Using iterative methods for local solves in Asynchronous Schwarz methods.

Pratik Nayak

Joint work with Ichitaro Yamazaki and Hartwig Anzt

Doctoral candidate at Karlsruhe Institute of Technology.

Iterative methods.

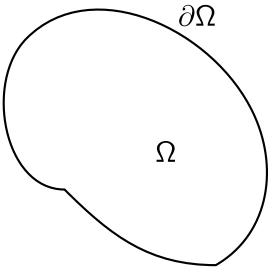


Figure: Generic Domain

Problem:

$$\mathcal{L}x = f \;\; in \; \Omega; \qquad \mathcal{B}x = g \;\; on \; \partial\Omega$$

Linear system:

$$Ax = f$$

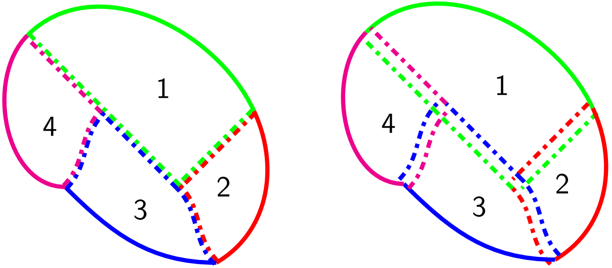
Iterative methods

Stationary iterative method:

$$x^{k+1} = Bx^k + c$$

For convergence, $\rho(B) < 1$.

Domain decomposition methods



(a) Non-overlapping domain (b) Overlapping domain

Figure: Overlapping and non-overlapping domains

Schwarz methods

1. Initially used to prove convergence of the Poisson problem for general domains (Schwarz, 1870). Slow convergence.
2. Gained popularity with parallel computers.
3. Restricted additive Schwarz methods: An improvement of the parallel version of the Schwarz method for faster convergence.

Restricted Additive Schwarz methods

Used widely as a preconditioner:

$$M_{RAS}^{-1} = \sum_{j}^{N} \tilde{R}_j^T A_j^{-1} R_j$$

Group unknowns into subsets:

$$x_j = \tilde{R}_j x, \;\; j = 1, ..., N$$

$\tilde{R}_j$ is the rectangular Restriction matrices which corresponds to a non-overlapping decomposition.

Solve each subset(subdomain) independently and communicate between each "iteration".

Restricted Additive Schwarz methods

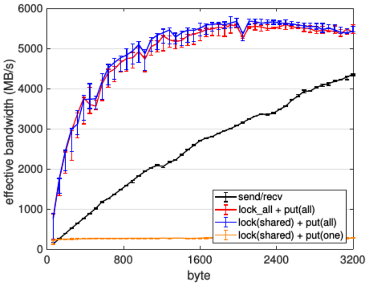
RAS:

$$x_p^{k+1} = x_p^k + \sum_{j}^{N} \tilde{R}_p (R_j f - (R_j A R_j^T)^{-1} R_j x^k)$$

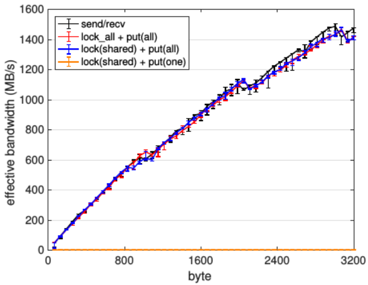
Advantages:

1. Saves communication compared to Additive Schwarz.
2. Reduced iteration count compared to Additive Schwarz.

Synchronous vs Asynchronous



(a) One node



(b) Two nodes

Figure: MPI performance between nodes [1]

[1]: Yamazaki et.al, 2018, To be published

Synchronous vs Asynchronous: ORAS, Laplacian 2D

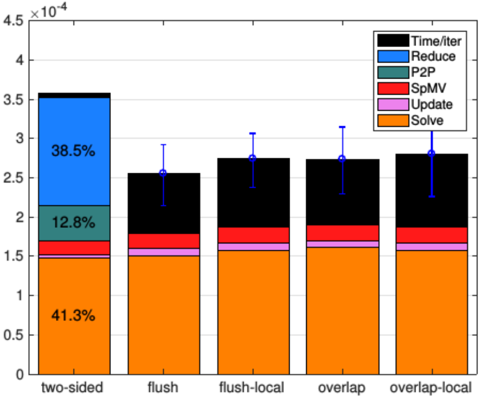
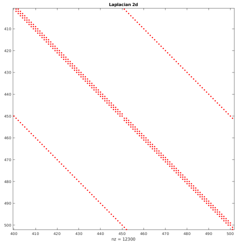


Figure: Optimized Restricted Additive Schwarz timings [1]

[1]: Yamazaki et.al, 2018, To be published

RAS, Laplacian 2D



#processes	Total iteration count		Overall time (s)	
	Synch	Asynch	Synch	Asynch
4	38038	53028	0.08	0.11
8	97948	146865	0.07	0.09
16	167210	279054	0.04	0.05

Table: Matrix: laplacian2d, N = 2500, nnz = 12300

Synchronous vs Asynchronous

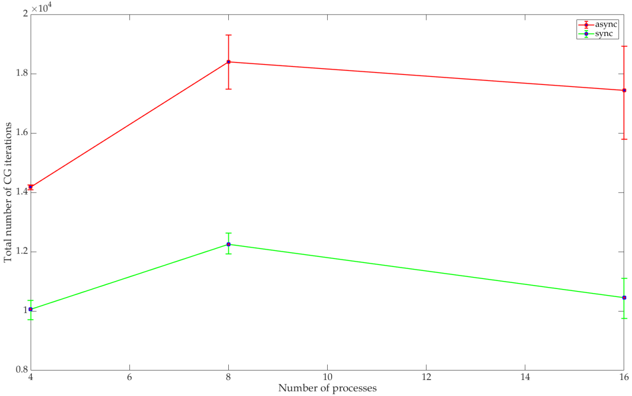
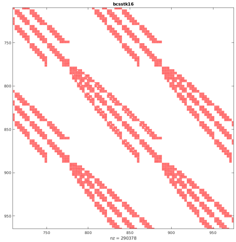


Figure: Laplacian 2d, Total CG iterations

RAS, bcsstk16



#processes	Total iteration count		Overall time (s)	
	Synch	Asynch	Synch	Asynch
4	56100	85194	1.16	1.75
8	99888	168660	0.72	1.02
16	191853	362718	0.62	0.78

Table: Matrix: bcsstk16, N = 4884, nnz = 290378

Synchronous vs Asynchronous

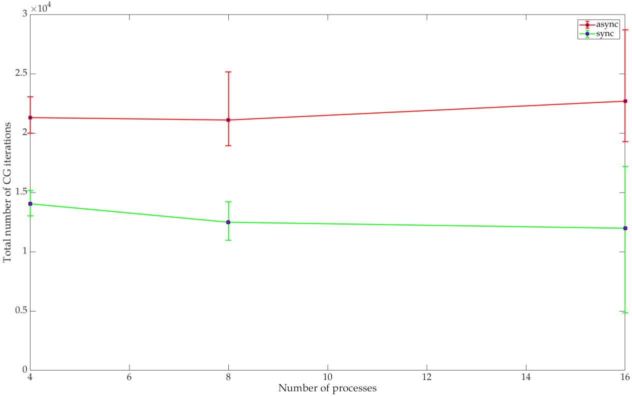
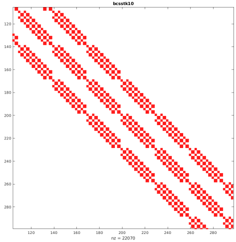


Figure: bcsstk16, Total CG iterations

RAS, bcsstk10



#processes	Total iteration count		Overall time (s)	
	Synch	Asynch	Synch	Asynch
4	165771	157755	0.57	0.27
8	163271	146865	0.10	0.09
16	184128	578383	0.10	0.10

Table: Matrix: bcsstk10, N = 1086, nnz = 22070

Synchronous vs Asynchronous

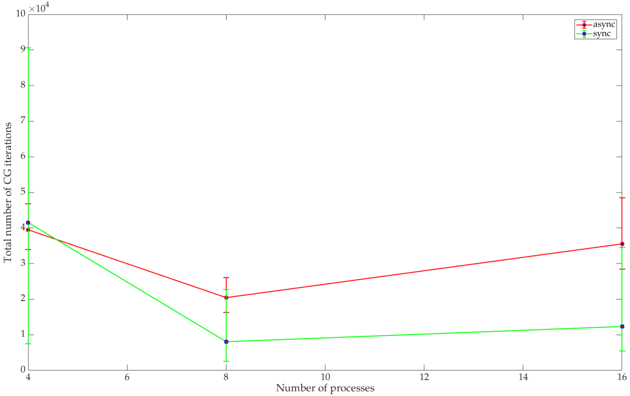


Figure: bcsstk10, Total CG iterations

Synchronous vs Asynchronous

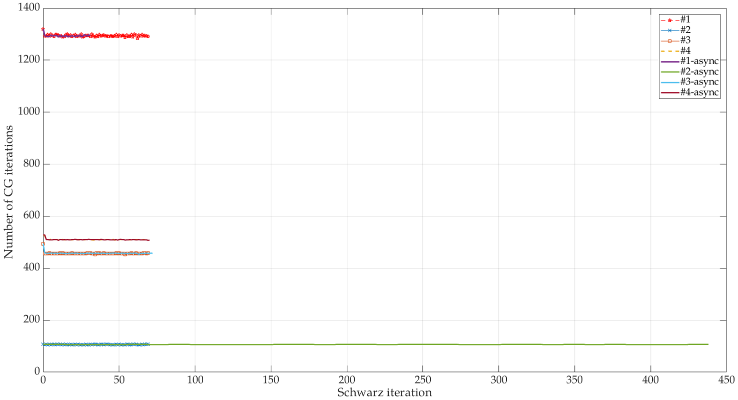


Figure: bcsstk10, Number of CG iterations per Schwarz iteration

How high can we go ?

1. Two tolerances are at play: Global tolerance and local tolerance.
2. Higher the global tolerance, faster the overall convergence.
3. But this is not the case for local tolerance. It depends on:
    1. Condition number of the local subdomain matrices.
    2. Can vary for each subdomain.
    3. Has to be lower than global tolerance (?)

Motivation for Optimized Schwarz

Problem:

$$\mathcal{L}x = f \;\; in \, \Omega; \qquad \mathcal{B}x = g \;\; on \, \partial\Omega$$

Impose artificial boundary condition on interface to accelerate convergence.
Advantages:

1. Faster convergence than RAS.
2. Possible improved performance in the asynchronous case.

Disadvantages:

1. Current theory only convergent for some physical problems with certain conditions (Laplace, Convection-reaction-diffusion)
2. Parameters can be difficult to tune.

Current and Future work

1. General framework for Schwarz decomposition methods.
    1. Using `deal.ii` and `p4est` develop a framework for general finite element solution.
    2. Will have the ability to impose custom artificial boundary conditions.
    3. Easily use adaptive mesh refinement.
    4. "Theoretically" should scale well.
    5. Use Ginkgo as fine-grained solver: Offloading to GPU also possible.
2. Load imbalance characteristics for RAS and possible improvements for asynchronous.