A Multilevel Extension of the GDSW Overlapping Schwarz Preconditioner in Two Dimensions

Heinlein, Alexander; Rheinbach, Oliver; Röver, Friederike

DOI
10.1515/cmam-2022-0168

Publication date
2023

Document Version
Final published version

Published in
Computational Methods in Applied Mathematics

Research Article

Alexander Heinlein, Oliver Rheinbach* and Friederike Röver

A Multilevel Extension of the GDSW Overlapping Schwarz Preconditioner in Two Dimensions

https://doi.org/10.1515/cmam-2022-0168
Received August 18, 2022; revised March 16, 2023; accepted May 8, 2023

Abstract: Multilevel extensions of overlapping Schwarz domain decomposition preconditioners of Generalized Dryja–Smith–Widlund (GDSW) type are considered in this paper. The original GDSW preconditioner is a two-level overlapping Schwarz domain decomposition preconditioner, which can be constructed algebraically from the fully assembled stiffness matrix. The FROSch software, which belongs to the ShyLU package of the Trilinos software library, provides parallel implementations of different variants of GDSW preconditioners. The coarse problem can limit the parallel scalability of two-level GDSW preconditioners. As a remedy, in the past, three-level GDSW approaches have been proposed, which can significantly extend the range of scalability. Here, a multilevel extension of the GDSW preconditioner is introduced and analyzed. Finally, parallel results for the implementation in FROSch for up to 40 000 cores of the SuperMUC-NG supercomputer at Leibniz Supercomputing Centre (LRZ) and to 48 000 cores of the JUWELS supercomputer at Jülich Supercomputing Centre (JSC) are presented.

Keywords: Domain Decomposition, Overlapping Schwarz Methods, Multilevel, Parallel Implicit Solvers

MSC 2020: 65M55, 65Y05, 65M60

1 Introduction

We consider multilevel extensions of overlapping Schwarz domain decomposition preconditioners of *Generalized Dryja–Smith–Widlund* (GDSW) type [7]. The original GDSW preconditioner is a two-level overlapping Schwarz domain decomposition preconditioner [37] with an energy-minimizing coarse space and well-understood convergence for a wide range of model problems [6, 7, 9, 20]. Moreover, there has been recent development regarding adaptive GDSW coarse spaces [14], which are related to the GenEO (Generalized Eigenvalues in the Overlaps) coarse spaces [35, 36]. An important feature of the GDSW coarse space is that it can be constructed algebraically from the fully assembled stiffness matrix; in particular, it does neither require a coarse triangulation nor Neumann matrices for the subdomains. Adaptive GDSW spaces are generally not algebraic as they require local Neumann matrices; however, an algebraic extension has recently been developed in [23]. Versions of the GDSW preconditioners are implemented in parallel in the FROSch software [15, 17], which is part of the Trilinos software library [41]. The parallel scalability of the two-level GDSW preconditioner [6, 7, 9, 20] is limited by the direct solution of the coarse problem. Therefore, we have considered three-level extensions, where

***Corresponding author: Oliver Rheinbach,** Fakultät für Mathematik und Informatik, Zentrum für effiziente Hochtemperatur-Stoffwandlung (ZeHS); and Universitätsrechenzentrum (URZ), Technische Universität Bergakademie Freiberg, 09596 Freiberg, Germany, e-mail: oliver.rheinbach@math.tu-freiberg.de. https://orcid.org/0000-0002-9310-8533
Alexander Heinlein, Delft Institute of Applied Mathematics, Faculty of Electrical Engineering Mathematics & Computer Science, Delft University of Technology, Mekelweg 4, 2628 CD Delft, Netherlands, e-mail: a.heinlein@tudelft.nl. https://orcid.org/0000-0003-1578-8104
Friederike Röver, Fakultät für Mathematik und Informatik, Zentrum für effiziente Hochtemperatur-Stoffwandlung (ZeHS); and Universitätsrechenzentrum (URZ), Technische Universität Bergakademie Freiberg, 09596 Freiberg, Germany, e-mail: friederike.roever@math.tu-freiberg.de

the coarse problem is solved inexactly by applying another two-level GDSW preconditioner. Note that GDSW coarse spaces of reduced dimension (RGDSW) [9, 20] also show an improved parallel scalability compared with standard GDSW. Two-level GDSW preconditioners are typically suitable for thousands of cores, while two-level RGDSW preconditioners can scale to tens of thousands of cores. For large numbers of subdomains, however, the coarse problem of both methods may become too large to be solved by a sparse direct linear solver.

In [18, 19], three-level GDSW/RGDSW preconditioners have been introduced for two- and three-dimensional problems and first results have been presented. Furthermore, in [21], the partitioning of the coarse problem has been discussed. In [22], we discuss the three-level implementation in FROSch in detail and show parallel results for scalability up to 220 000 cores on the ALCF Theta supercomputer and 85 000 cores of the SuperMUC-NG supercomputer. Our GDSW implementation in FROSch is related to other parallel implementations of scalable overlapping Schwarz methods [24, 31]. The three-level GDSW and RGDSW approaches are also related to three-level (or multilevel) BDDC methods [2, 30, 34, 38].

2 Model Problem

We consider the weak formulation associated with the Poisson model problem on a two-dimensional domain Ω: find $u \in H^1_{\Gamma_D}(\Omega)$ such that

$$a(u,v) = \int_\Omega \rho \nabla u \cdot \nabla v = \int_\Omega f v = f(v) \quad \text{for all } v \in H^1_{\Gamma_D}(\Omega),$$

where $\rho(x) > 0$ for all $x \in \Omega$ and uniform in a subdomain Ω_i. For simplicity, we assume that $\rho(x) \equiv 1$. For the boundary conditions, we consider a Dirichlet boundary condition on a measurable subset $\partial\Omega_D$ on $\partial\Omega$ and a Neumann boundary condition on the remaining boundary $\partial\Omega_N = \partial\Omega \setminus \partial\Omega_D$. We use piecewise linear finite elements on a shape regular triangulation. In the following, we denote the finite element space by V^h. For the construction of a two-level overlapping Schwarz preconditioner, we need a decomposition into overlapping subdomains Ω_i'. In practice, we start with a decomposition into non-overlapping subdomains Ω_i; see also the left image of Figure 1 for a visualization. In order to obtain Ω_i', we extend each non-overlapping subdomain Ω_i by q layers of elements. We then obtain the overlap $\delta = qh$.

3 The Two-Level GDSW Preconditioner

The GDSW coarse space V^0 is spanned by basis functions defined by their values on the interface Γ of the non-overlapping decomposition. We extend the values on the interface discrete harmonically to the interior of each subdomain such that the energy is minimized; cf., e.g., [37, Section 4.4]. The interface values are chosen as restrictions of the nullspace of the global Neumann matrix to the interface components, vertices v and edges ε. We obtain vertex basis functions $\theta_v(x)$ and edge basis functions $\theta_\varepsilon(x)$. See also Figure 1 (right) for a visualization of the interface coarse components.

Following the theory of overlapping Schwarz methods, we define a splitting of $u \in V^h$ by

$$u = \sum_{i=0}^{N} R_i^T u_i, \quad \{u_i \in V_i,\ 0 \le i \le N\}, \tag{3.1}$$

where the coarse function $u_0 = \mathcal{U}$ is chosen as

$$\mathcal{U} = \sum_{v \in \overline{\Omega}} \theta_v(x) u(v) + \sum_{\varepsilon \in \overline{\Omega}} \theta_\varepsilon(x) \overline{u}_\varepsilon,$$

where $\sum_{v\in\overline{\Omega}}$ is the sum over all vertices on the interface and $\sum_{\varepsilon\in\overline{\Omega}}$ the sum over all edges on the interface, respectively. Here, $\overline{u}_\varepsilon$ indicates the average of u over the edge. Finite element discretization yields a linear equation system of the form

$$Kv = f.$$

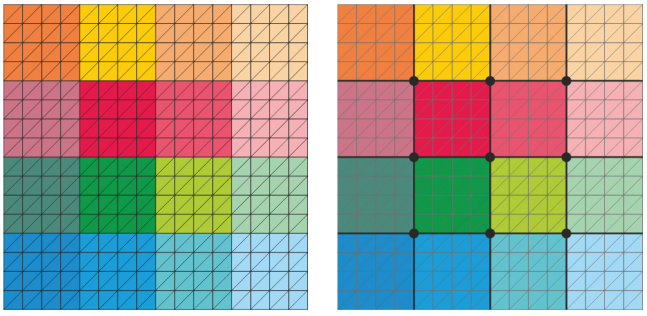


Figure 1: Structured decomposition of a two-dimensional domain (left) and corresponding coarse mesh, consisting of vertices (black dots) and edges (thicker black lines), for the GDSW coarse space (right). Image in parts taken from [22, Figure 1].

Then the preconditioned operator can be written in the following form:

$$P_{\mathrm{GDSW}} = M_{\mathrm{GDSW}}^{-1} K = \Phi {K_0}^{-1} \Phi^T K + \sum_{i=1}^{N} R_i^T K_i^{-1} R_i K,$$

where $K_0 = \Phi^T K \Phi$ and $K_i = R_i K R_i^T$ are the coarse matrix and the local subdomain matrices on the overlapping subdomains, respectively. The matrix Φ contains the coarse basis functions as columns, which span the coarse space V^0. In particular, we have

$$\Phi = \begin{pmatrix} \Phi_I \\ \Phi_\Gamma \end{pmatrix} = \begin{pmatrix} -K_{II}^{-1} K_{I\Gamma} \Phi_\Gamma \\ \Phi_\Gamma \end{pmatrix},$$

where K_{II} and $K_{I\Gamma}$ are submatrices of K, if ordered with respect to the interior (I) and interface (Γ) degrees of freedom,

$$K = \begin{pmatrix} K_{II} & K_{I\Gamma} \\ K_{\Gamma I} & K_{\Gamma\Gamma} \end{pmatrix}.$$

Note that $K_{II} = \mathrm{diag}(K_{II}^{(i)})$ is a block-diagonal matrix, where $K_{II}^{(i)}$ corresponds to the i-th non-overlapping subdomain. Therefore, the extensions can be computed independently and concurrently for all subdomains. For scalar elliptic problems and the general case of a decomposition into John domains [13], the condition number of GDSW preconditioner is bounded by

$$\kappa(M_{\mathrm{GDSW}}^{-1} K) \leq C \left(1 + \frac{H}{\delta}\right) \left(1 + \log\left(\frac{H}{h}\right)\right)^2,$$

where H is the diameter of the subdomains, h the typical diameter of a finite element, and δ is the overlap; C is a constant independent of the other problem parameters; cf. [6, 7]. The theory of GDSW preconditioners also provides a condition number bound, e.g., for the case of compressible elasticity in three dimensions; cf. [8]. Note that, if not defined otherwise, in the following, h is the diameter of the smallest finite element and H the largest diameter of the non-overlapping subdomains.

4 Multilevel Extension for the GDSW Preconditioners

For a large number of cores and thus a large number of subdomains, the solution of the coarse problem of the GDSW preconditioner may not be efficient any more. To overcome the scaling bottleneck arising from using a sequential or parallel direct solver on the second level of the GDSW preconditioner, we apply the GDSW preconditioner recursively to the coarse problem [18, 19]. A similar approach is also used in other multilevel domain decomposition methods [2, 27, 30, 31, 33, 34, 38] and, of course, in multigrid methods [12]. To specify the different levels of the preconditioner, we introduce new indices. We change the common order of numeration, known from the theory of overlapping Schwarz methods. Each index corresponds to the respective level of the preconditioner such that the coarse space of the two-level method becomes $V^{(2)}$; hence it is the second level. The

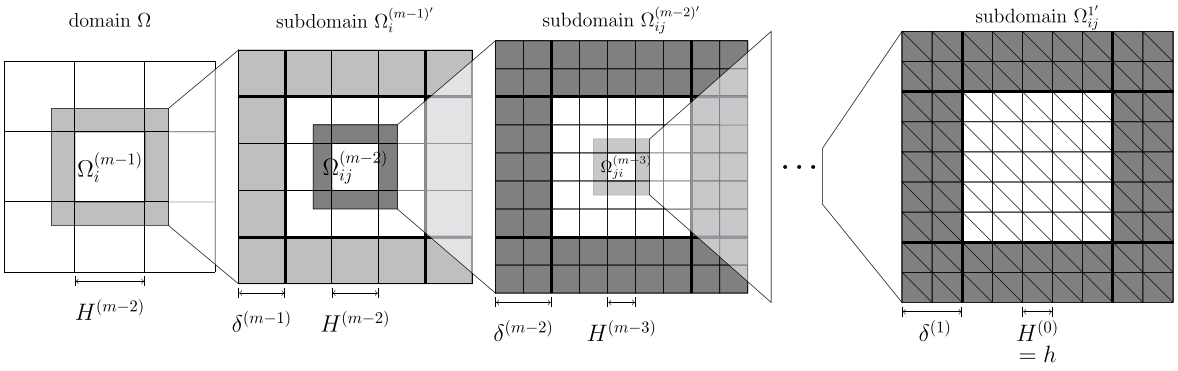


Figure 2: Structured decomposition of a two-dimensional computational domain Ω into non-overlapping subdomains $\Omega_i^{(k)}$, a zoom into one overlapping subdomain $\Omega_i^{(m-1)'}$ consisting of subdomains $\Omega_{ij}^{(m-2)}$ (left middle). Furthermore, we show the zoom into one overlapping subdomain $\Omega_{ij}^{(1)'}$ of the first level (right). For all coarser levels ($k > 1$), the degrees of freedom correspond to vertices ν_k and edges ε_k of the non-overlapping decomposition. This figure is based on [18, Figure 1].

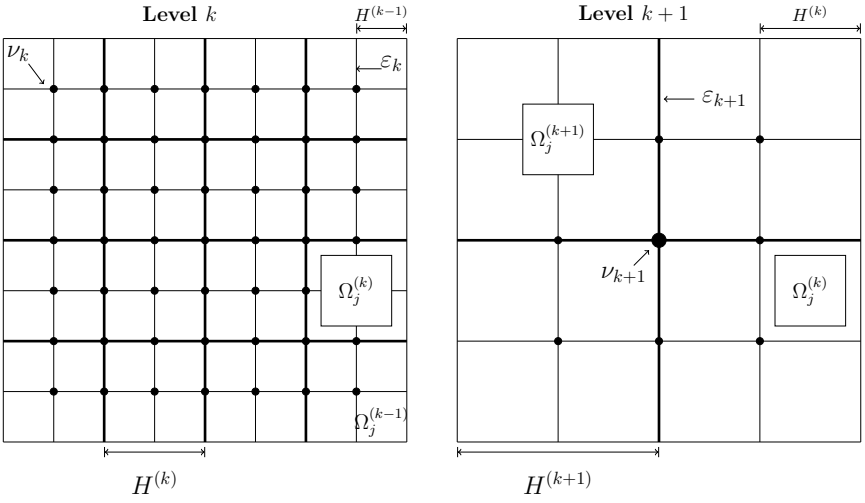


Figure 3: Grid and decomposition into non-overlapping subdomains on the coarser levels k and $k+1$. The dots correspond to the vertices and the line segments between the dots to the edges. Note that the dots on level k correspond to the vertices of the decomposition into the subdomains $\Omega_j^{(k-1)}$. We use the index k since it corresponds to the degrees of freedom on level k.

numbering of the space thus follows the numbering of the levels, i.e., the coarse space on level m of the m-level GDSW preconditioner is denoted by $V^{(m)}$. For each application of the preconditioner, we need an additional layer of decomposition. We therefore decompose the computational domain into $N^{(k)}$ subdomains $\Omega_j^{(k)}$ with diameter $H^{(k)}$ on level k. Each subdomain consists of a union of subdomains from the previous level,

$$\Omega_j^{(k)} = \bigcup_{i:\Omega_{ji}^{(k-1)} \subset \Omega_j^{(k)}} \Omega_{ji}^{(k-1)}.$$

As for the construction of the two-level GDSW preconditioner, we add $q^{(k)}$ layers of subdomains of level $k-1$ to $\Omega_j^{(k)}$ to obtain the overlapping subdomains $\Omega_j^{(k)'}$. We denote the overlap by $\delta^{(k)} = q^{(k)} H^{(k-1)}$ as we have done for the two-level method; see Figure 2 for an illustration of the decomposition into the m levels and Table 1 and Figure 3 for an overview of the notation for the respective level of the preconditioner.

Over all levels, we obtain a family of subspaces $V^{(k)}$, $k = 1, \dots, m$, where each subspace $V^{(k)}$, $k < m$, is decomposed into $V_j^{(k)}$, $j = 1, \dots, N_k$, the subspaces corresponding to the overlapping subdomains $\Omega_j^{(k)'}$. We introduce the new index for the level identification as well to the aforementioned restriction operators R_j such that

$$R_j^{(k)} \colon V^{(k)} \to V_j^{(k)},$$

Level	Domain/ subdomain	Function	Space	Restriction operator	Coarse basis function	Interpolation operator
1	$\Omega_{j_1}^{(1)}$	u_{j_1}	$V_{j_1}^{(1)}$	$R_{j_1}^{(1)}$		
2	$\Omega_{j_2}^{(2)}$	$\mathcal{U}_{j_2}^{(2)}$	$V_{j_2}^{(2)}$	$R_{j_2}^{(2)}$	$\Phi_1^{(2)}$	$\mathcal{R}_{j_2}^{(2)^T}$
⋮	⋮	⋮	⋮	⋮	⋮	
$m-1$	$\Omega_{j_{m-1}}^{(m-1)}$	$\mathcal{U}_{j_{m-1}}^{(m-1)}$	$V_{j_{m-1}}^{(m-1)}$	$R_{j_{m-1}}^{(m-1)}$	$\Phi_{m-2}^{(m-1)}$	$\mathcal{R}_{j_{m-1}}^{(m-1)^T}$
m	Ω	$\mathcal{U}^{(m)}$	$V^{(m)}$		$\Phi_{m-1}^{(m)}$	$\mathcal{R}^{(m)^T}$

Table 1: Overview of the notation for each level of the multilevel extensions. For better readability, we will omit the index of the subdomain number j such that we denote $\Omega_{j_k}^{(k)}$ simply by $\Omega_j^{(k)}$.

where $V_j^{(k)}$ is the local subspace of $V^{(k)}$ to the overlapping subdomain $\Omega_j^{(k)'}$. We define the m-level GDSW preconditioner

$$M_{\text{ML-GDSW}}^{-1} = \Phi_1^{(2)} \overbrace{\underbrace{K^{(2)^{-1}}}\Big(\Phi_2^{(3)} \overbrace{\underbrace{K^{(3)^{-1}}}\Big(\vdots \overbrace{\underbrace{\Phi_{m-1}^{(m)} K^{(m)^{-1}} \Phi_{m-1}^{(m)^T}}_{m\text{th level}} + \underbrace{\sum_{j=1}^{N^{(m-1)}} R_j^{(m-1)^T} K_j^{(m-1)^{-1}} R_j^{(m-1)}}_{(m-1)\text{th level}}}^{} \Big) \Phi_2^{(3)^T} + \underbrace{\sum_{j=1}^{N^{(2)}} R_j^{(2)^T} K_j^{(2)^{-1}} R_j^{(2)}}_{\text{2nd level}}}^{} \Big) \Phi_1^{(2)^T} + \underbrace{\sum_{i=1}^{N^{(1)}} R_i^{(1)^T} K_i^{(1)^{-1}} R_i^{(1)}}_{\text{1st level}},$$

where $K^{(k)} = \Phi_{k-1}^{(k)^T} K^{(k-1)} \Phi_{k-1}^{(k)}$ is the k-th coarse matrix related to the coarse space $V^{(k)}$, which is spanned by the coarse basis function contained in $\Phi_{k-1}^{(k)^T}$. Here, $K_j^{(k)} = R_j^{(k)} K^{(k)} R_j^{(k)^T}$ are the local subproblem matrices corresponding to the subdomain $\Omega_j^{(k)'}$.

4.1 Condition Number Bound for the Multilevel Extension

In order to establish a condition number bound, we follow the theory of two-level overlapping Schwarz methods given in [37, Chapters 2 and 3]. The proof is a direct extension of the proof given in [6, 7]. Following the standard notation of additive Schwarz methods, we define, on the finest level $k = 1$, corresponding to (3.1), a decomposition of u by summing over all levels,

$$u = \mathcal{R}(m)^T \mathcal{U}^{(m)} + \sum_{k=1}^{m-1} \sum_{i=1}^{N_k} \mathcal{R}_i^{k^T} \mathcal{U}_i^{(k)},$$

where $\mathcal{R}(m)^T \colon V^{(m)} \to V^h$ is the interpolation from the coarse space on level m to the standard finite element space V^h. In detail, $\mathcal{R}^m$ is built from the coarse basis functions such that $\mathcal{R}(m) = \Phi_{m-1}^{(m)^T} \Phi_{m-2}^{(m-1)^T} \cdots \Phi_2^{(3)^T} \Phi_1^{(2)^T}$. Correspondingly, we have $\mathcal{R}^k = R_j^{(k)} \Phi_{k-1}^{(k)^T} \Phi_{k-1}^{(k-1)^T} \cdots \Phi_1^{(2)^T}$.

We consider the higher levels k, $k \geq 2$, where we operate on the GDSW function space. Similar to (3.1), we define a splitting of $\mathcal{U}^{(k)}$, $k \geq 2$, by

$$\mathcal{U}^{(k)} = R_0^{(k)^T} \mathcal{U}_0^{(k)} + \sum_{j=1}^{N_k} R_j^{(k)^T} \mathcal{U}_j^{(k)},$$

where $\mathcal{U}_0^{(k)} = \mathcal{U}^{(k+1)}$, $R_j^{(k)^T} \colon V_j^{(k)} \to V^{(k)}$ and $R_0^{(k)^T}$ defines the interpolation from the higher level to the next lower level,

$$R_0^{(k)^T} \colon V^{(k+1)} \to V^{(k)}.$$

For the analysis, it is now essential to estimate the constant in the *Stable Decomposition*; see, e.g., [37, Assumption 2.2 and Lemma 2.5]; see also the early pioneering work of Lions on the Alternating Schwarz Method [28, 29]. Therefore, we need to find an estimate for C_0^2 in

$$a(\mathcal{U}^{(m)}, \mathcal{U}^{(m)}) + \sum_{j=k}^{m-1} \sum_{j=1}^{N^{(k)}} a_j^{(k)}(\mathcal{U}_j^{(k)}, \mathcal{U}_j^{(k)}) \leq C_0^2 a(u,u), \tag{4.1}$$

where we sum over the coarser levels and where $a_j^{(k)}(u,v) = \int_{\Omega_j^{(k)}} \nabla u \cdot \nabla v$. To obtain an estimate for C_0^2, we consider each level separately. The discrete Sobolev inequality is an essential tool to bound the functions $\mathcal{U}^{(k)}$ as a sum of vertex and edge functions. It is an important tool in the theory of iterative substructuring methods [4, 37]. We give the formulation presented in [37, Lemma 4.15]. Let $H^{(k)}$ be the maximum diameter of the non-overlapping subdomains on the current level k; see also Figure 3.

Lemma 4.1 (Discrete Sobolev Inequality). *Let α be any convex combinations of $u(x)$, with $x \in \Omega_i$ a bounded subdomain in the plane of diameter H. Then, for any $u \in V^h$,*

$$\|u\|^2_{L^\infty(\Omega_i)} \leq C(1 + \log(H/h))\|u\|^2_{H^1(\Omega_i)},$$
$$\|u - \alpha\|^2_{L^\infty(\Omega_i)} \leq C(1 + \log(H/h))|u|^2_{H^1(\Omega_i)}.$$

Different proofs are given in [3–5] and also [37, Lemma 4.15]. Note, however, that we need these bounds for the different, non-standard basis functions occurring in GDSW. We therefore formulate our next assumption.

Assumption 4.1. There exists a constant $C_{\text{Sob}}^{(k)}$ such that

$$\|\mathcal{U}^{(k)}\|^2_{L^\infty(\Omega_j^k)} \leq C_{\text{Sob}}^{(k)} \|\mathcal{U}^{(k)}\|^2_{H^1(\Omega_j^k)} \quad \text{for all } \mathcal{U}^{(k)} \in V^{(k)}. \tag{4.2}$$

The constant $C_{\text{Sob}}^{(k)}$ depends on the ratio between the sizes $H^{(k)}$ of the non-overlapping subdomains $\Omega_j^{(k)}$ and the element size h.

We will prove these assumptions and the assumptions stated below for GDSW and RGDSW type basis functions elsewhere, where will also give more details on the constants.

For the coarser levels, we define the function $\mathcal{U}^{(k)}$ as a sum of the vertex and edge functions, as in [37, (5.13)],

$$\mathcal{U}^{(k)}(x) = \sum_{\nu_k \in \overline{\Omega}} \mathcal{U}_{\nu_k}^{(k)}(x) + \sum_{\varepsilon_k \in \overline{\Omega}} \mathcal{U}_{\varepsilon_k}^{(k)}(x),$$

where

$$\mathcal{U}_{\nu_k}^{(k)}(x) = \mathcal{U}^{(k-1)}(\nu_k)\theta_{\nu_k}(x), \tag{4.3}$$
$$\mathcal{U}_{\varepsilon_k}^{(k)}(x) = \overline{\mathcal{U}_{\varepsilon_k}^{(k-1)}}\theta_{\varepsilon_k}(x)$$

and where $\overline{\mathcal{U}_{\varepsilon_k}^{(k-1)}}$ is the average over an edge ε_k. Besides Lemma 4.1, an estimate of the H_1-seminorm and L^2-norm of the edge functions is needed to establish a bound for the functions $\mathcal{U}^{(k)}$. We recall the estimate for the edge function θ_{ε_2} used to construct $\mathcal{U}^{(2)}$ given in [26], which was first given in [10].

Lemma 4.2. *Let $\Omega_i \subset \mathbb{R}^2$ be a John domain* [13] *with diameter $H^{(1)}$, let $\varepsilon_2 \subset \partial\Omega_i^{(1)}$ be an edge, and let $\theta_{\varepsilon_2} \in V^h(\Omega_i^{(1)})$ be a finite element function which equals* 1 *at all nodes of θ_{ε_2}, vanishes at the other nodes on $\partial\Omega_i^{(1)}$, and is discrete harmonic in $\Omega_i^{(1)}$. We have*

$$|\theta_{\varepsilon_2}|^2_{H^1(\Omega_i^{(1)})} \leq C(1 + \log(H^{(1)}/h)), \tag{4.4}$$
$$\|\theta_{\varepsilon_2}\|^2_{L^2(\Omega_i^{(1)})} \leq C{H^{(1)}}^2(1 + \log(H^{(1)}/h)).$$

The edges of the respective level of the edge function θ_{ε_k} is defined correspondingly to θ_{ε_2}. The values in the interior of the subdomains are defined by a GDSW function interpolation. We define a linear interpolation operator I^k to the coarse space $V^{(k)}$ by

$$I^k(h)(x) = \sum_{\nu_k \in \overline{\Omega}} h(\nu_k)\theta_{\nu_k}(x) + \sum_{\varepsilon_k \in \overline{\Omega}} \overline{h}_{\varepsilon_k}\theta_{\varepsilon_k}(x).$$

Assumption 4.2. There exists a constant $C_{\theta_{\varepsilon_k}}$, which depends on the ratio of $H^{(k)}$ and h and which bounds the energy of the edge function θ_{ε_k} and satisfies the estimate for the L^2-norm such that

$$\begin{aligned} |\theta_{\varepsilon_k}|^2_{H^1(\Omega_j^k)} &\leq C_{\theta_{\varepsilon_k}}, \\ \|\theta_{\varepsilon_k}\|^2_{L^2(\Omega_j^k)} &\leq {H^{(k)}}^2 C_{\theta_{\varepsilon_k}}. \end{aligned} \tag{4.5}$$

The estimate for the functions $\mathcal{U}^{(k)}$, $k = 1, \ldots, m$, is built on the same arguments; thus we discuss an arbitrary level in the hierarchy as an example. Using (4.2), we obtain, for the vertex contributions of $\mathcal{U}^{(k)}$ (see (4.3)),

$$\begin{aligned} |\mathcal{U}^{(k)}_{v_k}(x)|_{H^1(\Omega_j^{(k-1)})^2} = |\mathcal{U}^{(k-1)}(v_k)\theta_{v_k}(x)|_{H^1(\Omega_j^{(k-1)})^2} &\leq |\mathcal{U}^{(k-1)}(v_k)|^2 |\theta_{v_k}(x)|_{H^1(\Omega_j^{(k-1)})^2} \\ &\leq \|\mathcal{U}^{(k-1)}\|_{L^\infty(\Omega_j^{(k-1)})^2} |\theta_{v_k}(x)|_{H^1(\Omega_j^{(k-1)})^2} \\ &\overset{(4.2)}{\leq} C^{(k-1)}_{\text{Sob}} \|\mathcal{U}^{(k-1)}\|_{H^1(\Omega_j^{(k-1)})^2} \end{aligned} \tag{4.6}$$

and the edge contributions (see 4.4),

$$\begin{aligned} |\mathcal{U}^{(k)}_{\varepsilon_k}(x)|_{H^1(\Omega_j^{(k-1)})^2} = |\overline{\mathcal{U}^{(k-1)}_{\varepsilon_k}}\theta_{\varepsilon_k}(x)|_{H^1(\Omega_j^{(k-1)})^2} &\leq |\overline{\mathcal{U}^{(k-1)}_{\varepsilon_k}}|^2 |\theta_{\varepsilon_k}(x)|_{H^1(\Omega_j^{(k-1)})^2} \\ &\leq \|\mathcal{U}^{(k-1)}\|_{L^\infty(\Omega_j^{(k)})^2} |\theta_{\varepsilon_k}(x)|_{H^1(\Omega_j^{(k-1)})^2} \\ &\overset{(4.2),\,(4.5)}{\leq} C^{(k-1)}_{\text{Sob}} C_{\theta_{\varepsilon_k}} \|\mathcal{U}^{(k-1)}\|_{H^1(\Omega_j^{(k-1)})^2}. \end{aligned} \tag{4.7}$$

Combining (4.6) and (4.7), the coarse function is bounded by

$$|\mathcal{U}^{(k)}|_{H^1(\Omega_j^{(k-1)})^2} \leq C C^{(k-1)}_{\text{Sob}} C_{\theta_{\varepsilon_k}} \|\mathcal{U}^{(k-1)}\|_{H^1(\Omega_j^{(k-1)})^2}$$

The L^2-norm is removed by subtracting the integral mean value of $\mathcal{U}^{(k-1)}$ over each subdomain $\Omega_j^{(k-1)}$. It remains to consider the parts of (4.1) associated with the local subspaces $V_j^{(k)}$. For the estimate of the first level of overlapping subdomains, we refer to the proof of [37, Lemma 3.12], which allows the estimate of the parts related to the local spaces $V_j^{(1)}$ that depends on $1 + \frac{H^{(1)}}{\delta^{(1)}}$. Parts of the proof for the local subspaces depend on the finite elements functions being linear. Furthermore, the proof indicates a dependency on the coarser level since the function restricted to one overlapping subdomain $\Omega_j^{(1)}$ can be expressed as $u_j = R_j^{(1)^T}(\theta_j^{(1)}\omega^{(1)})$, where $\theta_j^{(1)}$ is a partition of unity as defined in [37, Lemma 3.4] and $\omega^{(1)} = u - \mathcal{U}^{(2)}$. However, on the higher levels $k = 2, \ldots, m-1$, the functions are not standard piecewise linear finite elements functions; instead, we operate on the GDSW function space. Following the ideas for the standard case, we make the following assumption.

Assumption 4.3. There exist constants $C^{(k)}$ and $C_{\delta^{(k)}}$, which depend on the next higher level and on the ratio of the subdomain size $H^{(k)}$ and the overlap $\delta^{(k)}$, such that

$$\sum_{j=1}^{N^{(k)}} a_j^{(k)}(\mathcal{U}_j^{(k)}, \mathcal{U}_j^{(k)}) \leq C^{(k)} C_{\delta^{(k)}}.$$

Performing and combining the estimates given above for the coarse levels, we can estimate the coarsest level as well as the coarse overlapping levels such that we obtain

$$C_0^2 \leq \prod_{k=1}^{m} C^{(k-1)}_{\text{Sob}} C_{\theta_{\varepsilon_k}} C^{(k)} C_{\delta^{(k)}}$$

for the constant in the *Stable Decomposition*. In summary, we obtain the following estimate for the condition number bound of the m-level GDSW preconditioner.

Theorem 4.1. *For $\Omega \in \mathbb{R}^2$ being decomposed into John domains, the condition number of the m-level GDSW preconditioner is bounded by*

$$\kappa(M^{-1}_{\text{ML-GDSW}} K) \leq C \prod_{k=1}^{m} C^{(k-1)}_{\text{Sob}} C_{\theta_{\varepsilon_k}} C^{(k)} C_{\delta^{(k)}}, \tag{4.8}$$

where $C^{(k)}_{\text{Sob}}$ and $C_{\theta_{\varepsilon_k}}$ depend on the ratio of the subdomain sizes $H^{(k)}$ and the elements size h. The constant $C_{\delta^{(k)}}$ depends on the ratio of the overlap $\delta^{(k)}$ and the subdomain size $H^{(k)}$.

The theorem indicates that, with each additional level, the condition number bound rises. Therefore, we expect the condition number as well as the number of iterations to increase with an increasing number of levels. This assumption is confirmed by our numerical results presented in Section 5.1. Note that a theory of multilevel GDSW may, possibly, also be built on the theory of inexact subdomain solvers as in [11].

5 Numerical Results

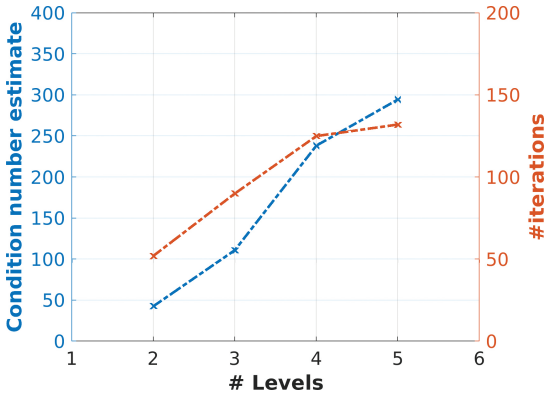
For the numerical and parallel evaluation of the multilevel extension, we consider our Laplacian model problem presented in Section 2 in two dimensions with a generic right-hand side vector of ones $(1 \dots 1)^T$. For the assembly of the stiffness matrix, we employ the Trilinos package Galeri [40]; we use the default boundary conditions of Galeri, that is, homogeneous Dirichlet boundary conditions on the whole boundary $\partial\Omega$. The parallel implementation of the multilevel extension is included in the Fast and Robust Overlapping (FROSch) preconditioner framework [15, 17] of the Trilinos software library [41]. We refer to [22] for a detailed description of the additions to the software in order to apply the multilevel extension. As a Krylov iteration method, we apply the preconditioned conjugate gradient method (PCG) provided by the Trilinos package Belos (BelosPseudoBlockCG) [39] with a relative stopping criterion $\|r_k\|_1/\|r_0\|_2 \leq 10^{-6}$, where r_k is the residual after the k-th iteration and r_0 the initial residual. The implementation offers a condition number estimate using the tridiagonal matrix constructed in the Lanczos process. We solve the arising subproblems using PardisoMKL [1] interfaced through the Trilinos package Amesos2 [32]. In all of our numerical experiments, on all levels, each subdomain is assigned to a specific MPI rank and processor core. We performed all numerical tests on the GCS supercomputer SuperMUC-NG.

5.1 Numerical Scalability of the Multilevel Extension

In this section, we present the numerical results supporting the condition number bound (4.8) in Theorem 4.1. Here, we use a structured decomposition on all levels. According to (4.8), each level introduces additional constants to the bound; thus we expect the condition number to increase with each additional level. Applying the preconditioner with two to five levels, we obtain an increase for the estimated condition number from 42.66 for two levels to 294.51 for five levels; see the table in Figure 4. From two to three levels and three to four level, the condition number more than doubles. However, from four to five levels, the rise is not so severe. The number of iterations confirms the results of an increasing condition number bound with each recursive application of the GDSW preconditioner, although the increase in the number of iterations is not as strong as the increase for the condition number $\kappa(M^{-1}K)$. For our setup, the size of the coarsest problem $\dim(K^{(k)})$ reduces by more than 90 % with each recursive application of the preconditioner; see the table in Figure 4 (right). This coincides with the objective to extend the scalability of the preconditioner. In Section 5.2, we discuss the weak parallel scalability of the implementation. Considering (4.8), classically, we expect a logarithmic dependency on the ratio of the size of the subdomains $H^{(k)}$ and the element size h. Moreover, we expect a linear dependency on the ratio of the size of the subdomains $H^{(k)}$ and the overlap $\delta^{(k)}$.

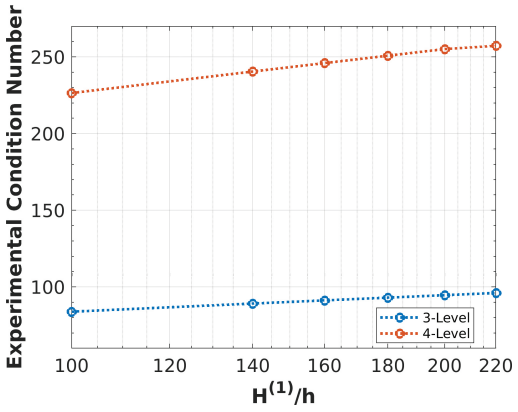
Correspondingly, we perform test runs varying the parameters of the preconditioner. For this study, we apply three- and four-level GDSW preconditioners. We only vary the parameters of the levels one, two, and three. Applying the fourth level, we keep the parameters fixed. Note that the computation of the overlap is performed in an algebraic fashion such that the size of the overlap deviations from the geometric value are possible in certain geometric configurations. Indeed, we note that deviation of all geometric values is inherent to the algebraic construction, and therefore, in this paper, the values for ratios of the subdomains sizes, element sizes, and the overlaps are always approximations.

We first vary the parameters from the first level. Assembling the stiffness matrix using the Galeri package, we set the number of nodes on each subdomain and not the number of elements. Thus, it follows that $H^{(1)}/h$ is an approximate value.



Level	dim($K^{(k)}$)	$\kappa(M^{-1})$	Iter
2	48 641	42.66	52
3	2 945	110.99	90
4	161	238.37	125
5	5	294.51	132

Figure 4: Experimental condition number and number of iterations for the GDSW preconditioner in two dimensions using two to five levels. We apply the preconditioner to the Laplace model problem. The number of subdomains on the first level and the number of cores is $N_1 = 16\,384$. We always choose 180^2 nodes for each non-overlapping subdomain $\Omega_j^{(1)}$. The number of subdomains on the coarser levels is chosen such that 4^2 subdomains $\Omega_{ji}^{(k-1)}$ form $\Omega_j^{(k)}$. Thus we have $H^{(2)}/H^{(1)} = 4$, $H^{(3)}/H^{(2)} = 4$, and $H^{(4)}/H^{(3)} = 4$. We choose $\delta^{(1)} \approx 4h$ and $\delta^{(k)} = 1$, $k = 2, 3, 4$.



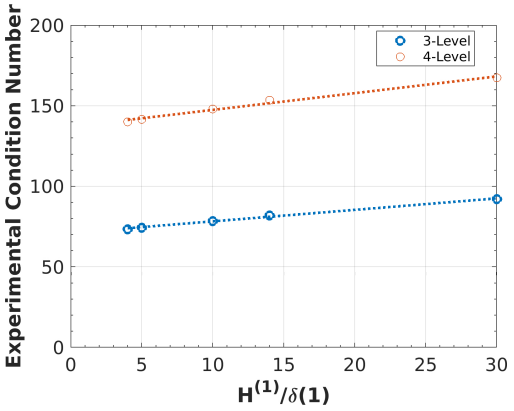
	3 levels		4 levels	
$H^{(1)}/h$	$\kappa(M^{-1}K)$	Iter	$\kappa(M^{-1}K)$	Iter
100	83.95	77	226.53	117
140	89.26	79	240.55	121
160	91.33	81	246.06	122
180	93.13	82	250.88	123
200	94.73	83	255.18	123
220	96.16	86	257.33	127

Figure 5: Numerical scalability, semi-logarithmic plot, varying $H^{(1)}/h$ for the multilevel GDSW methods with three and four levels. We apply the preconditioner to the two-dimensional Laplace model problem. The number of subdomains on the first level and the number of cores is $N^{(1)} = 16\,384$. The other parameters determining the condition number bound are kept constant such that we have $H^{(2)}/H^{(1)} = 4$, $H^{(1)}/\delta^{(1)} = 20$, and $H^{(2)}/\delta^{(2)} = 4$. For the fourth level, we chose 16 cores, thus $H^{(3)}/H^{(2)} = 8$, and we chose $H^{(3)}/\delta^{(3)} = 8$. The problem size changes from 163 840 000 for the smallest test to 792 985 600 for the largest test.

In Figure 5, we vary the number of degrees of freedom per subdomain. The results for the three- and four-level method indicate a polylogarithmic dependency on $H^{(1)}/h$, which is consistent with our expectations considering (4.8) and our assumptions. As expected, the experimental condition number for the four-level method is higher, precisely by factor of around 2.7. The number of Krylov iterations increases slowly. For the three-level method, the number of iterations increases from 77 to 86 and for the four-level method from 117 to 127; see also the table in Figure 5.

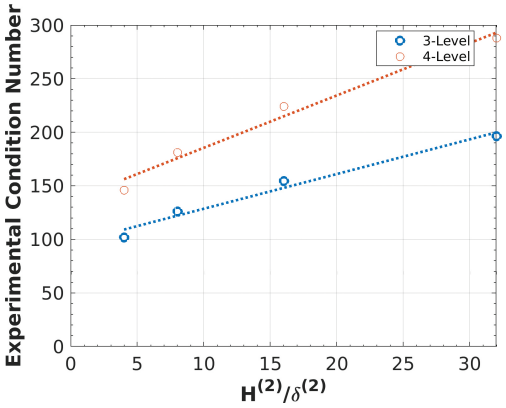
The condition number bound (4.8) depends on the overlap on the different levels. We first consider a change of the overlap $\delta^{(1)}$ on the first level. The results presented in Figure 6 indicate the linear dependency on $H^{(1)}/\delta^{(1)}$, consistent with our expectations. For this test, the estimated condition number of the four-level method is higher by a factor of 1.9. Note that, for the test varying $H^{(1)}/h$, the value for $H^{(3)}/H^{(2)}$ was higher, and thus a larger increase from three to four levels is expected. The increase of the number of iterations is similar. In particular, we obtain an increase from 73 to 80 for the three-level method and from 86 to 96 for the four-level method; see also the table in Figure 6.

We move forward to the coarse level parameters $H^{(2)}$ and $\delta^{(2)}$. Changing the overlap $\delta^{(2)}$ on the second level, we expect a linear dependency, which is consistent with the results presented in Figure 7. The factor for the increase in the condition number from three to four levels is 1.5; see also the column $\kappa(M^{-1}K)$ in the table



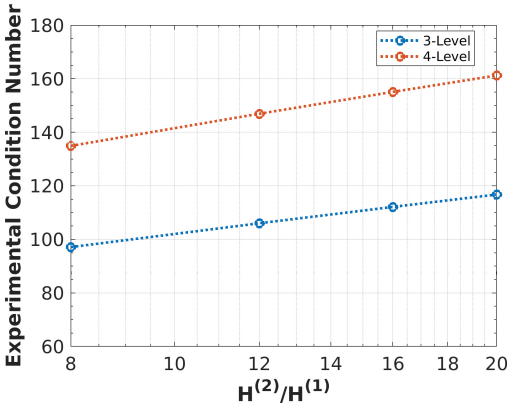
	3 levels		4 levels	
$H^{(1)}/\delta^{(1)}$	$\kappa(M^{-1}K)$	Iter	$\kappa(M^{-1}K)$	Iter
4	73.55	73	140.10	86
5	74.50	74	141.80	87
10	78.51	74	148.19	90
15	82.13	76	153.47	91
30	92.17	80	167.43	96

Figure 6: Numerical scalability varying $H^{(1)}/\delta^{(1)}$ for the multilevel GDSW methods with three and four levels. We apply the preconditioner to the two-dimensional Laplace model problem. The number of subdomains on the first level and the number of cores is 4 096. We choose $H^{(1)}/h = 120$ such that the problem size is 58 982 400. The other parameters determining the condition number bound are kept constant such that we have $H^{(2)}/H^{(1)} = 4$ and $H^{(2)}/\delta^{(2)} = 4$. For the fourth level, we chose 16 cores, thus $H^{(3)}/H^{(2)} = 4$, and we chose $H^{(3)}/\delta^{(3)} = 4$. The dotted line is a least square fit of a linear function to the data.



	3 levels		4 levels	
$H^{(2)}/\delta^{(2)}$	$\kappa(M^{-1})$	Iter	$\kappa(M^{-1})$	Iter
4	102.14	78	146.12	83
8	126.23	91	181.41	93
16	154.64	99	224.44	104
32	196.64	104	288.29	115

Figure 7: Numerical scalability varying $H^{(2)}/\delta^{(2)}$ for the multilevel GDSW methods with three and four levels. We apply the preconditioner to the two-dimensional Laplace model problem. The number of subdomains on the first level and the number of cores is 16 384. We choose $H^{(1)}/h = 80$ such that the problem size is 104 857 600. The other parameters determining the condition number bound are kept constant such that we have $H^{(2)}/H^{(1)} = 32$ and $H^{(1)}/\delta^{(1)} = 40$. We choose $H^{(3)}/H^{(2)} = 2$. The overlap $\delta^{(3)}$ is chosen such that only the degrees of freedoms on the interface are shared among the processes. The dotted line is a least square fit of a linear function to the data.



		3 levels		4 levels	
$H^{(2)}/H^{(1)}$	# cores	$\kappa(M^{-1}K)$	Iter	$\kappa(M^{-1}K)$	Iter
8	4 096	97.16	82	134.97	87
12	9 212	106.06	86	146.97	93
16	16 384	112.17	89	155.13	95
20	25 600	116.79	90	161.27	98

Figure 8: Numerical scalability, semi-logarithmic plot, varying $H^{(2)}/H^{(1)}$ for the multilevel GDSW methods with three and four levels. We apply the preconditioner to the two-dimensional Laplace model problem. To vary $H^{(2)}/H^{(1)}$, we change the number of cores. The overlap on all levels is kept constant such that $H^{(1)}/h = 100$, $H^{(1)}/\delta^{(1)} = 25$, and $H^{(2)}/\delta^{(2)} = 4$. The size of the largest problem in this study is 256 000 000. For the fourth level, we chose 4 cores, thus $H^{(3)}/H^{(2)} = 4$, and we chose $H^{(3)}/\delta^{(3)} = 4$.

in Figure 7. The smaller value compared to the other tests is consistent with the smaller value for $H^{(3)}/H^{(2)}$. The slope in the number of iterations and condition number bound is higher than the slope obtained for the variation of $H^{(1)}/\delta^{(1)}$. The condition number almost doubles changing $H^{(2)}/\delta^{(2)}$ from 4 to 32 for both methods.

We considered a variation of $H^{(2)}/H^{(1)}$ as well. Varying $H^{(2)}/H^{(1)}$, we expect a linear graph on semi-logarithmic scale, as we expect $C_{\text{Sob}}^{(k)}$ to be logarithmically dependent on the ratio of the subdomain and element sizes. This is confirmed by the results presented in Figure 8. The number of iterations increases by 12 for the three-level method and 11 for the four-level method; see the column *Iter* in the table of Figure 8. In total, our experiments confirm the expected linear and logarithmic dependencies. Furthermore, every additional level increases the experimental condition number as well as the number of iterations, confirming the estimate for the condition number bound with the assumptions made for the constants.

5.2 Weak Parallel Scalability Study

The FROSch framework has shown good weak parallel scalability employing three levels for the Laplace and linear elasticity model problem [16, 18, 19, 22] up to the order of 100 000 subdomains and cores. Here, we show new weak parallel scalability results in two dimensions, applying the most recent parallel implementation of the three-level method in FROSch. We compare the results to the standard two-level method scaling from 900 to 40 000 cores. For the decomposition of the coarse level, we compare an explicit structured decomposition to the decomposition obtained by partitioning the coarse problem by the parallel hypergraph method (PHG) provided by the Trilinos package Zoltan2 [42]. The partitioning of the coarse problem by Zoltan2 is the default procedure for the multilevel extensions in FROSch as described in [22]. For linear elasticity, good parallel scalability results have been obtained using PHG in [16, 22]. In all tests, we have 200^2 nodes on each subdomain; thus the largest problem is of size 1 600 000 000 degrees of freedom. Typically, the two-level method will fail for a large number of cores since the direct solver will run out of memory when factorizing the coarse problem. Here, the subdomain problems are chosen quite small in order to leave a substantial amount of main memory for the factorization of the coarse problem in the two-level method. We will then be able to compare the estimated condition numbers for the two-level and the three-level method also for a large number of cores. We set the overlap $\delta^{(1)} \approx 10h$. The overlap on the coarser level is set to one ($\delta^{(2)} \approx 1H^{(1)}$). The number of subdomains on the higher level is always approximately smaller by a factor of 100. The coarse problem is solved on a subset of processes for the three-level method, while for the two-level method, the problem is solved using the PardisoMKL direct solver on a single process. The communication is performed in two steps for the two-level method and in a single step for the three-level method. For a detailed description of the communication pattern used in FROSch, we refer to [22]. With respect to the time to solution and the weak parallel scalability, our results in Table 2 and Figure 9 show very similar results for the two-level and the three-level methods for the problem parameters and the range of processor cores considered here. This is mostly due to our scalar two-dimensional problem with a setup using small subdomains, which still allows the two-level method to factorize the coarse problem. Nevertheless, for a larger number of cores, the performance of the two-level method will eventually degrade before failing to factorize the coarse problem.

In the following, we discuss the numerical results for the different methods and the computation time. By *Solver Time*, we denote the sum of the time to build the preconditioner (*Setup Time*) and of the time for the Krylov iterations (*Krylov Time*). According to the theory, we can expect numerical scalability, which is what we observe; see, e.g., Figure 10. We further expect the three-level method to have higher estimated condition numbers as well as iteration counts. Our assumptions are confirmed by the results of our tests; see Figure 10 and Table 2. For the largest test, we obtain an experimental condition number of 30.95 for the two-level method and 129.78 for the three-level method and a structured decomposition into subdomains $\Omega_j^{(2)}$. Consistent therewith, we obtain higher iteration counts using three levels (100 iterations for 40 000 cores) than using the standard method (48 iterations for 40 000 cores). It also seems that the two-level method approaches the asymptotic bound earlier than the three-level method; see Table 2, columns *Iter* and $\kappa(M^{-1}K)$. However, employing PHG for the partitioning of the coarse problem, the number of iterations increases for the three-level method by a factor of 1.5. The unstructured decomposition on the coarse level has no effect on the two-level method since, in this case, the

		Two-level GDSW				Three-level GDSW			
#subdom.($\Omega_j^{(1)}$) = #cores	#subdom.($\Omega_j^{(2)}$)	$\kappa(M^{-1}K)$	Iter	Solver Time	Parallel effic.	$\kappa(M^{-1}K)$	Iter	Solver Time	Parallel effic.
900	9	30.90	45	3.84 s	100 %	106.03	70	4.24 s	91 %
1 600	16	30.91	45	3.92 s	98 %	114.77	78	4.51 s	85 %
6 400	64	30.93	47	4.73 s	81 %	126.26	92	5.26 s	73 %
10 000	100	30.94	47	4.82 s	80 %	127.81	96	5.83 s	66 %
32 400	324	30.95	48	7.77 s	49 %	129.64	99	7.05 s	54 %
40 000	400	30.95	48	7.64 s	50 %	129.78	100	6.49 s	59 %
						Three-level GDSW PHG partitioning			
900	9					96.01	83	4.84 s	79 %
1 600	16					122.66	91	5.02 s	76 %
6 400	64					146.52	115	5.89 s	65 %
10 000	100					176.82	129	6.39 s	60 %
32 400	324					204.84	145	8.33 s	46 %
40 000	400					220.48	147	10.42 s	37 %

Table 2: Weak parallel scalability for the Laplace model problem in two dimensions on SuperMUC-NG. We apply the preconditioner with two and three levels. On the second level, we apply an unstructured decomposition by Zoltan2, which we denote by *PHG*. We have 200^2 nodes on each subdomain. The parameters determining the condition number are chosen such that $\delta^{(1)} = 10h$, $H^{(2)}/H^{(1)} = 10$, and $\delta^{(2)} \approx 1H^{(1)}$. By *Solver Time*, we denote the time to build the preconditioner (*Setup Time*) and to perform the Krylov iteration (*Krylov Time*). The parallel efficiency is computed considering the fastest method for 900 cores (3.84 s) as a baseline.

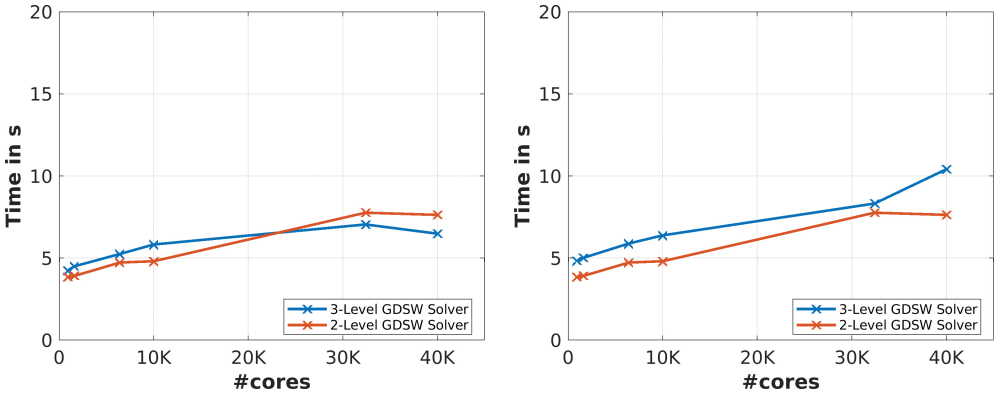


Figure 9: Weak scalability results for the *Solver Time* with a structured decomposition (left) and an unstructured decomposition by Zoltan2 (right) of the coarser level.

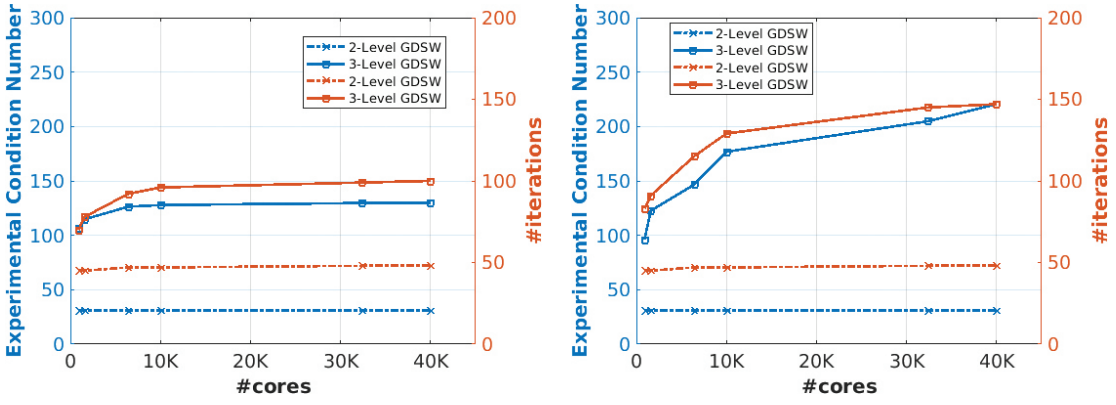


Figure 10: Weak numerical scalability for the Laplace model problem in two dimensions using a structured decomposition on both levels (left) and using an unstructured decomposition on the coarse level (right). Corresponding data in Table 2.

#subdom.($\Omega_j^{(1)}$)	#subdom.($\Omega_j^{(2)}$)	dim($K^{(2)}$)	Struct. dim($K^{(3)}$)	PHG dim($K^{(3)}$)
900	9	2 581	16	24
1 600	16	4 641	33	49
6 400	64	18 881	161	255
10 000	100	29 601	261	438
32 400	324	96 481	901	1 557
40 000	400	119 201	1 121	1 964

Table 3: Dimension of the coarse problem corresponding to the data in Table 2. By *Struct.*, we denote the structured decomposition of the coarser level, and by *PHG*, we denote the unstructured partitioning by Zoltan2 using the parallel hypergraph (PHG) method.

coarse problem is solved in a single process. Moreover, we note that the experimental condition number is higher for the unstructured decomposition. This is also reflected in the numbers of iterations; see Figure 10 (right). Scaling from 900 to 40 000 cores, the condition number more than doubles ($\kappa = 96.01$ for 900 cores, $\kappa = 220.48$ for 40 000 cores), while the number of iteration increases from 83 to 147; see Table 2. Considering the *Solver Time*, all methods are very fast such that the time stays below 10 s, except for 40 000 cores (10.42 s) and a partition of the coarse problem by Zoltan2. Nevertheless, for the structured case, the three-level method is faster reaching 32 400 cores. Based on the sizes of the coarse problem, presented in Table 3, we expect the three-level method to scale, while the two-level method will reach its scalability limit. For 40 000 cores, the coarse problem of the two-level method is larger than 100 000 degrees of freedom, whereas $\dim(K^{(3)}) = 1\,121$ (structured) and $\dim(K^{(3)}) = 1\,964$ (PHG). Thus, also the default partitioning in FROSch is expected to outperform the two-level method for a higher number of cores.

For a detailed description of the additional steps to be performed to partition the coarse problem by PHG, we refer to [22]. Note that these steps include communication. In addition, the size of the coarse problem increases using *PHG*; see Table 3.

Parallel Scalability on JUWELS

We performed further weak parallel scalability test on the supercomputer JUWELS at the Jülich Supercomputing Centre (JSC). The results are listed in Table 4. The setup of the test is as on SuperMUC-NG. On JUWELS, however, we use an algebraic construction of the interface on the first level and the unstructured decomposition by Zoltan2 with the PHG partitioning method on the coarse level. This application of FROSch is completely algebraic. As a result, we see small differences for the experimental condition number as well as for the number of Krylov iterations.

For $H/h = 200$, the two-level method fails for 40 000 cores and more; see Table 4. The three-level method can solve the problem on 48 400 cores; however, the parallel efficiency is low since the local problems are too small. For $H/h = 400$, the method shows a better parallel efficiency; however, we only have results for up to 10 000 cores; see Table 5.

6 Conclusion

We have introduced a multilevel extension to the overlapping Schwarz GDSW preconditioner. For our scalar elliptic model problem, we have established a condition number bound, where each recursive application of the preconditioner degrades the bound. This is similar to multilevel BDDC methods. The bound is based on three assumptions involving constants depending on parameters of the preconditioner. The standard estimates cannot be used since the bounds are needed for GDSW basis functions. To make the dependency on the parameters explicit for these functions remains future work.

		Two-level GDSW				Three-level GDSW			
#subdom.($\Omega_j^{(1)}$) = #cores	#subdom.($\Omega_j^{(2)}$)	$\kappa(M^{-1}K)$	Iter	Solver Time	Parallel effic.	$\kappa(M^{-1}K)$	Iter	Solver Time	Parallel effic.
900	9	30.90	44	2.89 s	100 %	96.44	84	3.56 s	84 %
1 600	16	30.92	46	3.09 s	96 %	122.53	92	3.64 s	82 %
6 400	64	30.94	48	4.45 s	67 %	146.81	114	6.30 s	47 %
10 000	100	30.04	48	5.59 s	53 %	177.92	129	7.81 s	38 %
32 400	324	30.95	49	30.21 s	1 %	208.64	145	9.48 s	31 %
40 000	400	—	—	—	—	222.95	147	10.43 s	29 %
48 400	484	—	—	—	—	202.08	146	10.24 s	29 %

Table 4: Weak parallel scalability for the Laplace model problem in two dimensions on the JUWELS supercomputer. We apply the preconditioner with two and three levels. On the second level, we apply an unstructured decomposition by Zoltan2. The computation of the interface components on the first level is performed in an algebraic fashion. We have 200^2 nodes on each subdomain. The parameters determining the condition number are chosen such that $\delta^{(1)} = 10h$, $H^{(2)}/H^{(1)} = 10$, and $\delta^{(2)} \approx 1H^{(1)}$. By *Solver Time*, we denote the time to build the preconditioner (*Setup Time*) and to perform the Krylov iteration (*Krylov Time*). The parallel efficiency is computed considering the fastest method for 900 cores as a baseline.

		Two-level GDSW				Three-level GDSW			
#subdom.($\Omega_j^{(1)}$) = #cores	#subdom.($\Omega_j^{(2)}$)	$\kappa(M^{-1}K)$	Iter	Solver Time	Parallel effic.	$\kappa(M^{-1}K)$	Iter	Solver Time	Parallel effic.
900	9	45.19	55	10.40 s	100 %	124.22	97	12.89 s	81 %
1 600	16	45.19	55	10.58 s	98 %	150.67	108	13.87 s	75 %
6 400	64	45.20	56	12.88 s	81 %	179.62	129	17.34 s	60 %
10 000	100	45.21	56	14.08 s	74 %	214.05	145	19.78 s	53 %

Table 5: Weak parallel scalability for the Laplace model problem in two dimensions on the JUWELS supercomputer. We apply the preconditioner with two and three levels. On the second level, we apply an unstructured decomposition by Zoltan2. The computation of the interface components on the first level is performed in an algebraic fashion. We have 400^2 nodes on each subdomain. The parameters determining the condition number are chosen such that $\delta^{(1)} = 10h$, $H^{(2)}/H^{(1)} = 10$, and $\delta^{(2)} \approx 1H^{(1)}$. By *Solver Time*, we denote the time to build the preconditioner (*Setup Time*) and to perform the Krylov iteration (*Krylov Time*). The parallel efficiency is computed considering the fastest method for 900 cores as a baseline.

We have used the parallel implementation of the GDSW preconditioner included in the FROSch framework for our numerical tests. The results in our numerical study, using up to five levels, are consistent with the theory. The results for the weak parallel scalability study show that both the two- and three-level methods show a comparable performance for the problem parameters and the number of cores considered here. For a larger number of cores, we expect the three-level method to outperform the two-level method as in [16, 22].

With respect to the parameters of the multilevel method, it remains to be investigated which choice is most efficient, depending on the application. In practice, we often found that using a small overlap (even a single element) is most efficient in terms of the time to solution even if it results in a larger number of iterations. This is the case since it saves significant time in the factorizations of the local subdomain problems. With respect to the number of levels, our strategy currently is to increase the number of levels when a significant portion of the computing time is spent in the coarse problem. However, this may not be the most efficient strategy, and this needs to be tested in the future for different types of problems.

Acknowledgment: This work is part of the Ph.D. thesis of the third author at Fakultät 1 of Technische Universität Bergakademie Freiberg, Germany. The authors also gratefully acknowledge the Gauss Centre for Supercomputing e. V. (www.gauss-centre.eu) for computing time on the GCS Supercomputer SuperMUC-NG at Leibniz Supercomputing Centre (www.lrz.de). The authors gratefully acknowledge the Gauss Centre for Supercomputing e. V. for funding this project by providing computing time through the John von Neumann Institute for Computing (NIC) on the GCS Supercomputer JUWELS [25] at Jülich Supercomputing Centre (JSC). The authors

also acknowledge the compute cluster (DFG project no. 397252409) of the Faculty of Mathematics and Computer Science of Technische Universität Bergakademie Freiberg, operated by the Universitätsrechenzentrum URZ.

Funding: The second and third author would like to acknowledge funding by the Deutsche Forschungsgemeinschaft (DFG) under the DFG project number 441509557 within the DFG SPP 2256 *Variational Methods for Predicting Complex Phenomena in Engineering Structures and Materials*.

References

[1] C. Alappat, A. Basermann, A. R. Bishop, H. Fehske, G. Hager, O. Schenk, J. Thies and G. Wellein, A recursive algebraic coloring technique for hardware-efficient symmetric sparse matrix-vector multiplication, *ACM Trans. Parallel Comput.* **7** (2020), DOI 10.1145/3399732.

[2] S. Badia, A. F. Martín and J. Principe, Multilevel balancing domain decomposition at extreme scales, *SIAM J. Sci. Comput.* **38** (2016), no. 1, C22–C52.

[3] J. H. Bramble, A second order finite difference analog of the first biharmonic boundary value problem, *Numer. Math.* **9** (1966), 236–249.

[4] J. H. Bramble, J. E. Pasciak and A. H. Schatz, The construction of preconditioners for elliptic problems by substructuring. I, *Math. Comp.* **47** (1986), no. 175, 103–134.

[5] S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, Springer, New York, 2008.

[6] C. R. Dohrmann, A. Klawonn and O. B. Widlund, A family of energy minimizing coarse spaces for overlapping Schwarz preconditioners, in: *Domain Decomposition Methods in Science and Engineering XVII*, Lect. Notes Comput. Sci. Eng. 60, Springer, Berlin (2008), 247–254.

[7] C. R. Dohrmann, A. Klawonn and O. B. Widlund, Domain decomposition for less regular subdomains: Overlapping Schwarz in two dimensions, *SIAM J. Numer. Anal.* **46** (2008), no. 4, 2153–2168.

[8] C. R. Dohrmann and O. B. Widlund, Hybrid domain decomposition algorithms for compressible and almost incompressible elasticity, *Internat. J. Numer. Methods Engrg.* **82** (2010), no. 2, 157–183.

[9] C. R. Dohrmann and O. B. Widlund, On the design of small coarse spaces for domain decomposition algorithms, *SIAM J. Sci. Comput.* **39** (2017), no. 4, A1466–A1488.

[10] M. Dryja and O. B. Widlund, Some domain decomposition algorithms for elliptic problems, in: *Iterative Methods for Large Linear Systems*, Academic Press, Boston (1990), 273–291.

[11] G. Haase, U. Langer and A. Meyer, Domain decomposition preconditioners with inexact subdomain solvers, *J. Numer. Linear Algebra Appl.* **1** (1992), no. 1, 27–41.

[12] W. Hackbusch, *Multigrid Methods and Applications*, Springer Ser. Comput. Math. 4, Springer, Berlin, 1985.

[13] P. Hajłasz, Sobolev inequalities, truncation method, and John domains, in: *Papers on Analysis: A Volume Dedicated to Olli Martio on the Occasion of his 60th Birthday*, Rep. Univ. Jyväskylä Dep. Math. Stat. 83, University of Jyväskylä, Jyväskylä (2001), 109–126.

[14] A. Heinlein, A. Klawonn, J. Knepper, O. Rheinbach and O. B. Widlund, Adaptive GDSW coarse spaces of reduced dimension for overlapping Schwarz methods, *SIAM J. Sci. Comput.* **44** (2022), no. 3, A1176–A1204.

[15] A. Heinlein, A. Klawonn, S. Rajamanickam and O. Rheinbach, FROSch: A fast and robust overlapping Schwarz domain decomposition preconditioner based on Xpetra in Trilinos, in: *Domain Decomposition Methods in Science and Engineering XXV*, Lect. Notes Comput. Sci. Eng. 138, Springer, Cham (2020), 176–184.

[16] A. Heinlein, A. Klawonn, S. Rajamanickam and O. Rheinbach, A three-level extension for fast and robust overlapping Schwarz (FROSch) preconditioners with reduced dimensional coarse space, in: *Domain Decomposition Methods in Science and Engineering XXVI*, Lect. Notes Comput. Sci. Eng. 145, Springer, Cham (2023), 505–513.

[17] A. Heinlein, A. Klawonn and O. Rheinbach, A parallel implementation of a two-level overlapping Schwarz method with energy-minimizing coarse space based on Trilinos, *SIAM J. Sci. Comput.* **38** (2016), no. 6, C713–C747.

[18] A. Heinlein, A. Klawonn, O. Rheinbach and F. Röver, A three-level extension of the GDSW overlapping Schwarz preconditioner in two dimensions, in: *Advanced Finite Element Methods with Applications*, Lect. Notes Comput. Sci. Eng. 128, Springer, Cham (2019), 187–204.

[19] A. Heinlein, A. Klawonn, O. Rheinbach and F. Röver, A three-level extension of the GDSW overlapping Schwarz preconditioner in three dimensions, in: *Domain Decomposition Methods in Science and Engineering XXV*, Lect. Notes Comput. Sci. Eng. 138, Springer, Cham (2020), 185–192.

[20] A. Heinlein, A. Klawonn, O. Rheinbach and O. B. Widlund, Improving the parallel performance of overlapping Schwarz methods by using a smaller energy minimizing coarse space, in: *Domain Decomposition Methods in Science and Engineering XXIV*, Lect. Notes Comput. Sci. Eng. 125, Springer, Cham (2018), 383–392.

[21] A. Heinlein, O. Rheinbach and F. Röver, Choosing the subregions in three-level FROSch preconditioners, in: *WCCM-ECCOMAS 2020*, Virtual Congress (2021)), http://dx.doi.org/10.23967/wccm-eccomas.2020.084.

[22] A. Heinlein, O. Rheinbach and F. Röver, Parallel scalability of three-level FROSch preconditioners to 220000 cores using the Theta supercomputer, *SIAM J. Sci. Comput.* (2022), DOI 10.1137/21M1431205.
[23] A. Heinlein and K. Smetana, A fully algebraic and robust two-level Schwarz method based on optimal local approximation spaces, July, preprint (2022), https://arxiv.org/abs/2207.05559.
[24] P. Jolivet, F. Hecht, F. Nataf and C. Prud'homme, Scalable domain decomposition preconditioners for heterogeneous elliptic problems, in: *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ACM, New York (2013), 1–11.
[25] Jülich Supercomputing Centre, JUWELS cluster and booster: Exascale pathfinder with modular supercomputing architecture at Juelich supercomputing centre, *J. Large-Scale Res. Facilities* **7** (2021), DOI 10.17815/jlsrf-7-183.
[26] A. Klawonn, O. Rheinbach and O. B. Widlund, An analysis of a FETI-DP algorithm on irregular subdomains in the plane, *SIAM J. Numer. Anal.* **46** (2008), no. 5, 2484–2504.
[27] F. Kong and X.-C. Cai, A highly scalable multilevel Schwarz method with boundary geometry preserving coarse spaces for 3D elasticity problems on domains with complex geometry, *SIAM J. Sci. Comput.* **38** (2016), no. 2, C73–C95.
[28] P.-L. Lions, Interprétation stochastique de la méthode alternée de Schwarz, *C. R. Acad. Sci. Paris Sér. A-B* **286** (1978), no. 7, A325–A328.
[29] P.-L. Lions, On the Schwarz alternating method. I, in: *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, Society for Industrial and Applied Mathematics, Philadelphia (1988), 1–42.
[30] J. Mandel, B. Sousedík and C. R. Dohrmann, Multispace and multilevel BDDC, *Computing* **83** (2008), no. 2–3, 55–85.
[31] L. F. Pavarino and S. Scacchi, Parallel multilevel Schwarz and block preconditioners for the Bidomain parabolic-parabolic and parabolic-elliptic formulations, *SIAM J. Sci. Comput.* **33** (2011), no. 4, 1897–1919.
[32] M. Sala, K. S. Stanley and M. A. Heroux, On the design of interfaces to sparse direct solvers, *ACM Trans. Math. Software* **34** (2008), no. 2, Article ID 9.
[33] S. Scacchi, A hybrid multilevel Schwarz method for the bidomain model, *Comput. Methods Appl. Mech. Engrg.* **197** (2008), no. 45–48, 4051–4061.
[34] J. Šístek, B. Sousedík, J. Mandel and P. Burda, Parallel implementation of multilevel BDDC, in: *Numerical Mathematics and Advanced Applications 2011*, Springer, Berlin (2012), 681–689.
[35] N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein and R. Scheichl, A robust two-level domain decomposition preconditioner for systems of PDEs, *C. R. Math. Acad. Sci. Paris* **349** (2011), no. 23–24, 1255–1259.
[36] N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein and R. Scheichl, Abstract robust coarse spaces for systems of PDEs via generalized eigenproblems in the overlaps, *Numer. Math.* **126** (2014), no. 4, 741–770.
[37] A. Toselli and O. Widlund, *Domain Decomposition Methods—Algorithms and Theory*, Springer Ser. Comput. Math. 34, Springer, Berlin, 2005.
[38] X. Tu, Three-level BDDC in two dimensions, *Internat. J. Numer. Methods Engrg.* **69** (2007), no. 1, 33–59.
[39] The Belos project team, the Belos project website.
[40] The Galeri project team, the Galeri project website.
[41] Trilinos public git repository, Web, 2022.
[42] The Zoltan2 project team, the Zoltan2 project website.