A highly scalable asynchronous implementation of Balancing Domain Decomposition by Constraints

Santiago Badia, Alberto F. Martín and Javier Príncipe

Centre Internacional de Mètodes Numèrics a l'Enginyeria (CIMNE)
Castelldefels, Spain

Universitat Politècnica de Catalunya
Barcelona, Spain

Sparse Days Meeting 2014
CERFACS, Toulouse, France
5-6 June, 2014

Outline

Outline

1. BDDC preconditioner
2. Overlapped BDDC implementation
3. Scalability analysis (overlapped)
4. Inexact BDDC
5. Scalability analysis (overlapped/inexact)
6. Conclusions and future work

Problem statement

Given a bounded domain Ω and a FE partition $\mathcal{T}$, we build a conforming (nodal) finite element (FE) space, i.e. $V_h \subset H_0^1(\Omega)$.

- **Variational problem**: find $u \in V_h$ such that

$$a(u, v) = (f, v), \qquad \text{for any } v \in V_h,$$

assuming $a(\cdot, \cdot)$ symmetric, coercive (e.g. Laplacian or linear elasticity)

- **Algebraic problem**: Equivalent to find $x \in \mathbb{R}^n$ such that

$$Ax = b,$$

where A is a large and sparse symmetric positive definite matrix

Problem statement

Given a bounded domain Ω and a FE partition $\mathcal{T}$, we build a conforming (nodal) finite element (FE) space, i.e. $V_h \subset H_0^1(\Omega)$.

- **Variational problem**: find $u \in V_h$ such that

$$a(u, v) = (f, v), \qquad \text{for any } v \in V_h,$$

  assuming $a(\cdot, \cdot)$ symmetric, coercive (e.g. Laplacian or linear elasticity)

- **Algebraic problem**: Equivalent to find $x \in \mathbb{R}^n$ such that

$$Ax = b,$$

  where A is a large and sparse symmetric positive definite matrix

Problem statement

Given a bounded domain Ω and a FE partition $\mathcal{T}$, we build a conforming (nodal) finite element (FE) space, i.e. $V_h \subset H_0^1(\Omega)$.

- **Variational problem**: find $u \in V_h$ such that

$$a(u, v) = (f, v), \qquad \text{for any } v \in V_h,$$

  assuming $a(\cdot, \cdot)$ symmetric, coercive (e.g. Laplacian or linear elasticity)

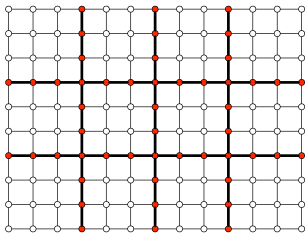
- **Algebraic problem**: Equivalent to find $x \in \mathbb{R}^n$ such that

$$Ax = b,$$

  where A is a large and sparse symmetric positive definite matrix

Motivation:

Efficient exploitation of distributed-memory machines for large scale FE problems $\Rightarrow$ Domain decomposition framework



$\circ$: interior DoFs (I); $\bullet$: interface dofs (Γ)

Interface (Schur complement) problem

- The domain partition induces a block structure

$$Ax = \begin{bmatrix} A_{II} & A_{I\Gamma} \\ A_{\Gamma I} & A_{\Gamma\Gamma} \end{bmatrix} \begin{bmatrix} x_I \\ x_\Gamma \end{bmatrix} = \begin{bmatrix} b_I \\ b_\Gamma \end{bmatrix} = b,$$

where

$$A_{II} = diag\left(A_{II}^{(1)}, A_{II}^{(2)}, \ldots, A_{II}^{(P)}\right)$$

- After the interior correction $[A_{II}^{-1} b_I, 0]$, a reduced system for x_Γ is obtained

$$Sx_\Gamma = g,$$

where $S = A_{\Gamma\Gamma} - A_{\Gamma I} A_{II}^{-1} A_{I\Gamma}$ is the interface *Schur complement*

- **Approach:** Consider a Krylov subspace solver for $Sx_\Gamma = g$
  $\rightarrow$ Preconditioning plays a major role for optimality and scalability
- Alternatively, the preconditioner can be extended to $Ax = f$ (equivalent as soon as A_{II}^{-1} *exactly*)

- The domain partition induces a block structure

$$Ax = \begin{bmatrix} A_{II} & A_{I\Gamma} \\ A_{\Gamma I} & A_{\Gamma\Gamma} \end{bmatrix} \begin{bmatrix} x_I \\ x_\Gamma \end{bmatrix} = \begin{bmatrix} b_I \\ b_\Gamma \end{bmatrix} = b,$$

where

$$A_{II} = diag\left(A_{II}^{(1)}, A_{II}^{(2)}, \ldots, A_{II}^{(P)}\right)$$

- After the interior correction $[A_{II}^{-1} b_I, 0]$, a reduced system for x_Γ is obtained

$$Sx_\Gamma = g,$$

where $S = A_{\Gamma\Gamma} - A_{\Gamma I} A_{II}^{-1} A_{I\Gamma}$ is the interface *Schur complement*

- **Approach:** Consider a Krylov subspace solver for $Sx_\Gamma = g$
  $\rightarrow$ Preconditioning plays a major role for optimality and scalability
- Alternatively, the preconditioner can be extended to $Ax = f$ (equivalent as soon as A_{II}^{-1} *exactly*)

Interface (Schur complement) problem

- The domain partition induces a block structure

$$Ax = \begin{bmatrix} A_{II} & A_{I\Gamma} \\ A_{\Gamma I} & A_{\Gamma\Gamma} \end{bmatrix} \begin{bmatrix} x_I \\ x_\Gamma \end{bmatrix} = \begin{bmatrix} b_I \\ b_\Gamma \end{bmatrix} = b,$$

  where

$$A_{II} = diag\left(A_{II}^{(1)}, A_{II}^{(2)}, \ldots, A_{II}^{(P)}\right)$$

- After the interior correction $[A_{II}^{-1} b_I, 0]$, a reduced system for x_Γ is obtained

$$Sx_\Gamma = g,$$

  where $S = A_{\Gamma\Gamma} - A_{\Gamma I} A_{II}^{-1} A_{I\Gamma}$ is the interface *Schur complement*
- **Approach:** Consider a Krylov subspace solver for $Sx_\Gamma = g$
  → Preconditioning plays a major role for optimality and scalability
- Alternatively, the preconditioner can be extended to $Ax = f$ (equivalent as soon as A_{II}^{-1} *exactly*)

[Dohrmann, Mandel, Cros, Fragakis, Papadrakakis, Le Tallec, Vidrascu, ...]

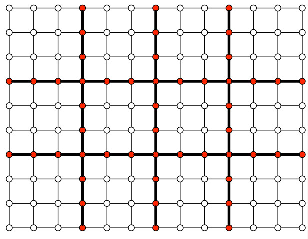
Idea: Solve global problem w/ reduced continuity

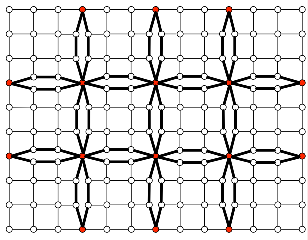
- Replace V_h by $\tilde{V}_h$ (reduced continuity)
- Define the injection $I : \tilde{V}_h \longrightarrow V_h$ weight, comm and add
- Find $\tilde{x}_h \in \tilde{V}_h$ such that:

$$a(\tilde{x}_h, \tilde{v}_h) = \langle I^t r_h, \tilde{v}_h \rangle, \quad \forall \tilde{v}_h \in \tilde{V}_h$$

and obtain $z_h = \mathcal{E} I \tilde{x}_h$, where $z_h = M_{BDDC}^{-1} r_h$

V_h





$\tilde{V}_h$

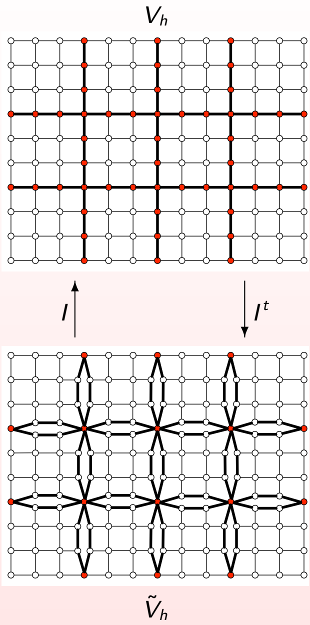
[Dohrmann, Mandel, Cros, Fragakis, Papadrakakis, Le Tallec, Vidrascu, ...]

Idea: Solve global problem w/ reduced continuity

- Replace V_h by $\tilde{V}_h$ (reduced continuity)
- Define the injection $I : \tilde{V}_h \longrightarrow V_h$
  weight, comm and add
- Find $\tilde{x}_h \in \tilde{V}_h$ such that:

$$a(\tilde{x}_h, \tilde{v}_h) = \langle I^t r_h, \tilde{v}_h \rangle, \quad \forall \tilde{v}_h \in \tilde{V}_h$$

and obtain $z_h = \mathcal{E} I \tilde{x}_h$, where $z_h = M_{BDDC}^{-1} r_h$



[Dohrmann, Mandel, Cros, Fragakis, Papadrakakis, Le Tallec, Vidrascu, ...]

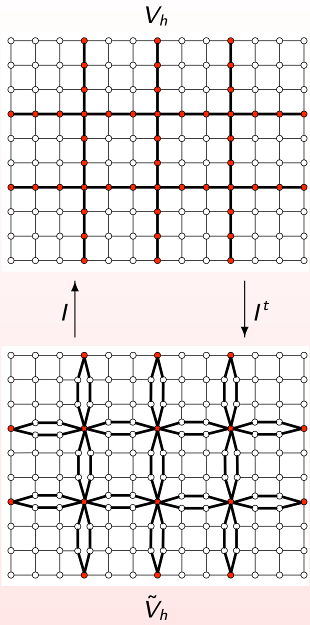
Idea: Solve global problem w/ reduced continuity

- Replace V_h by $\tilde{V}_h$ (reduced continuity)
- Define the injection $I : \tilde{V}_h \longrightarrow V_h$ weight, comm and add
- Find $\tilde{x}_h \in \tilde{V}_h$ such that:

$$a(\tilde{x}_h, \tilde{v}_h) = \langle I^t r_h, \tilde{v}_h \rangle, \quad \forall \tilde{v}_h \in \tilde{V}_h$$

and obtain $z_h = \mathcal{E} I \tilde{x}_h$, where $z_h = M_{BDDC}^{-1} r_h$

- Last correction: $\mathcal{E}$ is the harmonic extension of the boundary values, which implies local Dirichlet solvers



[Dohrmann, Mandel, Cros, Fragakis, Papadrakakis, Le Tallec, Vidrascu, ...]

Idea: Solve global problem w/ reduced continuity

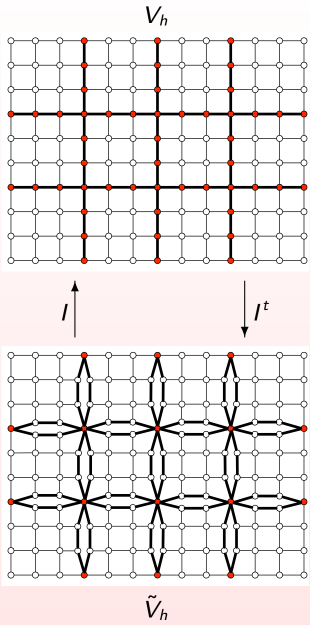
- Alternatively,

  Find $\tilde{x} \in \mathbb{R}^{\tilde{n}}$ such that:

$$\tilde{A}\tilde{x} = I^t r$$

  and obtain $z = \mathcal{E} I \tilde{x}$, where $z = M_{BDDC}^{-1} r$

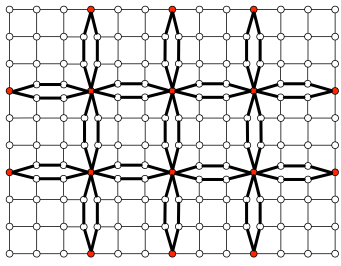
- $\tilde{A}$ is a sub-assembled global matrix (only assembled the red corners in the figure)
- $\mathcal{E} = \begin{bmatrix} 0 & -A_{II}^{-1} A_{I\Gamma} \\ 0 & I_\Gamma \end{bmatrix}$



- Let $\tilde{V}_h = [\tilde{v}_\circ \ \tilde{v}_\bullet]$ and decompose $\tilde{V}_h$ as

$$\tilde{V}_h = \tilde{V}_F \oplus \tilde{V}_C, \ \text{with} \begin{cases} \tilde{V}_F = [\tilde{v}_\circ \ 0] \\ \tilde{V}_C \perp_{\tilde{A}} \tilde{V}_F \end{cases}$$

- Now, problem split into fine-grid ($\tilde{x}_F$) and coarse-grid ($\tilde{x}_C$) correction

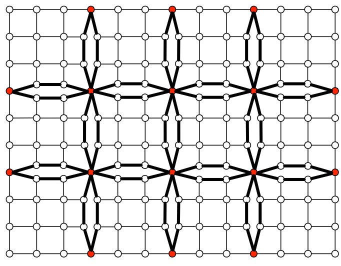


$\tilde{V}_h$

- Let $\tilde{V}_h = [\tilde{v}_{\circ}\ \tilde{v}_{\bullet}]$ and decompose $\tilde{V}_h$ as

$$\tilde{V}_h = \tilde{V}_F \oplus \tilde{V}_C, \text{ with} \begin{cases} \tilde{V}_F = [\tilde{v}_{\circ}\ 0] \\ \tilde{V}_C \perp_{\tilde{A}} \tilde{V}_F \end{cases}$$

- Now, problem split into fine-grid ($\tilde{x}_F$) and coarse-grid ($\tilde{x}_C$) correction



$\tilde{V}_h$

Balancing domain decomposition by constraints (BDDC)

- Let $\tilde{V}_h = [\tilde{v}_\circ \ \tilde{v}_\bullet]$ and decompose $\tilde{V}_h$ as

$$\tilde{V}_h = \tilde{V}_F \oplus \tilde{V}_C, \text{ with } \begin{cases} \tilde{V}_F = [\tilde{v}_\circ \ 0] \\ \tilde{V}_C \perp_{\tilde{A}} \tilde{V}_F \end{cases}$$

- Now, problem split into fine-grid $(\tilde{x}_F)$ and coarse-grid $(\tilde{x}_C)$ correction

Fine-grid correction $(\tilde{x}_F)$

- Find $\tilde{x}_F \in \mathbb{R}^{\tilde{n}}$ such that

$$\tilde{A}\tilde{x}_F = I^t r$$

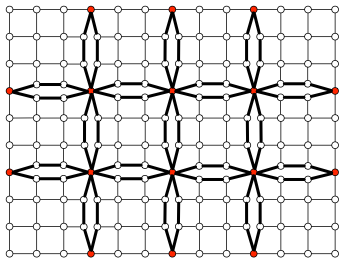
constrained to $(\tilde{x}_F)_\bullet = 0$

- Equivalent to P independent problems

Find $\tilde{x}_F^{(i)} \in \mathbb{R}^{\tilde{n}^{(i)}}$ such that

$$A^{(i)}\tilde{x}_F^{(i)} = I_i^t r$$

constrained to $(\tilde{x}_F^{(i)})_\bullet = 0$



$\tilde{V}_h$

- Let $\tilde{V}_h = [\tilde{v}_{\circ} \ \tilde{v}_{\bullet}]$ and decompose $\tilde{V}_h$ as

$$\tilde{V}_h = \tilde{V}_F \oplus \tilde{V}_C, \text{ with } \begin{cases} \tilde{V}_F = [\tilde{v}_{\circ} \ 0] \\ \tilde{V}_C \perp_{\tilde{A}} \tilde{V}_F \end{cases}$$

- Now, problem split into fine-grid ($\tilde{x}_F$) and coarse-grid ($\tilde{x}_C$) correction

Coarse-grid correction ($\tilde{x}_C$)

Computation of $\tilde{V}_C = \mathrm{span}\{\Phi_1, \Phi_2, \ldots, \Phi_{n_C}\}$

- Find $\Phi \in \mathbb{R}^{\tilde{n} \times n_C}$ such that

$$\tilde{A}\tilde{\Phi} = 0$$

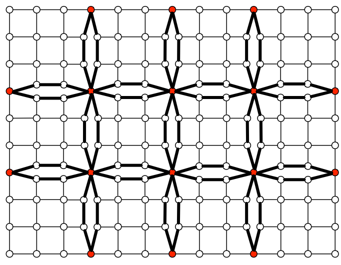
  constrained to $\Phi_{\bullet} = I$

- Equivalent to P independent problems

  Find $\Phi^{(i)} \in \mathbb{R}^{\tilde{n} \times n_C^{(i)}}$ such that

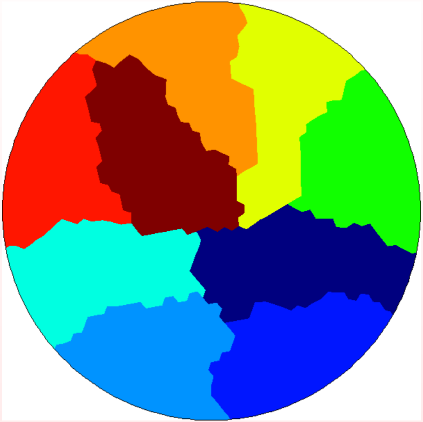
$$A^{(i)}\Phi^{(i)} = 0$$

  constrained to $\Phi_{\bullet}^{(i)} = I$

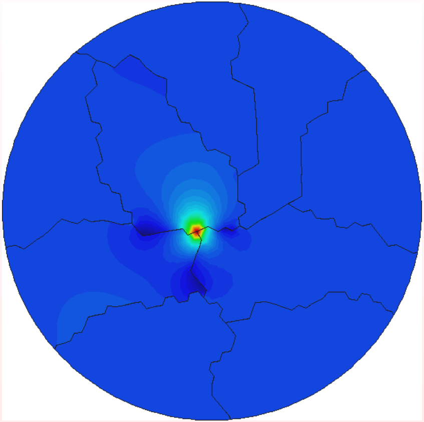


$\tilde{V}_h$

Balancing domain decomposition by constraints (BDDC)



Circle domain partitioned into 9 subdomains



Φ_j ($\tilde{V}_C$'s basis vector)

Balancing domain decomposition by constraints (BDDC)

- Let $\tilde{V}_h = [\tilde{v}_\circ \ \tilde{v}_\bullet]$ and decompose $\tilde{V}_h$ as

$$\tilde{V}_h = \tilde{V}_F \oplus \tilde{V}_C, \ \text{with} \begin{cases} \tilde{V}_F = [\tilde{v}_\circ \ 0] \\ \tilde{V}_C \perp_{\tilde{A}} \tilde{V}_F \end{cases}$$

- Now, problem split into fine-grid ($\tilde{x}_F$) and coarse-grid ($\tilde{x}_C$) correction

Coarse-grid correction ($\tilde{x}_C$)

Assembly and solution of coarse-grid problem

$$A_C = \text{assembly}(A_C^{(i)}) = \text{assembly}(\Phi^t A^{(i)} \Phi), \qquad \text{Solve } A_C \alpha_c = \Phi^t I^t r, \qquad \tilde{x}_C = \Phi \alpha_C$$

coarse-grid problem is

- Global, i.e. couples all subdomains
- But much smaller than S (size $\mathrm{n_C}$)
- Potential loss of parallel efficiency with P

Coarse dofs definition

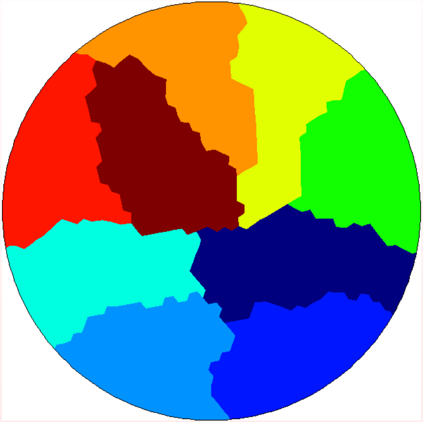
Key aspect: Selection of coarse dofs, i.e. continuity among subdomains

Properties of BDDC preconditioner

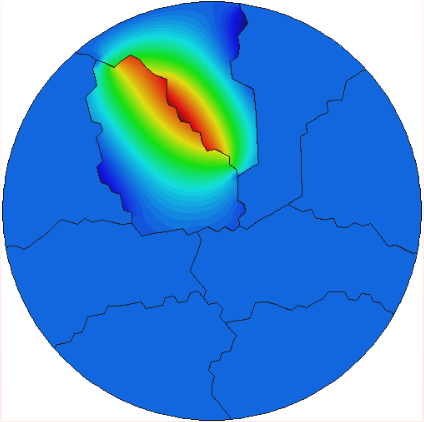
- Optimality ($\kappa(M^{-1}S)$ bounded by a constant for fixed N/P and $\uparrow P$)
- $N/P = (H/h)^d$ large in practice (e.g. $\mathcal{O}(10^4)$ for sparse direct solvers)
- In general, BDDC(ce) and BDDC(cef) require much less iterations in $3D$
- But at the expense of a more costly coarse-grid problem

Coarse dofs vs. $\kappa(M^{-1}S)$:	$d = 2$	$d = 3$
Continuity on corners	$\left[1 + d^{-1}\log^2\left(\frac{N}{P}\right)\right]$	$\frac{N}{P}\left[1 + d^{-1}\log^2\left(\frac{N}{P}\right)\right]$
Continuity of mean value on edges too	$\left[1 + d^{-1}\log^2\left(\frac{N}{P}\right)\right]$	$\left[1 + d^{-1}\log^2\left(\frac{N}{P}\right)\right]$
Continuity of mean value on faces too	-	$\left[1 + d^{-1}\log^2\left(\frac{N}{P}\right)\right]$

Balancing domain decomposition by constraints (BDDC)



Circle domain partitioned into 9 subdomains



Φ_j ($\tilde{V}_C$'s basis vector)

Good features of BDDC towards extreme scales

BDDC has some salient properties that make it an excellent candidate for extreme scale solver design:

1. The method allows for a (mathematically supported) extremely aggressive coarsening ($10^5 - 10^6$ size reduction between fine/coarse level)
2. The coarse matrix has a similar sparsity as the original matrix
3. Coarse and local components can be computed in a parallel (additive) way
4. Local (constrained) Neumann and coarse solvers can be solved in an inexact way (AMG-cycle instead of sparse direct solvers)
5. A multilevel extension of the method is possible (for extreme core counts)

Good features of BDDC towards extreme scales

BDDC has some salient properties that make it an excellent candidate for extreme scale solver design:

1. The method allows for a (mathematically supported) extremely aggressive coarsening ($10^5 - 10^6$ size reduction between fine/coarse level)
2. The coarse matrix has a similar sparsity as the original matrix
3. Coarse and local components can be computed in a parallel (additive) way
4. Local (constrained) Neumann and coarse solvers can be solved in an inexact way (AMG-cycle instead of sparse direct solvers)
5. A multilevel extension of the method is possible (for extreme core counts)

Good features of BDDC towards extreme scales

BDDC has some salient properties that make it an excellent candidate for extreme scale solver design:

1. The method allows for a (mathematically supported) extremely aggressive coarsening ($10^5 - 10^6$ size reduction between fine/coarse level)
2. The coarse matrix has a similar sparsity as the original matrix
3. Coarse and local components can be computed in a parallel (additive) way
4. Local (constrained) Neumann and coarse solvers can be solved in an inexact way (AMG-cycle instead of sparse direct solvers)
5. A multilevel extension of the method is possible (for extreme core counts)

Good features of BDDC towards extreme scales

BDDC has some salient properties that make it an excellent candidate for extreme scale solver design:

1. The method allows for a (mathematically supported) extremely aggressive coarsening ($10^5 - 10^6$ size reduction between fine/coarse level)
2. The coarse matrix has a similar sparsity as the original matrix
3. Coarse and local components can be computed in a parallel (additive) way
4. Local (constrained) Neumann and coarse solvers can be solved in an inexact way (AMG-cycle instead of sparse direct solvers)
5. A multilevel extension of the method is possible (for extreme core counts)

Good features of BDDC towards extreme scales

BDDC has some salient properties that make it an excellent candidate for extreme scale solver design:

1. The method allows for a (mathematically supported) extremely aggressive coarsening ($10^5 - 10^6$ size reduction between fine/coarse level)
2. The coarse matrix has a similar sparsity as the original matrix
3. Coarse and local components can be computed in a parallel (additive) way
4. Local (constrained) Neumann and coarse solvers can be solved in an inexact way (AMG-cycle instead of sparse direct solvers)
5. A multilevel extension of the method is possible (for extreme core counts)

Good features of BDDC towards extreme scales

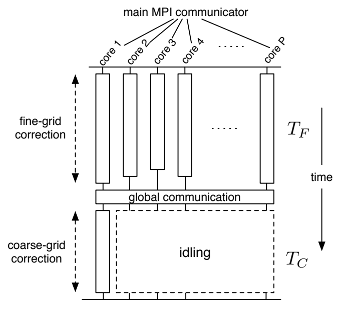
BDDC has some salient properties that make it an excellent candidate for extreme scale solver design:

1. The method allows for a (mathematically supported) extremely aggressive coarsening ($10^5 - 10^6$ size reduction between fine/coarse level)
2. The coarse matrix has a similar sparsity as the original matrix
3. Coarse and local components can be computed in a parallel (additive) way
4. Local (constrained) Neumann and coarse solvers can be solved in an inexact way (AMG-cycle instead of sparse direct solvers)
5. A multilevel extension of the method is possible (for extreme core counts)

- (1)-(2) always exploited in BDDC implementations
- Let us see how to exploit (3), in order to reduce synchronization and boost scalability (overlapped implementation)

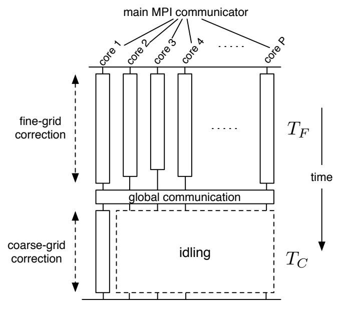
Outline

Typical parallel implementation
(e.g., PETSc. FreeFEM, BDDCML)



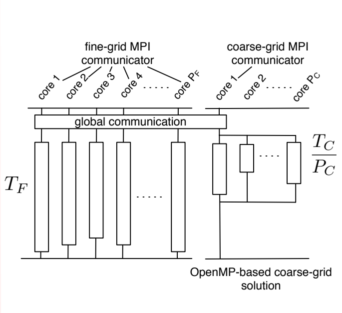
- All MPI tasks have f-g duties and one/several have also c-g duties
- Computation of f-g and c-g correction is serialized (but they are independent!)
- T_C grows as $O(P^2)$ and mem as $\mathcal{O}(P^{\frac{4}{3}})$
  $\rightarrow$ becomes a bottleneck with P
  $\rightarrow$ mem per core rapidly exceeded
- Parallel coarse solvers / multilevel extensions reduce this effect

Typical parallel implementation
(e.g., PETSc. FreeFEM, BDDCML)



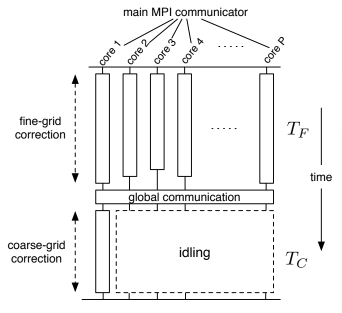
- All MPI tasks have f-g duties and one/several have also c-g duties
- Computation of f-g and c-g correction is serialized (but they are independent!)
- T_C grows as $O(P^2)$ and mem as $\mathcal{O}(P^{\frac{4}{3}})$
  → becomes a bottleneck with P
  → mem per core rapidly exceeded
- Parallel coarse solvers / multilevel extensions reduce this effect

Highly-scalable parallel implementation
Overlapping of fine-grid/coarse-grid duties

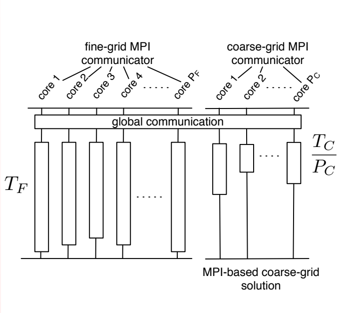


- MPI tasks have either f-g duties or c-g duties (but not both)
- Computation of f-g and c-g correction can be overlapped in time (asynchronous)
- Full node(s) resources (memory and cores) can be devoted to coarse-grid duties
- MPI-based or OpenMP-based (this work) solutions are possible for c-g correction

Typical parallel implementation (e.g., PETSc)



Highly-scalable parallel implementation Overlapping of fine-grid/coarse-grid duties



- All MPI tasks have f-g duties and one/several have also c-g duties
- Computation of f-g and c-g correction is serialized (but they are independent!)
- T_C grows as $O(P^2)$ and mem as $\mathcal{O}(P^{\frac{4}{3}})$
  $\rightarrow$ becomes a bottleneck with P
  $\rightarrow$ mem per core rapidly exceeded
- Parallel coarse solvers / multilevel extensions reduce this effect

- MPI tasks have either f-g duties or c-g duties (but not both)
- Computation of f-g and c-g correction can be overlapped in time (asynchronous)
- Full node(s) resources (memory and cores) can be devoted to coarse-grid duties
- MPI-based or OpenMP-based (this work) solutions are possible for c-g correction

BDDC-PCG basic building blocks

Solve $Ax = b$ via BDDC-PCG

Schur complement set-up (S)
Precond set-up (M_{BDDC})
$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
call PCG(S,M_{BDDC},g,x_Γ)
$x_I := A_{II}^{-1}(b_I - A_{I\Gamma} x_\Gamma)$

PCG

$r_0 := g - S x_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
  $s_{j+1} = S p_j$
  ...
  $z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
  ...
end for

Fine-grid tasks	Coarse-grid task

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble($A_C^{(i)}$)
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble($A_C^{(i)}$)
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble($A_C^{(i)}$)
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble($A_C^{(i)}$)
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact$(G_{A_F^{(i)}})$ $\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization$(G_{A_{II}^{(i)}})$

Schur set-up (numeric)

Numerical factorization$(A_{II}^{(i)})$

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization$(G_{A_F^{(i)}})$
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization$(A_F^{(i)})$
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C := \text{assemble}(A_C^{(i)})$
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C := \text{assemble}(A_C^{(i)})$
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact$(G_{A_F^{(i)}})$ $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact$(G_{A_{II}^{(i)}})$ $\mathcal{O}(n_i^{\frac{4}{3}})$ Num fact$(A_F^{(i)})$ $\mathcal{O}(n_i^2)$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization$(G_{A_{II}^{(i)}})$

Schur set-up (numeric)

Numerical factorization$(A_{II}^{(i)})$

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization$(G_{A_F^{(i)}})$
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization$(A_F^{(i)})$
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble$(A_C^{(i)})$
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$ Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble($A_C^{(i)}$)
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$ Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$ $A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble($A_C^{(i)}$)
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$ Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$ $A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$
Gather $A_C^{(i)}$ GC	

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble($A_C^{(i)}$)
Numerical factorization(A_C)

Fine-grid tasks		Coarse-grid task	
Identify local coarse DoFs			
Construct G_{A_C} GC			
Symb fact($G_{A_F^{(i)}}$)	$\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C})	$\mathcal{O}(P^{\frac{4}{3}})$
Symb fact($G_{A_{II}^{(i)}}$)	$\mathcal{O}(n_i^{\frac{4}{3}})$		
Num fact($A_F^{(i)}$)	$\mathcal{O}(n_i^2)$		
Compute Φ_i	$\mathcal{O}(n_i^{\frac{4}{3}})$		
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$			
Gather $A_C^{(i)}$ GC			
		$A_C :=$ assble($A_C^{(i)}$)	

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble($A_C^{(i)}$)
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$
Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$	
Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$	
Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$	
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	
Gather $A_C^{(i)}$ GC	
	$A_C :=$ assble($A_C^{(i)}$)
	Num fact(A_C) $\mathcal{O}(P^2)$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble($A_C^{(i)}$)
Numerical factorization(A_C)

Fine-grid tasks		Coarse-grid task	
Identify local coarse DoFs			
Construct G_{A_C} GC			
Symb fact($G_{A_F^{(i)}}$)	$\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C})	$\mathcal{O}(P^{\frac{4}{3}})$
Symb fact($G_{A_{II}^{(i)}}$)	$\mathcal{O}(n_i^{\frac{4}{3}})$		
Num fact($A_F^{(i)}$)	$\mathcal{O}(n_i^2)$		
Compute Φ_i	$\mathcal{O}(n_i^{\frac{4}{3}})$		
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$			
Gather $A_C^{(i)}$ GC			
Num fact($A_{II}^{(i)}$)	$\mathcal{O}(n_i^2)$	$A_C :=$ assble($A_C^{(i)}$)	
		Num fact(A_C)	$\mathcal{O}(P^2)$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

Schur set-up (symbolic)

Symbolic factorization($G_{A_{II}^{(i)}}$)

Schur set-up (numeric)

Numerical factorization($A_{II}^{(i)}$)

BDDC set-up (symbolic)

Identify local coarse DoFs
Symbolic factorization($G_{A_F^{(i)}}$)
Construct G_{A_C}
Symbolic factorization (G_{A_C})

BDDC set-up (numeric)

Numerical factorization($A_F^{(i)}$)
Compute Φ_i
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$
Gather $A_C^{(i)}$
$A_C :=$ assemble($A_C^{(i)}$)
Numerical factorization(A_C)

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$
Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$	
Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$	
Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$	
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	
Gather $A_C^{(i)}$ GC	
Num fact($A_{II}^{(i)}$) $\mathcal{O}(n_i^2)$	$A_C :=$ assble($A_C^{(i)}$)
$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$ $\mathcal{O}(n_i^{\frac{4}{3}})$	Num fact(A_C) $\mathcal{O}(P^2)$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - S x_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
$s_{j+1} = S p_j$
$\ldots$
$z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
$\ldots$
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C :=$ assemble($r_C^{(i)}$)
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i(s_F^{(i)} + s_C^{(i)})$

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$ Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$ $A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$
Gather $A_C^{(i)}$ GC	
Num fact($A_{II}^{(i)}$) $\mathcal{O}(n_i^2)$ $g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r_0 := g - Sx_\Gamma$ $\mathcal{O}(n_i^{\frac{4}{3}})$	$A_C :=$ assble($A_C^{(i)}$) Num fact(A_C) $\mathcal{O}(P^2)$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - Sx_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
  $s_{j+1} = Sp_j$
  ...
  $z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
  ...
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C :=$ assemble($r_C^{(i)}$)
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i(s_F^{(i)} + s_C^{(i)})$

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$
Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$	
Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$	
Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$	
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	
Gather $A_C^{(i)}$ GC	
Num fact($A_{II}^{(i)}$) $\mathcal{O}(n_i^2)$	$A_C :=$ assble($A_C^{(i)}$)
$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$ $\mathcal{O}(n_i^{\frac{4}{3}})$	Num fact(A_C) $\mathcal{O}(P^2)$
$r_0 := g - Sx_\Gamma$ $\mathcal{O}(n_i^{\frac{4}{3}})$	
$r^{(i)} := I_i^t r$ LC	

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - Sx_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
$s_{j+1} = Sp_j$
...
$z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
...
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C :=$ assemble($r_C^{(i)}$)
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i(s_F^{(i)} + s_C^{(i)})$

Fine-grid tasks		Coarse-grid task	
Identify local coarse DoFs			
Construct G_{A_C} GC			
Symb fact($G_{A_F^{(i)}}$)	$\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C})	$\mathcal{O}(P^{\frac{4}{3}})$
Symb fact($G_{A_{II}^{(i)}}$)	$\mathcal{O}(n_i^{\frac{4}{3}})$		
Num fact($A_F^{(i)}$)	$\mathcal{O}(n_i^2)$		
Compute Φ_i	$\mathcal{O}(n_i^{\frac{4}{3}})$		
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$			
Gather $A_C^{(i)}$ GC			
Num fact($A_{II}^{(i)}$)	$\mathcal{O}(n_i^2)$	$A_C :=$ assble($A_C^{(i)}$)	
$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$	$\mathcal{O}(n_i^{\frac{4}{3}})$	Num fact(A_C)	$\mathcal{O}(P^2)$
$r_0 := g - S x_\Gamma$	$\mathcal{O}(n_i^{\frac{4}{3}})$		
$r^{(i)} := I_i^t r$	LC		
$r_C^{(i)} := \Phi_i^t r^{(i)}$			

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - S x_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
  $s_{j+1} = S p_j$
  ...
  $z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
  ...
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C :=$ assemble($r_C^{(i)}$)
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i (s_F^{(i)} + s_C^{(i)})$

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$ Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$ $A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$
Gather $A_C^{(i)}$ GC	
Num fact($A_{II}^{(i)}$) $\mathcal{O}(n_i^2)$ $g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r_0 := g - S x_\Gamma$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r^{(i)} := I_i^t r$ LC $r_C^{(i)} := \Phi_i^t r^{(i)}$	$A_C :=$ assble($A_C^{(i)}$) Num fact(A_C) $\mathcal{O}(P^2)$
Gather $r_C^{(i)}$ GC	

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - S x_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
  $s_{j+1} = S p_j$
  ...
  $z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
  ...
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C :=$ assemble($r_C^{(i)}$)
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i(s_F^{(i)} + s_C^{(i)})$

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$ Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$ $A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$
Gather $A_C^{(i)}$ GC	
Num fact($A_{II}^{(i)}$) $\mathcal{O}(n_i^2)$ $g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r_0 := g - S x_\Gamma$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r^{(i)} := I_i^t r$ LC $r_C^{(i)} := \Phi_i^t r^{(i)}$	$A_C :=$ assble($A_C^{(i)}$) Num fact(A_C) $\mathcal{O}(P^2)$
Gather $r_C^{(i)}$ GC	
	$r_C :=$ assble($r_C^{(i)}$)

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - S x_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
  $s_{j+1} = S p_j$
  ...
  $z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
  ...
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C :=$ assemble($r_C^{(i)}$)
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i (s_F^{(i)} + s_C^{(i)})$

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$ Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$ $A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$
Gather $A_C^{(i)}$ GC	
Num fact($A_{II}^{(i)}$) $\mathcal{O}(n_i^2)$ $g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r_0 := g - S x_\Gamma$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r^{(i)} := I_i^t r$ LC $r_C^{(i)} := \Phi_i^t r^{(i)}$	$A_C :=$ assble($A_C^{(i)}$) Num fact(A_C) $\mathcal{O}(P^2)$
Gather $r_C^{(i)}$ GC	
	$r_C :=$ assble($r_C^{(i)}$) Solve $A_C z_C = r_C$ $\mathcal{O}(P^{\frac{4}{3}})$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - S x_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
  $s_{j+1} = S p_j$
  ...
  $z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
  ...
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C :=$ assemble($r_C^{(i)}$)
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i (s_F^{(i)} + s_C^{(i)})$

Fine-grid tasks		Coarse-grid task	
Identify local coarse DoFs			
Construct G_{A_C} GC			
Symb fact($G_{A_F^{(i)}}$)	$\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C})	$\mathcal{O}(P^{\frac{4}{3}})$
Symb fact($G_{A_{II}^{(i)}}$)	$\mathcal{O}(n_i^{\frac{4}{3}})$		
Num fact($A_F^{(i)}$)	$\mathcal{O}(n_i^2)$		
Compute Φ_i	$\mathcal{O}(n_i^{\frac{4}{3}})$		
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$			
Gather $A_C^{(i)}$ GC			
Num fact($A_{II}^{(i)}$)	$\mathcal{O}(n_i^2)$	$A_C :=$ assble($A_C^{(i)}$)	
$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$	$\mathcal{O}(n_i^{\frac{4}{3}})$	Num fact(A_C)	$\mathcal{O}(P^2)$
$r_0 := g - Sx_\Gamma$	$\mathcal{O}(n_i^{\frac{4}{3}})$		
$r^{(i)} := I_i^t r$	LC		
$r_C^{(i)} := \Phi_i^t r^{(i)}$			
Gather $r_C^{(i)}$ GC			
		$r_C :=$ assble($r_C^{(i)}$)	
Compute $s_F^{(i)}$	$\mathcal{O}(n_i^{\frac{4}{3}})$	Solve $A_C z_C = r_C$	$\mathcal{O}(P^{\frac{4}{3}})$

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - Sx_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
$s_{j+1} = Sp_j$
...
$z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
...
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C :=$ assemble($r_C^{(i)}$)
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i(s_F^{(i)} + s_C^{(i)})$

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$ Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$ $A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$
Gather $A_C^{(i)}$ GC	
Num fact($A_{II}^{(i)}$) $\mathcal{O}(n_i^2)$ $g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r_0 := g - S x_\Gamma$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r^{(i)} := I_i^t r$ LC $r_C^{(i)} := \Phi_i^t r^{(i)}$	$A_C := \text{assble}(A_C^{(i)})$ Num fact(A_C) $\mathcal{O}(P^2)$
Gather $r_C^{(i)}$ GC	
Compute $s_F^{(i)}$ $\mathcal{O}(n_i^{\frac{4}{3}})$	$r_C := \text{assble}(r_C^{(i)})$ Solve $A_C z_C = r_C$ $\mathcal{O}(P^{\frac{4}{3}})$
Scatter z_C into $z_C^{(i)}$ GC	

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - S x_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
  $s_{j+1} = S p_j$
  ...
  $z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
  ...
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C := \text{assemble}(r_C^{(i)})$
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i (s_F^{(i)} + s_C^{(i)})$

Fine-grid tasks	Coarse-grid task
Identify local coarse DoFs	
Construct G_{A_C} GC	
Symb fact($G_{A_F^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Symb fact($G_{A_{II}^{(i)}}$) $\mathcal{O}(n_i^{\frac{4}{3}})$ Num fact($A_F^{(i)}$) $\mathcal{O}(n_i^2)$ Compute Φ_i $\mathcal{O}(n_i^{\frac{4}{3}})$ $A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$	Symb fact(G_{A_C}) $\mathcal{O}(P^{\frac{4}{3}})$
Gather $A_C^{(i)}$ GC	
Num fact($A_{II}^{(i)}$) $\mathcal{O}(n_i^2)$ $g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r_0 := g - S x_\Gamma$ $\mathcal{O}(n_i^{\frac{4}{3}})$ $r^{(i)} := I_i^t r$ LC $r_C^{(i)} := \Phi_i^t r^{(i)}$	$A_C :=$ assble($A_C^{(i)}$) Num fact(A_C) $\mathcal{O}(P^2)$
Gather $r_C^{(i)}$ GC	
Compute $s_F^{(i)}$ $\mathcal{O}(n_i^{\frac{4}{3}})$	$r_C :=$ assble($r_C^{(i)}$) Solve $A_C z_C = r_C$ $\mathcal{O}(P^{\frac{4}{3}})$
Scatter z_C into $z_C^{(i)}$ GC	
$s_C^{(i)} := \Phi_i z_C^{(i)}$	

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - S x_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
  $s_{j+1} = S p_j$
  ...
  $z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
  ...
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C :=$ assemble$(r_C^{(i)})$
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i (s_F^{(i)} + s_C^{(i)})$

Fine-grid tasks		Coarse-grid task	
Identify local coarse DoFs			
Construct G_{A_C} GC			
Symb fact($G_{A_F^{(i)}}$)	$\mathcal{O}(n_i^{\frac{4}{3}})$	Symb fact(G_{A_C})	$\mathcal{O}(P^{\frac{4}{3}})$
Symb fact($G_{A_{II}^{(i)}}$)	$\mathcal{O}(n_i^{\frac{4}{3}})$		
Num fact($A_F^{(i)}$)	$\mathcal{O}(n_i^2)$		
Compute Φ_i	$\mathcal{O}(n_i^{\frac{4}{3}})$		
$A_C^{(i)} := \Phi_i^t A^{(i)} \Phi_i$			
Gather $A_C^{(i)}$ GC			
Num fact($A_{II}^{(i)}$)	$\mathcal{O}(n_i^2)$	$A_C :=$ assble($A_C^{(i)}$)	
$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$	$\mathcal{O}(n_i^{\frac{4}{3}})$	Num fact(A_C)	$\mathcal{O}(P^2)$
$r_0 := g - S x_\Gamma$	$\mathcal{O}(n_i^{\frac{4}{3}})$		
$r^{(i)} := I_i^t r$	LC		
$r_C^{(i)} := \Phi_i^t r^{(i)}$			
Gather $r_C^{(i)}$ GC			
		$r_C :=$ assble($r_C^{(i)}$)	
Compute $s_F^{(i)}$	$\mathcal{O}(n_i^{\frac{4}{3}})$	Solve $A_C z_C = r_C$	$\mathcal{O}(P^{\frac{4}{3}})$
Scatter z_C into $z_C^{(i)}$ GC			
$s_C^{(i)} := \Phi_i z_C^{(i)}$			
$z^{(i)} := I_i(s_F^{(i)} + s_C^{(i)})$	LC		

LC: local communication (nearest neighbours)
GC: global communication (gather or scatter)

PCG

$g := b_\Gamma - A_{\Gamma I} A_{II}^{-1} b_I$
$r_0 := g - S x_\Gamma$
$z_0 := M_{\mathrm{BDDC}}^{-1} r_0$
$p_0 := z_0$
for $j = 0, \ldots,$ till CONV **do**
  $s_{j+1} = S p_j$
  ...
  $z_{j+1} := M_{\mathrm{BDDC}}^{-1} r_{j+1}$
  ...
end for

BDDC application

$r^{(i)} := I_i^t r$
Compute $s_F^{(i)}$
$r_C^{(i)} := \Phi_i^t r^{(i)}$
Gather $r_C^{(i)}$
$r_C :=$ assemble($r_C^{(i)}$)
Solve $A_C z_C = r_C$
Scatter z_C into $z_C^{(i)}$
$s_C^{(i)} := \Phi_i z_C^{(i)}$
$z^{(i)} := I_i(s_F^{(i)} + s_C^{(i)})$

Outline

1. BDDC preconditioner
2. Overlapped BDDC implementation
3. Scalability analysis (overlapped)
4. Inexact BDDC
5. Scalability analysis (overlapped/inexact)
6. Conclusions and future work

FEMPAR (in-house developed HPC software, free software GNU-GPL):

Finite Element Multiphysics PARallel software

- Massively parallel sw for the FE simulation of Multiphysics problems governed by PDEs
- Scalable preconditioning of fully coupled and implicit system via block preconditioning techniques (physics-based preconditioning)
- Scalable preconditioning for one-physics (elliptic) PDEs relies on BDDC, BNN
  $\rightarrow$ hybrid MPI/OpenMP implementation
- Relies on highly-efficient vendor implementations of the dense/sparse BLAS (Intel MKL, IBM ESSL, etc.)
- Provides interfaces to external multi-threaded sparse direct solvers (PARDISO, HSL_MA87, etc.) and serial AMG preconditioners (HSL_MI20)

FEMPAR (in-house developed HPC software, free software GNU-GPL):
Finite Element Multiphysics PARallel software

- Massively parallel sw for the FE simulation of Multiphysics problems governed by PDEs
- Scalable preconditioning of fully coupled and implicit system via block preconditioning techniques (physics-based preconditioning)
- Scalable preconditioning for one-physics (elliptic) PDEs relies on BDDC, BNN
  $\rightarrow$ hybrid MPI/OpenMP implementation
- Relies on highly-efficient vendor implementations of the dense/sparse BLAS (Intel MKL, IBM ESSL, etc.)
- Provides interfaces to external multi-threaded sparse direct solvers (PARDISO, HSL_MA87, etc.) and serial AMG preconditioners (HSL_MI20)

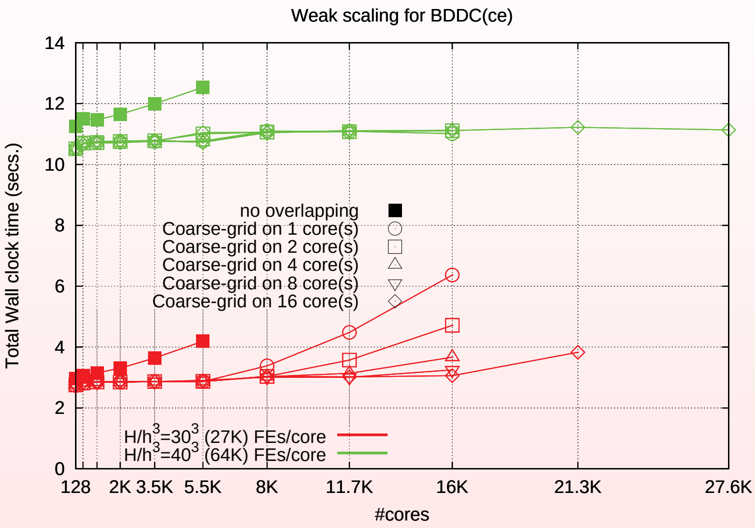
Weak scalability for 3D Poisson

Target machine: HELIOS@IFERC-CSC
4,410 bullx B510 compute blades (2 Intel Xeon E5-2680 8-core CPUs; 64GB)

- Target problem: $-\Delta u = f$ on $\overline{\Omega} = [0,2] \times [0,1] \times [0,1]$
- Uniform global mesh of hexahedral Q1 finite elements
- Uniform partition into rectangular grids of $4m \times 2m \times 2m$ cubic local meshes
- $m = 2^3, 3^3 \ldots, 12^3$ blades $(8, 432, \ldots, 27648$ cores) devoted to fine-grid duties
- Entire 16-core blade devoted to coarse-grid duties (multi-threaded PARDISO)
- Direct solution (PARDISO) of Dirichlet, Neumann, and coarse-grid corrections
- Gradually larger fixed local problem sizes $\frac{H}{h} = 30^3, 40^3$ FEs/core

BDDC(corners+edges) :: Poisson problem

Weak scaling for BDDC(corners+edges) :: Poisson problem



Outline

Good features of BDDC towards extreme scales

BDDC has some salient properties that make it an excellent candidate for extreme scale solver design:

1. The method allows for a (mathematically supported) extremely aggressive coarsening
2. The coarse matrix has a similar sparsity as the original matrix
3. Coarse and local components can be computed in a parallel (additive) way
4. Local (constrained) Neumann and coarse solvers can be solved in an inexact way
5. A multilevel extension of the method is possible (for extreme core counts)

- (1)-(2)-(3) always exploited in our overlapped BDDC implementations
- Let us see how to exploit (4), in order to boost scalability further and reduce memory requirements (overlapped/inexact implementation)

- The exact (using direct solvers) BDDC preconditioner leads to the most effective preconditioner
- However, also to the most computationally and memory demanding one
- In order to reduce these demands, one may solve only approximately some (or even all) of the internal problems using, e.g., AMG-based solvers
- Numerical analysis says that inexact BDDC preconditioners are also algoritmically scalable [Dohrmann, 2007]
- Benefit has to be viewed in light of future parallel architectures: the most scalable architectures (e.g., IBM BG) will have more limited memory per core
- Further, the coarse solver time increases as P instead of P^2, much less degradation for high core counts

Inexact BDDC

- The exact (using direct solvers) BDDC preconditioner leads to the most effective preconditioner
- However, also to the most computationally and memory demanding one
- In order to reduce these demands, one may solve only approximately some (or even all) of the internal problems using, e.g., AMG-based solvers
- Numerical analysis says that inexact BDDC preconditioners are also algoritmically scalable [Dohrmann, 2007]
- Benefit has to be viewed in light of future parallel architectures: the most scalable architectures (e.g., IBM BG) will have more limited memory per core
- Further, the coarse solver time increases as P instead of P^2, much less degradation for high core counts

Inexact BDDC

- The exact (using direct solvers) BDDC preconditioner leads to the most effective preconditioner
- However, also to the most computationally and memory demanding one
- In order to reduce these demands, one may solve only approximately some (or even all) of the internal problems using, e.g., AMG-based solvers
- Numerical analysis says that inexact BDDC preconditioners are also algoritmically scalable [Dohrmann, 2007]
- Benefit has to be viewed in light of future parallel architectures: the most scalable architectures (e.g., IBM BG) will have more limited memory per core
- Further, the coarse solver time increases as P instead of P^2, much less degradation for high core counts

Inexact BDDC

- The exact (using direct solvers) BDDC preconditioner leads to the most effective preconditioner
- However, also to the most computationally and memory demanding one
- In order to reduce these demands, one may solve only approximately some (or even all) of the internal problems using, e.g., AMG-based solvers
- Numerical analysis says that inexact BDDC preconditioners are also algoritmically scalable [Dohrmann, 2007]
- Benefit has to be viewed in light of future parallel architectures: the most scalable architectures (e.g., IBM BG) will have more limited memory per core
- Further, the coarse solver time increases as P instead of P^2, much less degradation for high core counts

4 different solvers in BDDC:

1. Dirichlet problem: approximate $(A_{II}^{(i)})^{-1}$ in $\mathcal{E} = \begin{bmatrix} 0 & -A_{II}^{-1}A_{I\Gamma} \\ 0 & I_\Gamma \end{bmatrix}$
2. Local Neumann problem: approximate $(A_c^{(i)})^{-1}$, where $A_c^{(i)}$ is the (sub-assembled) local matrix $A^{(i)}$ after eliminating the coarse corner rows/columns
3. Coarse problem: approximate $(A_C)^{-1}$
4. Computation of Φ: approximate $(A_c^{(i)})^{-1}$

From numerical analysis [Dohrmann, 2007]:

- (2)-(3) can be replaced by optimal preconditioners, e.g., AMG-cycle
- (1)-(4) more delicate, additional *null space* preservation required (not true in general)

Key question to be experimentally assessed

Sensivity of the algorithm to every inexact solver?

4 different solvers in BDDC:

1. Dirichlet problem: approximate $(A_{II}^{(i)})^{-1}$ in $\mathcal{E} = \begin{bmatrix} 0 & -A_{II}^{-1} A_{I\Gamma} \\ 0 & I_\Gamma \end{bmatrix}$
2. Local Neumann problem: approximate $(A_c^{(i)})^{-1}$, where $A_c^{(i)}$ is the (sub-assembled) local matrix $A^{(i)}$ after eliminating the coarse corner rows/columns
3. Coarse problem: approximate $(A_C)^{-1}$
4. Computation of Φ: approximate $(A_c^{(i)})^{-1}$

From numerical analysis [Dohrmann, 2007]:

- (2)-(3) can be replaced by optimal preconditioners, e.g., AMG-cycle
- (1)-(4) more delicate, additional *null space* preservation required (not true in general)

Key question to be experimentally assessed

Sensivity of the algorithm to every inexact solver?

4 different solvers in BDDC:

1. Dirichlet problem: approximate $(A_{II}^{(i)})^{-1}$ in $\mathcal{E} = \begin{bmatrix} 0 & -A_{II}^{-1}A_{I\Gamma} \\ 0 & I_\Gamma \end{bmatrix}$
2. Local Neumann problem: approximate $(A_c^{(i)})^{-1}$, where $A_c^{(i)}$ is the (sub-assembled) local matrix $A^{(i)}$ after eliminating the coarse corner rows/columns
3. Coarse problem: approximate $(A_C)^{-1}$
4. Computation of Φ: approximate $(A_c^{(i)})^{-1}$

From numerical analysis [Dohrmann, 2007]:

- (2)-(3) can be replaced by optimal preconditioners, e.g., AMG-cycle
- (1)-(4) more delicate, additional *null space* preservation required (not true in general)

Key question to be experimentally assessed

Sensivity of the algorithm to every inexact solver?

4 different solvers in BDDC:

1. Dirichlet problem: approximate $(A_{II}^{(i)})^{-1}$ in $\mathcal{E} = \begin{bmatrix} 0 & -A_{II}^{-1} A_{I\Gamma} \\ 0 & I_\Gamma \end{bmatrix}$
2. Local Neumann problem: approximate $(A_c^{(i)})^{-1}$, where $A_c^{(i)}$ is the (sub-assembled) local matrix $A^{(i)}$ after eliminating the coarse corner rows/columns
3. Coarse problem: approximate $(A_C)^{-1}$
4. Computation of Φ: approximate $(A_c^{(i)})^{-1}$

From numerical analysis [Dohrmann, 2007]:

- (2)-(3) can be replaced by optimal preconditioners, e.g., AMG-cycle
- (1)-(4) more delicate, additional *null space* preservation required (not true in general)

Key question to be experimentally assessed

Sensivity of the algorithm to every inexact solver?

Inexact BDDC

4 different solvers in BDDC:

1. Dirichlet problem: approximate $(A_{II}^{(i)})^{-1}$ in $\mathcal{E} = \begin{bmatrix} 0 & -A_{II}^{-1} A_{I\Gamma} \\ 0 & I_\Gamma \end{bmatrix}$
2. Local Neumann problem: approximate $(A_c^{(i)})^{-1}$, where $A_c^{(i)}$ is the (sub-assembled) local matrix $A^{(i)}$ after eliminating the coarse corner rows/columns
3. Coarse problem: approximate $(A_C)^{-1}$
4. Computation of Φ: approximate $(A_c^{(i)})^{-1}$

From numerical analysis [Dohrmann, 2007]:

- (2)-(3) can be replaced by optimal preconditioners, e.g., AMG-cycle
- (1)-(4) more delicate, additional *null space* preservation required (not true in general)

Key question to be experimentally assessed

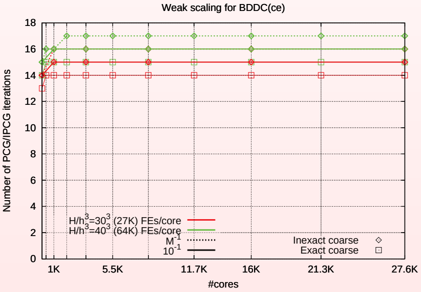
Sensivity of the algorithm to every inexact solver?

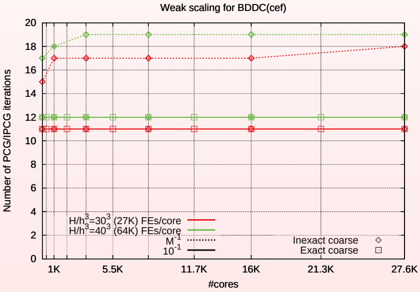
Outline

The effect of approximately solving the internal problems

Coarse-grid problem

- The problem that can harm scalability (couples ALL subdomain)
- Fortunately, it can be highly perturbed without impact in the scalability (AMG-cycle suffices)

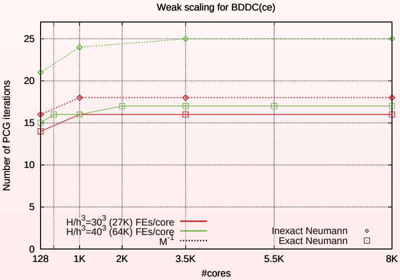


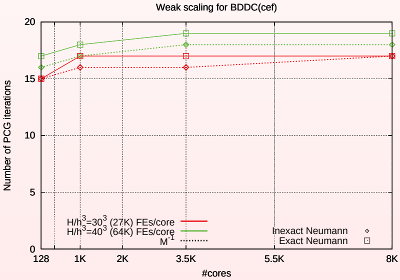


The effect of approximately solving the internal problems

Neumann Problem

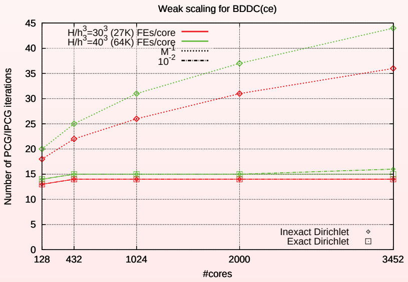
Neumann problem can be highly perturbed without impact in the scalability (AMG-cycle suffices)

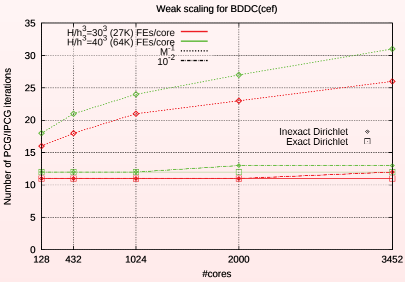




Dirichlet problem

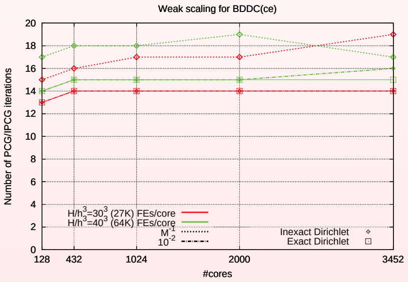
- AMG-cycle wo/ null space preservation (deflation) not algorithmically scalable
- But with loose tolerance enough to make it scalable

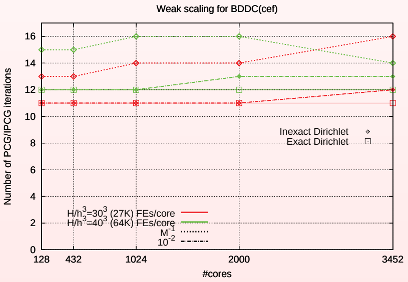




Dirichlet problem

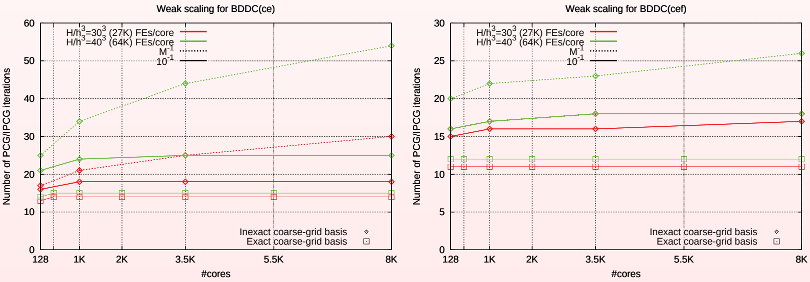
- AMG-cycle w/ null space preservation (deflation) not algorithmically scalable
- But with loose tolerance enough to make it scalable





Coarse-grid basis vectors

- AMG-cycle wo/ null space preservation not algorithmically scalable
- But with loose tolerance enough to make it scalable



A set of strategically selected inexact variants

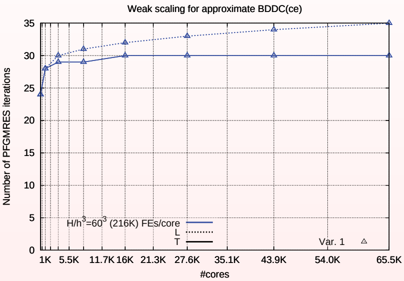
Target machine: JUQUEEN@JSC
28,672 compute nodes (16-core, 64-way threaded IBM PPC A2; 16 GB)

- Target problem: $-\Delta u = f$ on $\overline{\Omega} = [0,2] \times [0,1] \times [0,1]$
- Uniform global mesh of hexahedral Q1 finite elements
- Uniform partition into rectangular grids of $4m \times 2m \times 2m$ cubic local meshes
- $m = 2^3, 3^3 \ldots, 16^3$ nodes $(8, 432, \ldots, 65535$ cores) devoted to fine-grid duties
- Entire node devoted to coarse-grid duties (restricted to only 1 core/GB)
- Gradually larger fixed local problem sizes $\frac{H}{h} = 60^3$ FEs/core

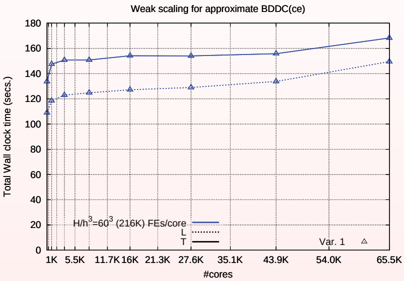
Inexact BDDC(corners+edges) :: Poisson problem

Weak scaling for inexact BDDC(corners+edges) :: Poisson problem

$$\frac{H}{h} = 60 \text{ (216K FEs/core)}$$



# of outer solver iterations



Total time (secs.)

	Outer solver	Φ	Dirichlet	Neumann	Coarse
Var. 1T	FGMRES	PCG-AMG(10^{-1})	PCG-AMG(10^{-4})	AMG(1)	AMG(1)
Var. 1L	FGMRES	PCG-AMG(10^{-1})	PCG-AMG(10^{-2})	AMG(1)	AMG(1)

Memory usage:

- Fine proc's: 538.6MB ($<$ 1GB)
- Coarse proc's (65.5K cores): 392.7MB ($<$ 1GB)

Outline

Conclusions:

- Highly scalable asynchronous implementation of BDDC
- Overlapping of fine-grid and coarse-grid computations
- OpenMP parallelization for coarse-grid problem in the exact case
- Exploitation of AMG-based solvers in the inexact case
- Weakly scalable for many ranges of interest
- Memory limitations clearly improved
- High scalability in a memory constrained environment (JUQUEEN)

Conclusions and future work

Conclusions:

- Highly scalable asynchronous implementation of BDDC
- Overlapping of fine-grid and coarse-grid computations
- OpenMP parallelization for coarse-grid problem in the exact case
- Exploitation of AMG-based solvers in the inexact case
- Weakly scalable for many ranges of interest
- Memory limitations clearly improved
- High scalability in a memory constrained environment (JUQUEEN)

Good features of BDDC towards extreme scales

BDDC has some salient properties that make it an excellent candidate for extreme scale solver design:

1. The method allows for a (mathematically supported) extremely aggressive coarsening
2. The coarse matrix has a similar sparsity as the original matrix
3. Coarse and local components can be computed in a parallel (additive) way
4. Local (constrained) Neumann and coarse solvers can be solved in an inexact way
5. A multilevel extension of the method is possible (for extreme core counts)

- (1)-(2)-(3)-(4) exploited in our inexact/overlapped BDDC implementations
- Next step: Exploit (5), for to boost scalability even further (overlapped/inexact/multilevel implementation)

- S. Badia, A. F. Martín and J. Principe. Enhanced balancing Neumann-Neumann preconditioning in computational fluid and solid mechanics. *International Journal for Numerical Methods in Engineering*. Vol. 96(4), pp. 203-230, 2013.
- S. Badia, A. F. Martín and J. Principe. Implementation and scalability analysis of balancing domain decomposition methods. *Archives of Computational Methods in Engineering*. Vol. 20(3), pp. 239-262, 2013.
- S. Badia, A. F. Martín and J. Principe. A highly scalable parallel implementation of balancing domain decomposition by constraints. *SIAM Journal on Scientific Computing*. Vol. 36(2), pp. C190-C218, 2014.
- S. Badia, A. F. Martín and J. Principe. On the scalability of inexact balancing domain decomposition by constraints with overlapped coarse/fine corrections. In preparation, 2014.

- Preprints available at `http://badia.rmee.upc.edu/sbadia_ar.html`
- COMFUS team: `https://web.cimne.upc.edu/groups/comfus/`

Work funded by the European Research Council under Starting Grant 258443 - COMFUS: Computational Methods for Fusion Technology

erc 2007 - 2012 Years of excellent IDEAS European Research Council