincidence matrices* A_s *map a global vector* $\underline{u}$ *onto a local vector* $\underline{u}_s$*, i.e.,* $A_s : \mathbb{R}^{\omega} \mapsto \mathbb{R}^{\omega_s}$*. Because of the f.e. isomorphism*

(1.4), this is equivalent to restricting a globally defined function $u(x)$ to the subdomain $\overline{\Omega}_s$ extended by the next layer of elements (support of base functions on $\partial\Omega_s$). The transposed operation maps a local vector back to the global one.

LEMMA 2.6. *By definitions 2.2 and 2.5 we achieve for all $s = \overline{1,P}$ the arithmetic relations :*

$$A_s^T \cdot A_{s\,N\times N} = \operatorname*{diag}_{i\in\omega} \begin{cases} 1 & \text{iff } i \in \omega_s \\ 0 & \text{iff } i \notin \omega_s \end{cases} \tag{2.11a}$$

$$A_s^T \cdot A_{t\,N\times N} = \operatorname*{diag}_{i\in\omega} \begin{cases} 1 & \text{iff } i \in \omega_s \cap \omega_t \\ 0 & \text{iff } i \notin \omega_s \cap \omega_t \end{cases} = \operatorname*{diag}_{i\in\omega} \begin{cases} 1 & \text{iff } \{s,t\} \subseteq \sigma^{[i]} \\ 0 & \text{else} \end{cases} \tag{2.11b}$$

$$A_s \cdot A_{s\,N_s\times N_s}^T = \operatorname*{diag}_{i\in\omega_s}\{1\} \tag{2.11c}$$

$$A_s \cdot A_{t\,N_s\times N_t}^T = \begin{cases} 1 & \text{iff } i=j \in \omega_s \cap \omega_t \\ 0 & \text{else} \end{cases} = \begin{cases} 1 & \text{iff } \{s,t\} \subseteq \sigma^{[i]} \cap \sigma^{[j]} \\ 0 & \text{else} \end{cases} \tag{2.11d}$$

3. Vector and matrix types.

3.1. Vector types. In all domain decomposition methods we have a certain subset of nodes which belong to more than one subdomain, e.g., nodes $22, 25$ in Fig. 2.1 and nodes $21, 22, 24, 25$, in Fig. 2.2.

We store the data of a vector components $u^{[i]}$ in each subdomain Ω_s that component belongs to, i.e., $s \in \sigma^{[i]}$. There are (at least) two opportunities to store those components and at the end that vector.

DEFINITION 3.1 (Accumulated vector). *The restriction of a global vector onto a local one, defined by*

$$\underline{\mathfrak{u}}_s \;:=\; A_s \cdot \underline{\mathfrak{u}} \;, \tag{3.1}$$

results in an accumulated vector $\underline{\mathfrak{u}}_s$ ($\underline{\mathfrak{u}}$). This means, that all subdomains Ω_s, $s \in \sigma^{[i]}$ possess the full value of component $\mathfrak{u}^{[i]}$.

DEFINITION 3.2 (Distributed vector). *If the sum of all local vectors*

$$\underline{\mathfrak{r}} \;=\; \sum_{s=1}^{P} A_s^T \cdot \underline{\mathfrak{r}}_s \;, \tag{3.2}$$

results in the appropriate global vector for all vector components, then we call $\underline{\mathfrak{r}}_s$ ($\underline{\mathfrak{r}}$) a distributed vector. This means, that a subdomain Ω_s, $s \in \sigma^{[i]}$ possesses only a portion of the vector values.

The conversion of a distributed vector $\underline{\mathfrak{v}}$ into an accumulated vector $\underline{\mathfrak{w}}$ can be done by performing the sum in (3.2) followed by restriction (3.1), i.e.,

$$\underline{\mathfrak{w}} \leftarrow \underline{\mathfrak{v}} \qquad : \qquad \underline{\mathfrak{w}}_s := A_s \cdot \underline{\mathfrak{w}} = A_s \cdot \sum_{s=1}^{P} A_s^T \cdot \underline{\mathfrak{v}}_s \;. \tag{3.3}$$

The conversion in the other direction is not unique - we prefer an equal weighted distribution of the accumulated vector. A matrix weighted distribution is also feasible in case of jumping coefficients. The weights are chosen such that re-conversion (3.3) will result in the original vector :

$$\underline{\mathfrak{v}} \leftarrow \underline{\mathfrak{w}} \qquad : \qquad \underline{\mathfrak{v}}_s := (R_s)^{-1} \cdot \underline{\mathfrak{w}}_s \;, \tag{3.4}$$

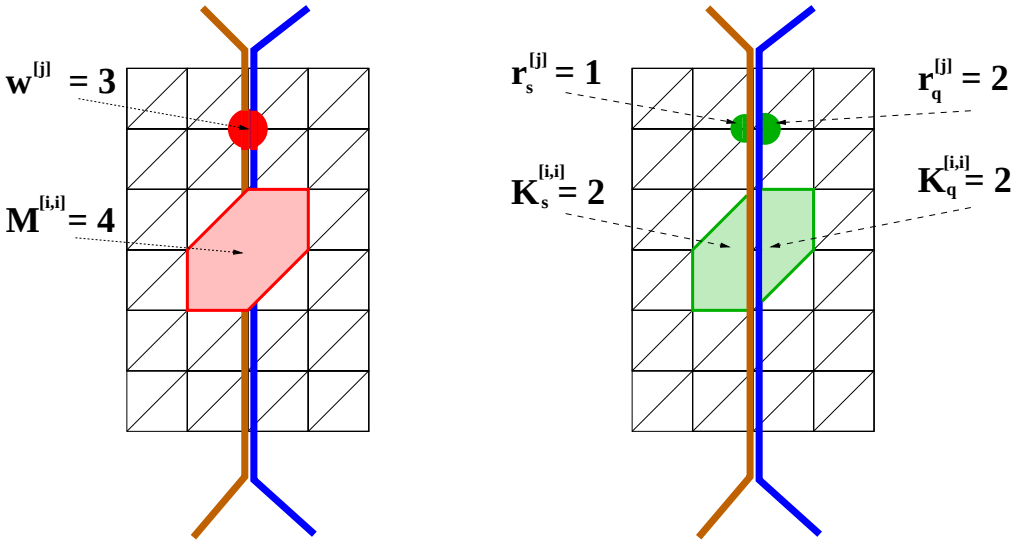

FIGURE 3.1. *Illustration for accumulated and distributed vectors and matrices.*

with $R_s = \underset{i\in\omega_s}{\mathrm{diag}}\{|\sigma^{[i]}|\}$.

REMARK 3.1. *It can be seen easily by (2.11a) that*

$$R \;:=\; \underset{i\in\omega}{\mathrm{diag}}\{|\sigma^{[i]}|\} \;=\; \sum_{s=1}^{P} A_s^T \cdot A_s \quad . \tag{3.5}$$

3.2. Matrix types. The matrix defined by the bilinear form in (1.6) can also be stored in two ways. Regarding to an element-wise domain decomposition, we can store the f.e. matrix accumulated or distributed.

DEFINITION 3.3 (accumulated matrix). *All local restrictions of an accumulated matrix $\mathfrak{M}$ possess the full entries of it and we write*

$$\mathfrak{M}_s \;:=\; A_s \cdot \mathfrak{M} \cdot A_s^T \quad . \tag{3.6}$$

DEFINITION 3.4 (distributed matrix). *We call a matrix K distributed if*

$$\mathsf{K} \;:=\; \sum_{s=1}^{P} A_s^T \cdot \mathsf{K}_s \cdot A_s \quad . \tag{3.7}$$

holds, i.e., each subdomain $\overline{\Omega}_s$ stores only a part of its full values.

3.3. Construction of distributed matrices. The construction of distributed matrices for a non-overlapping element decomposition (see Ex. 2.1) is well known. Here, we generalize that approach for arbitrary overlapping subdomains.

In the f.e. context, a matrix entry $K^{[i,j]}$ is defined by the sum over all bilinear forms on the elements

$$K^{[i,j]} \;:=\; a_\Omega(\varphi^{[j]},\varphi^{[i]}) \;=\; \sum_{\delta^{(r)}\subseteq\Omega} a_{\delta^{(r)}}\left(\varphi^{[j]},\varphi^{[i]}\right) \quad , \tag{3.8}$$

with base functions $\varphi^{[i]}(x)$, $i \in \omega$ and the bilinear form from Sec. 1.

A decomposition of $\overline{\Omega} = \bigcup_{s=1}^{P} \overline{\Omega}_s$ results for the bilinear form $a(\cdot,\cdot)$ in a weighted summation over all subdomains and all their elements.

$$\begin{aligned} a_\Omega(\varphi^{[j]},\varphi^{[i]}) &= \sum_{s=1}^{P} \sum_{\delta^{(r)} \subseteq \Omega_s} \frac{1}{W^{(r)}} a_{\delta^{(r)}}(\varphi^{[j]},\varphi^{[i]}) \\ &= \sum_{\delta^{(r)} \subseteq \Omega} a_{\delta^{(r)}}(\varphi^{[j]},\varphi^{[i]}) \cdot \left(\frac{1}{W^{(r)}} \sum_{s:\, \delta^{(r)} \subseteq \Omega_s} 1 \right) \end{aligned} \tag{3.9}$$

The comparison of (3.8) with (3.9) give us directly the definition for the weights per element.

THEOREM 3.5 (Construction principle for distributed matrices). *In the multi domain approach with $\overline{\Omega} = \bigcup_{s=1}^{P} \overline{\Omega}_s$, a distributed f.e. matrix*

$$\mathsf{K} := \sum_{s=1}^{P} A_s^T \mathsf{K}_s A_s \ . \tag{3.7}$$

has to be calculated in each subdomain by

$$\mathsf{K}_s^{[i,j]} = \sum_{\delta^{(r)} \subseteq \Omega_s} \frac{1}{W^{(r)}} \underbrace{a_{\delta^{(r)}}(\varphi^{[j]},\varphi^{[i]})}_{K_{\delta^{(r)}}^{[i,j]}} \qquad \forall i,j \in \omega_s \tag{3.10}$$

with the weights for each element of the mesh

$$W^{(r)} := \sum_{s:\, \delta^{(r)} \subseteq \Omega_s} 1 \ . \tag{3.11}$$

Proof. The proof was given from Eq. (3.8) to (3.9). ∎

REMARK 3.2. *Eq. (3.10) describes the local accumulation of weighted element matrices. The weight $W^{(r)}$ in (3.11) is the number of subdomains Ω_s an element $\delta^{(r)}$ belongs to. The weight of an element is equal to the minimal weight of its nodes, i.e.,*

$$W^{(r)} = \min_{x^{[i]} \in \overline{\delta}^{(r)}} \left\{ R^{[i,i]} \right\} \ . \tag{3.12}$$

- The non-overlapping elements in Fig. 2.1 imply $W^{(r)} \equiv 1$ for all elements, so that no weighting is necessary. K_s is the local accumulated f.e. matrix in this case .
- The weights for the overlapping elements in Fig. 2.2 are 1, 2 or 4.

4. Matrix-by-vector operations.

4.1. Abstract results. It can be easily seen by Def. 2.5 and Eq. (3.1)-(3.3), (3.7) that

$$\mathsf{K} \cdot \underline{\mathfrak{w}} = \sum_{s=1}^{P} A_s^T \mathsf{K}_s A_s \cdot \underline{\mathfrak{w}} = \sum_{s=1}^{P} A_s^T \mathsf{K}_s \cdot \underline{\mathfrak{w}}_s = \sum_{s=1}^{P} A_s^T \underline{\mathfrak{v}}_s = \underline{\mathfrak{v}} \tag{4.1}$$

can be performed in parallel without communication and results in a distributed vector $\underline{\mathfrak{v}}$. A multiplication with a distributed vector requires conversion (3.3) previously.

The multiplication of an accumulated matrix $\mathfrak{M}$ with a vector requires detailed investigations. The following two theorems will present conditions for admissible matrix patterns for certain classes of multiplications.

THEOREM 4.1. *The multiplication of an accumulated matrix with an accumulated vector results in a vector of the same type*

$$\underline{\mathfrak{w}} \;=\; \mathfrak{M} \cdot \underline{\mathfrak{u}} \tag{4.2}$$

and can be performed in parallel, i.e.

$$\underline{\mathfrak{w}}_s \;=\; \mathfrak{M}_s \cdot \underline{\mathfrak{u}}_s \qquad \forall s = \overline{1,P} \;, \tag{4.3}$$

if and only if

$$\forall i,j \in \omega \;: \qquad \sigma^{[i]} \not\subseteq \sigma^{[j]} \implies \mathfrak{M}^{[i,j]} = 0 \tag{4.4}$$

is true.

Proof. For the proof of theorem 4.1 we start with the definition of an accumulated vector (3.1) and investigate the resulting vector component-wise in global numbering.

$$\begin{aligned}
& \underline{\mathfrak{w}}_s = A_s \underline{\mathfrak{w}} && \forall s = \overline{1,p} \\
\overset{(4.3)}{\Longleftrightarrow} \quad & \mathfrak{M}_s \underline{\mathfrak{u}}_s = A_s \mathfrak{M} \underline{\mathfrak{u}} && \forall s = \overline{1,p} \\
\overset{(3.6)}{\Longleftrightarrow} \quad & \left(A_s \mathfrak{M} A_s^T \cdot \underline{\mathfrak{u}}_s\right)^{[i]} = \left(A_s \mathfrak{M} \cdot \underline{\mathfrak{u}}\right)^{[i]} && \forall s = \overline{1,p}, \quad \forall i \in \omega_s \\
\Longleftrightarrow \quad & \sum_{j \in \omega_s} \mathfrak{M}^{[i,j]} \mathfrak{u}^{[j]} = \sum_{j \in \omega} \mathfrak{M}^{[i,j]} \mathfrak{u}^{[j]} && \forall s = \overline{1,p}, \quad \forall i \in \omega_s
\end{aligned}$$

The statement of the last line is true if and only if the following is true

$$\begin{aligned}
& \forall s = \overline{1,p}, \quad \forall i \in \omega_s \;: && j \notin \omega_s && \implies \quad \mathfrak{M}^{[i,j]} = 0 \\
\overset{(2.9),(2.10)}{\Longleftrightarrow} \quad & \forall s = \overline{1,p} \;: && \{s\} \in \sigma^{[i]} \wedge \{s\} \notin \sigma^{[j]} && \implies \quad \mathfrak{M}^{[i,j]} = 0 \\
\Longleftrightarrow \quad & && \sigma^{[i]} \not\subseteq \sigma^{[j]} && \implies \quad \mathfrak{M}^{[i,j]} = 0
\end{aligned}$$

□

THEOREM 4.2. *The multiplication of an accumulated matrix with a distributed vector results in a vector of the same type*

$$\underline{\mathsf{r}} \;=\; \mathfrak{M} \cdot \underline{\mathsf{v}} \tag{4.5}$$

and can be performed in parallel, i.e.

$$\underline{\mathsf{r}}_s \;=\; \mathfrak{M}_s \cdot \underline{\mathsf{v}}_s \qquad \forall s = \overline{1,P} \;, \tag{4.6}$$

if and only if

$$\forall i,j \in \omega \;: \qquad \sigma^{[i]} \not\supseteq \sigma^{[j]} \implies \mathfrak{M}^{[i,j]} = 0 \tag{4.7}$$

is true.

Proof. For the proof of theorem 4.2 we start with the definition of a distributed vector (3.2) and investigate the resulting vector component-wise in global numbering.

Relation (2.11a) is used.

$$\sum_{s=1}^{P} A_s^T \underline{\mathsf{r}}_s = \underline{\mathsf{r}}$$

$$\sum_{s=1}^{P} A_s^T A_s \mathfrak{M}_s A_s^T \cdot \underline{\mathsf{v}}_s \overset{(3.6)}{=} \sum_{s=1}^{P} A_s^T \mathfrak{M}_s \cdot \underline{\mathsf{v}}_s = \mathfrak{M} \cdot \underline{\mathsf{v}} = \mathfrak{M} \cdot \sum_{s=1}^{P} A_s^T \underline{\mathsf{v}}_s$$

$$\begin{pmatrix} \sum_{s=1}^{P} \sum_{j \in \omega_s} \mathfrak{M}^{[i,j]} \cdot \mathsf{v}_s^{[j]} & , & i \in \omega_s \\ 0 & , & i \notin \omega_s \end{pmatrix} = \left(\sum_{j \in \omega} \mathfrak{M}^{[i,j]} \cdot \sum_{s=1}^{P} \mathsf{v}_s^{[j]} \quad , \quad i \in \omega \right)$$

$$\begin{pmatrix} \sum_{j \in \omega} \sum_{s \in \sigma^{[j]}} \mathfrak{M}^{[i,j]} \cdot \mathsf{v}_s^{[j]} & , & i \in \omega_s \\ 0 & , & i \notin \omega_s \end{pmatrix} = \left(\sum_{j \in \omega} \sum_{s=1}^{P} \mathfrak{M}^{[i,j]} \cdot \mathsf{v}_s^{[j]} \quad , \quad i \in \omega \right)$$

The statement of the last line is true if and only if

$$\begin{aligned} s \in \sigma^{[j]} \wedge i \notin \omega_s &\Longrightarrow \mathfrak{M}^{[i,j]} = 0 \\ \{s\} \subseteq \sigma^{[j]} \wedge \{s\} \not\subseteq \sigma^{[i]} &\Longrightarrow \mathfrak{M}^{[i,j]} = 0 \\ \sigma^{[j]} \not\subseteq \sigma^{[i]} &\Longrightarrow \mathfrak{M}^{[i,j]} = 0 \end{aligned}$$

□

Obviously, we can conclude that if a matrix $\mathfrak{M}$ fulfills (4.4) then $\mathfrak{M}^T$ fulfills also (4.7).

REMARK 4.1. *Intermediate steps in the proofs of theorems 4.1 and 4.2 lead directly to equivalent formulations for (4.4)*

$$\mathfrak{M} A_s^T A_s = \mathfrak{M} \qquad \forall s = \overline{1, P} \tag{4.8}$$

and for (4.7)

$$A_s^T A_s \mathfrak{M} = \mathfrak{M} \qquad \forall s = \overline{1, P} \ . \tag{4.9}$$

Eq. (4.8) and (4.9) have been derived for non-overlapping element domain decompositions by Groh [1] as a condition to apply (4.2) and (4.5) in parallel.

REMARK 4.2. *The sets of subdomains induce a block structure of the unknowns in that way that all nodes with the same set as the representative set $\sigma^{[k]}$ build one block. Now, it follows directly from theorems 4.1 and 4.2 that an accumulated matrix $\mathfrak{M}$ can be applied to (4.2) and (4.5) iff $\mathfrak{M}$ possesses a block diagonal structure induced by the described block structure of nodes. We can write*

$$\mathfrak{M} \ = \ \text{blockdiag}\{\mathfrak{M}_k\} \qquad \textit{with} \qquad \mathfrak{M}_k \ = \ \left\{ \mathfrak{M}^{[i,j]} \right\}_{i,j \in \omega(\sigma^{[k]})} \ . \tag{4.10}$$

REMARK 4.3. *If $\mathfrak{M}$ fulfills (4.10) then also the inverse $\mathfrak{M}^{-1}$ fulfills the proper block condition and can be applied in parallel on (4.2) and (4.5).*

THEOREM 4.3. *If the accumulated matrix $\mathfrak{P}$ fulfills pattern condition (4.4) then the matrix multiplications $\mathfrak{P}^T \cdot \mathsf{K} \cdot \mathfrak{P}$ result in a distributed matrix K^H which can be calculated locally in parallel*

$$\mathsf{K}^H \ = \ \sum_{s=1}^{P} A_s^T \mathsf{K}_s^H A_s \ = \ \sum_{s=1}^{P} A_s^T \left(\mathfrak{P}_s^T \cdot \mathsf{K}_s \cdot \mathfrak{P}_s \right) A_s \ = \ \mathfrak{P}^T \cdot \mathsf{K} \cdot \mathfrak{P} \ . \tag{4.11}$$

Proof. The pattern conditions (4.4) and (4.7) are equivalent to (4.8) and (4.9) which have to be hold for all subdomains. So, we can write

$$\begin{aligned}
\mathfrak{P}^T \cdot \mathsf{K} \cdot \mathfrak{P} &\overset{(3.7)}{=} \mathfrak{P}^T \cdot \sum_{s=1}^{P} A_s^T \mathsf{K}_s A_s \cdot \mathfrak{P} \;=\; \sum_{s=1}^{P} \mathfrak{P}^T A_s^T \mathsf{K}_s A_s \mathfrak{P} \\
&\overset{(4.8)(4.9)}{=} \sum_{s=1}^{P} A_s^T A_s \mathfrak{P}^T A_s^T K_s A_s \mathfrak{P} A_s^T A_s \\
&= \sum_{s=1}^{P} A_s^T \left(\mathfrak{P}_s^T \cdot \mathsf{K}_s \cdot \mathfrak{P}_s \right) A_s \\
&=: \sum_{s=1}^{P} A_s^T \mathsf{K}_s^H A_s \;=\; \mathfrak{P}^T \cdot \mathsf{K} \cdot \mathfrak{P}
\end{aligned}$$

□

4.2. Application of theorems 4.1 and 4.2. We will apply the results of theorems 4.1 and 4.2 on our Ex. 2.1 - 2.2.

There are 9 different sets of subdomains in Ex. 2.1. These sets will be arranged wrt. the number of subdomains therein :

$$\begin{aligned}
&\sigma_V = \sigma^{[25]}, \\
&\sigma_{E_1} = \sigma^{[4]}, \qquad \sigma_{E_2} = \sigma^{[22]}, \qquad \sigma_{E_3} = \sigma^{[28]}, \qquad \sigma_{E_4} = \sigma^{[46]}, \\
&\sigma_{I_1} = \sigma^{[1]}, \qquad \sigma_{I_2} = \sigma^{[7]}, \qquad \sigma_{I_3} = \sigma^{[8]}, \qquad \sigma_{I_4} = \sigma^{[49]} \;.
\end{aligned}$$

The block structure above implies an appropriate block structure in vectors and matrices. The matrix pattern admissible wrt. (4.4) is

$$\mathfrak{M} = \begin{pmatrix}
\mathfrak{M}_V & & & & & & & & \\
\mathfrak{M}_{E_1 V} & \mathfrak{M}_{E_1} & & & & & & & \\
\mathfrak{M}_{E_2 V} & 0 & \mathfrak{M}_{E_2} & & & & & & \\
\mathfrak{M}_{E_3 V} & 0 & 0 & \mathfrak{M}_{E_3} & & & 0 & & \\
\mathfrak{M}_{E_4 V} & 0 & 0 & 0 & \mathfrak{M}_{E_4} & & & & \\
\mathfrak{M}_{I_1 V} & \mathfrak{M}_{I_1 E_1} & \mathfrak{M}_{I_1 E_2} & 0 & 0 & \mathfrak{M}_{I_1} & & & \\
\mathfrak{M}_{I_2 V} & \mathfrak{M}_{I_2 E_1} & 0 & \mathfrak{M}_{I_2 E_3} & 0 & 0 & \mathfrak{M}_{I_2} & & \\
\mathfrak{M}_{I_3 V} & 0 & \mathfrak{M}_{I_3 E_2} & 0 & \mathfrak{M}_{I_3 E_4} & 0 & 0 & \mathfrak{M}_{I_3} & \\
\mathfrak{M}_{I_4 V} & 0 & 0 & \mathfrak{M}_{I_4 E_3} & \mathfrak{M}_{I_4 E_4} & 0 & 0 & 0 & \mathfrak{M}_{I_4}
\end{pmatrix}. \tag{4.12}$$

This matrix can be also expressed in the form

$$\mathfrak{M} \;=\; \begin{pmatrix} \mathfrak{M}_V & 0 & \\ \mathfrak{M}_{EV} & \mathfrak{M}_E & 0 \\ \mathfrak{M}_{IV} & \mathfrak{M}_{IE} & \mathfrak{M}_I \end{pmatrix} \;, \tag{4.13}$$

requiring the appropriate block structures of the submatrices from (4.12). Especially, we have to require that $\mathfrak{M}_V$, $\mathfrak{M}_E$ and $\mathfrak{M}_I$ are block diagonal matrices, from which follows by remark 4.3 that we can apply also the inverse of them.

The previous investigations are completely valid for Ex. 2.2! The only difference consist in the different sets of nodes grouped with the blocks.

5. A parallel AMG.

5.1. Parallel MG. With the data structures and methods introduced in the previous sections we can formulate a parallel MG algorithm, see also [4].

Algorithm 1 Parallel multigrid $\text{PMG}(\mathsf{K}, \underline{\mathfrak{u}}, \underline{\mathfrak{f}}, \ell)$

if $\ell == 1$ **then**
  solve $\sum_{s=1}^{P} A_s^T \mathsf{K} A_s \cdot \underline{\mathfrak{u}} = \underline{\mathfrak{f}}$
else
  $\widetilde{\underline{\mathfrak{u}}} \leftarrow \text{SMOOTH}(\mathsf{K}, \underline{\mathfrak{u}}, \underline{\mathfrak{f}}, \nu)$
  $\underline{\mathfrak{d}} \leftarrow \underline{\mathfrak{f}} - \mathsf{K} \cdot \underline{\mathfrak{u}}$
  $\underline{\mathfrak{d}}^H \leftarrow \mathfrak{P}^T \cdot \underline{\mathfrak{d}}$
  $\underline{\mathfrak{w}}^H \leftarrow 0$
  $\text{PMG}^\gamma(\mathsf{K}^H, \underline{\mathfrak{w}}^H, \underline{\mathfrak{d}}^H, \ell - 1)$
  $\underline{\mathfrak{w}} \leftarrow \mathfrak{P} \cdot \underline{\mathfrak{w}}^H$
  $\widehat{\underline{\mathfrak{u}}} \leftarrow \widetilde{\underline{\mathfrak{u}}} + \underline{\mathfrak{w}}$
  $\underline{\mathfrak{u}} \leftarrow \text{SMOOTH}(\mathsf{K}, \widehat{\underline{\mathfrak{u}}}, \underline{\mathfrak{f}}, \nu)$
end if

The components in Alg.1 have to be chosen such that the following requirements are fulfilled.

- The interpolation of the correction $\underline{\mathfrak{w}} := \mathfrak{P}\underline{\mathfrak{w}}^H$ needs an accumulated interpolation matrix with an *admissible matrix pattern*, see theorem 4.1. If $\mathfrak{P}$ possesses such a pattern then also $\mathfrak{P}^T$ can be applied in the defect restriction.
- The matrix K^H has to be again a distributed matrix to ensure the continuation of the recursive algorithm.
- The matrix K^H has to preserve the null space of K.

5.2. Sequential AMG. Each AMG algorithm consists of the following parts

1. Coarsening, i.e., $\omega = \omega_C \cup \omega_F$.
2. Determining the interpolation weights $P = \{\alpha_{i,j}\}_{i \in \omega,\, j \in \omega_C}$.
   Often $\alpha_{i,j} = \delta_{ij}$, $\forall i, j \in \omega_C$ holds.
3. Calculate the coarse matrix

$$K^H = P^T \cdot K \cdot P \ . \tag{5.1}$$

4. Apply some multigrid procedure using the components determined in 1.-3.

5.2.1. Sequential coarsening. We can use any coarsening strategy for our investigations. Therefore, we present only the simple coarsening instead of the classical approach [7]. We only have to adapt the sequential algorithm such that it works on an arbitrary given set of nodes, i.e., a subset of all nodes.

5.2.2. Determination of interpolation weights. Here, we restrict the calculation of interpolation weights α_{ij} on a given index set. Again, we can use an arbitrary routine with adaption for subsets of nodes.
A correct chosen matrix dependent interpolation is equivalent to the energy minimization technique for determining the interpolation weights. In case of the Laplacian, the condition $\sum_{j \in \omega} \alpha_{i,j} = 1\,,\ \forall i \in \omega$ guarantees the preserving of the null space on coarser grids.

Algorithm 2 Simple coarsening $\textsc{Coarse}(K, \omega, \omega_C, \omega_F)$

$\omega_t \leftarrow \omega$
while $\omega_t \neq \emptyset$ **do**
  pick some $i \in \omega_t$ (strong connections in $K|_\omega$)
  $\omega_C \leftarrow \omega_C \cup \{i\}$
  determine neighbors of $i \longrightarrow U_i$
  $\omega_F \leftarrow \omega_F \cup (U_i \cap \omega_t)$
  $\omega_t \leftarrow \omega_t \setminus (\{i\} \cup U_i \cap \omega_t)$
end while

Algorithm 3 Calculation of interpolation weights $\textsc{Weights}(K, \omega_C, \omega_F, \alpha)$

for all $i \in \omega_F$ **do**
  determine neighbors of $i \longrightarrow U_i$
  for all $j \in U_i \cap \omega_C$ **do**
    calculate α_{ij}
  end for
end for

5.2.3. Calculation of coarse matrix. The classical Galerkin approach

$$\left(K^H\right)^{[ij]} = \sum_{k\in\omega}\sum_{l\in\omega} \alpha_{ki} \cdot K^{[kl]} \cdot \alpha_{lj} \qquad \forall i,j \in \omega_C \tag{5.2}$$

leads directly to a standard routine for calculating the coarse grid matrix. Here, no adaption is necessary.

Algorithm 4 Calculation of coarse matrix $\textsc{Galerkin}(K^H, K, \omega_C, \omega_F, \alpha)$

Require: $K^H = 0$ before first call
for all $k \in \omega_F \cup \omega_C$ **do**
  determine fine and coarse neighbors of $k \longrightarrow F^i$, C^k
  for all $l \in F^k \cup C^k \cup \{k\}$ **do**
    $T \leftarrow\in C^k \cup C^l \cup (\{l\} \cap \omega_C)$
    for all $i \in T$ **do**
      for all $j \in T$ **do**
        $\left(K^H\right)^{[ij]} \leftarrow \left(K^H\right)^{[ij]} + \alpha_{ki} \cdot (K)^{[kl]} \cdot \alpha_{lj}$
      end for
    end for
  end for
end for

This Alg. 4can by dramatically accelerated by use of matrix structures.

5.3. The parallel coarsening. The control of the admissible pattern for interpolation and restriction (4.4,4.7) will be realized by appropriate subsets of nodes as input parameters for Alg. 2–4. Controlling these subsets is the main task of Alg. 5.

The initial call of the coarsening is $\textsc{SigmaCoarse}(\{1, \ldots, P\}, \omega^h, \emptyset, \emptyset)$.

- The statement $\omega := \omega \setminus \omega_t$ blocks redundant calculations by reducing the set of available nodes.
- The recursive algorithm comes to an end if σ consists of only one element.

Algorithm 5 Parallel coarsening $\textsc{SigmaCoarse}(K, \sigma, \omega, \omega_C, \omega_F, \alpha)$

```
if ω == ∅ then
  return
else
  ω_t ← ω(σ)
  Coarse(K, ω_t, ω_C, ω_F)
  Weights(K, ω_C ∩ ω̄(σ), ω_F ∩ ω_t, α)
  ω ← ω \ ω_t
  for all σ̂ ∈ P(σ) such that σ̂ ≠ σ ∧ ∄τ ∈ P(σ) : σ̂ ⊂ τ ≠ σ do
    SigmaCoarse(K, σ̂, ω, ω_C, ω_F, α)
  end for
end if
```

- The special parameters in the call $\textsc{Weights}(K, \omega_C \cap \overline{\omega}(\sigma), \omega_F \cap \omega_t, \alpha)$ guarantee an admissible matrix pattern in P, because

$$\left.\begin{array}{rcl} \omega_C \cap \overline{\omega}(\sigma) & = & \left\{ j \in \omega_C \,:\, \sigma^{[j]} \supseteq \sigma \right\} \\ \omega_F \cap \omega(\sigma) & = & \left\{ i \in \omega_F \,:\, \sigma^{[i]} = \sigma \right\} \end{array}\right\} \Longrightarrow \sigma^{[i]} \subseteq \sigma^{[j]} \ , \tag{5.3}$$

  i.e. by (4.4) that only those α_{ij} are calculated in Weights which fulfill the pattern condition on the interpolation $\mathfrak{P}$.
- The usage of blocking coarsening requires some semaphore structure to synchronize the work in the branches, i.e., which processor performs the coarsening on a set of intersection nodes. After one processor did the coarsening, it will send an bit array to all interested processors describing which nodes of the subset belong the coarse grid. The following calculation of interpolation weights should be done redundantly.
  On the other hand, a non-blocking coarsening requires identical results in the coarsening of all processors $s \in \sigma^{[i]}$ sharing a block of nodes represented by i. Here, we need an identical numbering of the nodes in $\omega(\sigma^{[i]})$ on all processors. Additionally the coarsening strategy has to be strongly deterministic in that case.

5.4. Parallel coarse grid matrix. Looking at the formula (5.2) and having in mind that $P = \{\alpha\}$ fulfills the pattern condition (see (5.3)) we can derive

$$\begin{aligned} \exists s \,:\, \sigma^{[i]} \supseteq \sigma^{[k]} \supseteq \{s\} \subseteq \sigma^{[l]} \subseteq \sigma^{[j]} &\Leftrightarrow \{s\} \subseteq \sigma^{[k]} \cap \sigma^{[l]} \subseteq \sigma^{[i]} \cap \sigma^{[j]} \\ &\Leftrightarrow k, l, i, j \in \omega_s \ . \end{aligned}$$

Therefore, we can write $\forall i, j \in \omega_C$

$$\begin{aligned} \left(K^H\right)^{[ij]} &= \sum_{k \in \omega} \sum_{l \in \omega} \alpha_{ki} \cdot K^{[kl]} \cdot \alpha_{lj} \\ &= \sum_{k \in \omega} \sum_{l \in \omega} \sum_{s \in \sigma^{[k]} \cap \sigma^{[l]}} \alpha_{ki} \cdot K_s^{[kl]} \cdot \alpha_{lj} \\ &= \sum_{s=1}^{P} \left(\sum_{k \in \omega_s} \sum_{l \in \omega_s} \alpha_{ki} \cdot K_s^{[kl]} \cdot \alpha_{lj} \right) \ , \end{aligned}$$

and finally

$$\mathsf{K}_H \;=\; \sum_{s=1}^{P} A_s^T \cdot \mathsf{K}_{H,s} \cdot A_s \;=\; \sum_{s=1}^{P} A_s^T \cdot \mathfrak{P}_s^T \mathsf{K}_{H,s} \mathfrak{P}_s \cdot A_s \;. \tag{5.4}$$

This is an alternative proof of theorem 4.3.

5.5. The parallel AMG algorithm. The parallel AMG algorithm simply collects the subroutines of previous sections. The routine PMG-A is the parallel multigrid algorithm with trivial changes.

Algorithm 6 Parallel AMG PARAMG$(\mathsf{K}, \underline{\mathfrak{u}}, \underline{\mathfrak{f}}, \omega)$

for all $s = 1, \ldots, P$ **do**
  $\mathsf{K}_s^1 \leftarrow \mathsf{K}_s$
  $\omega_s^1 \leftarrow \omega_s$
  $\ell \leftarrow 1$
  repeat
    $\mathfrak{M}_s \leftarrow A_s \left(\sum_{q=1}^{P} A_q^T \mathsf{K}_q^\ell A_q \right) A_s^T$
    $\omega_{C,s} \leftarrow \emptyset\,, \qquad \omega_{F,s} \leftarrow \emptyset$
    SIGMACOARSE$(\mathfrak{M}_s, \{1, \ldots, P\}, \omega_s^\ell, \omega_{C,s}, \omega_{F,s}, \alpha^{\ell+1})$
    $\mathsf{K}_s^{\ell+1} \leftarrow 0$
    GALERKIN$(\mathsf{K}_s^\ell, \mathsf{K}_s^{\ell+1}, \omega_{C,s}, \omega_{F,s}, \alpha^{\ell+1})$
    $\ell \leftarrow \ell + 1$
    $\omega_s^\ell \leftarrow \omega_{C,s}$
  until termination
end for
PMG-A$(\mathsf{K}, \underline{\mathfrak{u}}, \underline{\mathfrak{f}}, 1)$

The coarsening part of Alg. 6 terminates automatically when all sets $\omega(\sigma^{[i]})$ contain only one element ($i \in \omega_s^\ell$).

6. Conclusions. The main advantage of the proposed parallel AMG algorithms consists in the combination of standard AMG components and standard DD approaches. Therefore, it can be easily implemented in given f.e. packages by combining both parts.

The parallel AMG algorithm is not consistent with respect to the number of processors. i.e., the resulting coarse grids change with the processor number. This is no disadvantage in our opinion, because we needed a simple general approach and not one which will change with the hardware. An distribution of elements on the processors should additionally take into account strongly anisotropic coefficients or similar problems in that (macro-/DD-) coarsening. Although the implementation described in the paper above remains to future we are sure that our parallel AMG will achieve satisfying efficiencies.

REFERENCES

[1] U. Groh, *Local realization of vector operations on parallel computers*, Preprint SPC 94-2[1], TU Chemnitz, 1994. in german.

[2] G. Haase, B. Heise, M. Jung, and M. Kuhn, *FEM⊗BEM - a parallel solver for linear and nonlinear coupled FE/BE-equations*, DFG-Schwerpunkt "Randelementmethoden", Report 94-16, University Stuttgart, 1994.

[3] G. Haase, U. Langer, and A. Meyer, *The approximate Dirichlet decomposition method. part I,II*, Computing, 47 (1991), pp. 137–167.

[4] M. Jung, *On the parallelization of multi-grid methods using a non-overlapping domain decomposition data structure*, Applied Numerical Mathematics, 20 (1996).

[5] K. H. Law, *A parallel finite element solution method*, Computer and Structures, 23 (1989), pp. 845–858.

[6] A. Meyer, *A parallel preconditioned conjugate gradient method using domain decomposition and inexact solvers on each subdomain*, Computing, 45 (1990), pp. 217–234.

[7] J. W. Ruge and K. Stüben, *Algebraic multigrid (AMG)*, in Multigrid Methods, S. F. McCormick, ed., vol. 3 of Frontiers in Applied Mathematics, SIAM, Philadelphia, PA, 1987, pp. 73–130.

[1] http://www.tu-chemnitz.de/ftp-home/pub/Local/mathematik/SPC/spc94_2.ps.gz