A preconditioning technique based on two-level domain decomposition methods

Duk-Soon Oh

Courant Institute of Mathematical Sciences
New York University

Feburary 16, 2012

Introduction

- Conventional methods for large systems of algebraic equations arising from FDM and FEM
  - Direct methods - expensive to use
  - Iterative methods - may need many iteration counts due to a large condition number
- Domain decomposition methods
  - Provide good preconditioners
  - Combination of direct methods and iterative methods
  - Easy to parallelize

Idea of Domain Decomposition Methods

- Decompose the domain Ω into overlapping or non-overlapping subdomains.
- Assign one or several subdomains to each processor of parallel machine.

In each iteration:

- In each subdomain, solve small local subproblems.
- In addition, solve one small global problem (two-level methods).

Motivation

One-level vs Two-level

- the number of subdomains may effect the efficiency for one-level methods.
- the performance of two-level methods only depends on the size of local subproblems.
- in some cases, e.g., solving nonlinear problems, we can reuse the coarse solver.

Motivation

Conventional two-level methods

- we usually need additional information, e.g., coarse coordinate information.
- we need quite regular meshes.
- it is hard to apply for irregular subdomains.

Alternative Approach

Generalized Dryja, Smith, Widlund (GDSW) coarse space technique

- this technique is based on energy minimizing discrete harmonic extensions.
- it has been applied to many applications
  - almost incompressible elasticity (Dohrmann, Widlund)
  - Reissner-Mindlin plates (Lee)
  - Raviart-Thomas vector fields (Oh)

Alternative Approach

Advantage

- the method can be implemented in an algebraic manner - we do not need any coarse discretization.
- it works well for irregular subdomains and unstructured meshes.
- it has well-established theoretical results, e.g., upper bounds of condition number.

Discrete Harmonic Extension

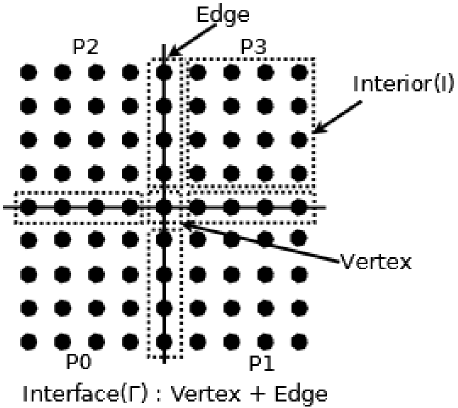
A function $u^{(i)}$ defined on Ω_i is said to be discrete harmonic on Ω_i if

$$A_{II}^{(i)} u_I^{(i)} + A_{I\Gamma}^{(i)} u_\Gamma^{(i)} = 0.$$

$u^{(i)}$ is completely defined by $u_\Gamma^{(i)}$.
The discrete harmonic extension has the minimal energy property.

$$\mathbf{a}(\mathbf{u}, \mathbf{u}) = \min_{\mathbf{v}|_\Gamma = \mathbf{u}_\Gamma} \mathbf{a}(\mathbf{v}, \mathbf{v})$$

Coarse Component



Coarse Component

- R_0 : restriction to coarse space
  - We choose one coarse edge or vertex and give 1 to the nodes on the edge or vertex.
  - We assign 0 to other nodes on the interface.
  - We use the discrete harmonic extension for interior parts.
- A_0 : $R_0 A R_0^T$

We note that this coarse component can be implemented in an algebraic manner. We do not need any coarse discretizations.

Additive Schwarz Perconditioner

Additive Schwarz Method

$$P^{-1} = R_0^T A_0^{-1} R_0 + \sum_{i=1}^{N} R_i^T A_i^{-1} R_i$$

- A_0 : coarse matrix (restriction to the coarse space)
- A_i : local matrix (restriction to extended subdomain Ω_i')
- R_0 : restriction to coarse space
- R_i : restriction to extended subdomain Ω_i'

Restricted Additive Schwarz Perconditioner

Restricted Additive Schwarz Method

$$P^{-1} = R_0^T A_0^{-1} R_0 + \sum_{i=1}^{N} \widetilde{R}_i^T A_i^{-1} R_i$$

- A_0 : coarse matrix (restriction to the coarse space)
- A_i : local matrix (restriction to extended subdomain Ω_i')
- R_0 : restriction to coarse space
- R_i : restriction to extended subdomain Ω_i'
- $\widetilde{R}_i$: restriction to subdomain Ω_i

Implementation

- We implemented the algorithm with Trilinos Ifpack interface.
- Ifpack supports one-level (restricted) additive schwarz preconditioners.
- We only need an Epetra RowMatrix and an Epetra Map to construct the coarse component.

Conclusion

- We can construct an algebraic, parallel and scalable preconditioner with our new coarse space technique.
- We are applying this preconditioner to ice-sheet problems.

Numerical Experiments

1024×1024 $2D$ Laplace equation
1 subdomain per each processor, preconditioned GMRES
local solver : Amesos KLU

Table: total elapsed time in second

# of processors	2	4	8	16	32
one-level method	132.14	69.24	49.47	32.71	24.58
two-level method	175.71	85.39	44.38	23.17	14.39

Table: iteration counts

# of processors	2	4	8	16	32
one-level method	76	93	155	162	183
two-level method	48	62	76	64	67