Provable convergence rate for asynchronous Schwarz

Daniel B. Szyld

Temple University, Philadelphia

INRIA Workshop
9 June 2022

Collaborators: Erik Boman (Sandia), Faïcal Chaouqui (Temple), Edmond Chow (Georgia Tech.), Mireille El Haddad (Université Laval, Quebec), Andreas Frommer (Wuppertal), José Garay (Lousiana State), Christian Glusa (Sandia), Frédéric Magoulès (CentraleSupelec, Paris-Saclay), Ichitaro Yamazaki (Sandia)

Thanks to DOE DE-SC0016578
Outline of the talk

- Restricted Additive and Optimized Schwarz methods
- Asynchronous methods
- Some numerical experiments (one- and two-level methods)
- Models of asynchronous methods
- Some convergence theorems
- New convergence results
The general problem

\[ Ax = b \]

\[ \begin{cases} \mathcal{L}(u) = f \text{ in } \Omega \\ \mathcal{C}(u) = g \text{ on } \partial\Omega, \end{cases} \]
The general problem

\[ Ax = b \]

\[
\begin{cases}
\mathcal{L}(u) = f \text{ in } \Omega \\
\mathcal{C}(u) = g \text{ on } \partial \Omega,
\end{cases}
\]

- Domain decomposition (classical Schwarz):
  Solve on subdomains with artificial Dirichlet transmission conditions
General idea of *alternating* Schwarz method: solve on left domain using as Dirichlet data for red line previous approx. of soln. in right domain; solve on right domain using as Dirichlet data for blue line previous approx. of soln. in left domain

Same idea for $q > 2$ subdomains. Go through all $q$ subdomains, then start again, i.e., $s = 1, \ldots, q$

One sweep is very good as a preconditioner for CG or other Krylov subspace methods

[Smith, Bjørstad, Gropp, 1996], [Quarteroni, Valli, 1999], [Toselli, Widlund, 2005], [Mathew, 2008], [Dolean, Jolivet, Nataf, 2015]
More on Schwarz

- Additive/multiplicative Schwarz can be interpreted as Block Jacobi/Gauss-Seidel with overlap. Thus convergence depends on spectral radius (or norm) of iteration operator.

- Restricted Additive Schwarz (RAS): compute with overlap, communicate without overlap.\(^1\)

\(^1\)[Cai, Sarkis, 1999], [Frommer, S, 2001]
For $i = 1, \ldots, q$

$$A_{ii}x_i^{(k+1)} = b_i - \sum_{j \neq i} A_{ij}x_j^{(k)}$$

Not convergent as a solver, double count on overlap
RAS: Keep only restriction of $x_i^{(k+1)}$ to non-overlapping variables
Take-home message 1: Overlap pays off!
Alternating Schwarz as fixed point method

- Can interpret Schwarz iterations as a fixed point map from boundary values to boundary values $v = T v$
Optimized Schwarz Methods (OSM)

- For example for elliptic problems:
  Robin transmission conditions - say $\partial_\nu u(x) + \alpha u(x)$
  Optimal convergence is obtained by optimizing the value of $\alpha$
  (this is called OO0)

- Second order transmission conditions:
  $\frac{\partial u}{\partial \nu} + \alpha u + \beta \frac{\partial^2 u}{\partial \tau^2}$
  (two parameters, called OO2)

- Algebraic version (no restriction on domain shape or PDE)
  (Block Gauss-Seidel with overlap and changing some entries in overlap)

- Optimized Schwarz (or optimized RAS) can be very fast as a solver

[Gander, Halpern, Nataf, 2001], [Japhet, Nataf, Rogier, 2001],
[Dolean, Lanteri, Nataf, 2002], [Côté, Gander, Laayouni, Loisel, 2004],
[Dubois, Gander, Loisel, St-Cyr, S., 2012], [Maday, Magoulès, 2006, 2007],
[Nier, 1998/9] [Dolean, Jolivet, Nataf, 2015]
Algebraic Optimized Schwarz Methods (OSM)

Figure: Square domain, two subdomains, alternating Schwarz
From [Gander, Loisel, S., 2012]
New Architectures, New Paradigms

- Exascale machines, hundreds of thousands of processors
- Communication is usually the bottleneck
- Inner products are prohibitive
- We repeat: For DD, usually outer Krylov, inner RAS / ORAS (preconditioning)
- One idea: Reverse the order, ORAS (or two-level RAS) as outer (solver), Krylov inner (for local problems)
New Architectures, New Paradigms

- Exascale machines, hundreds of thousands of processors
- Communication is usually the bottleneck
- Inner products are prohibitive
- We repeat: For DD, usually outer Krylov, inner RAS / ORAS (preconditioning)
- One idea: Reverse the order, ORAS (or two-level RAS) as outer (solver), Krylov inner (for local problems)
- Another idea: Let us do this asynchronously!
What we do

We do this asynchronously!

For each $s$, repeat until global convergence test satisfied

\[
\begin{align*}
\mathcal{L} (u^{(s)}) &= 0 \text{ in } \Omega^{(s)}, \\
C(u^{(s)}) &= 0 \text{ on } \partial \Omega \cap \partial \Omega^{(s)}, \\
\left( \frac{\partial}{\partial \nu_i^{(s)}} - \Lambda^{(s-)} \right) u^{(s)} &= \left( \frac{\partial}{\partial \nu_i^{(s)}} - \Lambda^{(s-)} \right) u^{(s-1)} \text{ on } \Gamma_i^{(s)}, \\
\left( \frac{\partial}{\partial \nu_r^{(s)}} - \Lambda^{(s+)} \right) u^{(s)} &= \left( \frac{\partial}{\partial \nu_r^{(s)}} - \Lambda^{(s+)} \right) u^{(s+1)} \text{ on } \Gamma_r^{(s)}.
\end{align*}
\]

Each local processor proceeds with whatever boundary information it has, even if it may be repeated.
Stopping criterion also asynchronous.
Algebraic view

In process $i$
- Read $x_j$ ($j \neq i$) (say from shared memory - or from local memory)
- Solve

$$A_{ii}x_i = b_i - \sum_{j \neq i} A_{ij}x_j$$

- Write restricted values of $x_i$ (to shared memory - or other processors)
Algebraic view

In process $i$
- Read $x_j$ ($j \neq i$) (say from shared memory - or from local memory)
- Solve
  \[ A_{ii}x_i = b_i - \sum_{j \neq i} A_{ij}x_j \]
- Write restricted values of $x_i$ (to shared memory - or other processors)
  - No iteration counts
Algebraic view

In process $i$
- Read $x_j$ ($j \neq i$) (say from shared memory - or from local memory)
- Solve
  
  $$A_{ii}x_i = b_i - \sum_{j \neq i} A_{ij}x_j$$

- Write restricted values of $x_i$ (to shared memory - or other processors)
  
  ▶ No iteration counts
  
  ▶ Can tag $x_i$ with wall clock when writing it
Algebraic view

In process $i$
- Read $x_j$ ($j \neq i$) (say from shared memory - or from local memory)
- Solve
  \[ A_{ii}x_i = b_i - \sum_{j \neq i} A_{ij}x_j \]
- Write restricted values of $x_i$ (to shared memory - or other processors)
  - No iteration counts
  - Can tag $x_i$ with wall clock when writing it
  - Take-home message 2: Asynchronous iterations work very well!
An application. Numerical experiments

Chicxulub Crater, created by a collision of an asteroid approx. 66 million years ago: Cretaceous-Paleogen boundary: extinction of dinosaurs, approx. diameter 180km (pictures NASA, 2010)
Our experiments

We want to compute the gravitational potential $\Phi$ on a parallelepiped geometric domain of dimensions $250\,km \times 250\,km \times 15\,km$.

Finite element mesh
Equation to solve

\[ \Delta \Phi = -4\pi G \delta \rho \]

- \( G = 6.672 \times 10^{-11} \text{m}^3\text{kg}^{-1}\text{s}^{-2} \) gravitational constant
- \( \delta \rho \) anomaly density distribution computed from data acquisition on a salt dome (produced by the impact)

Close up of the salt dome geometry
[Magoules, S., Venet, 2017]
Three discretizations of box

- case I has 2,491,632 DOF (256 subdomains)
- case II has 19,933,056 DOF (512 subdomains)
- case III has 146,707,292 DOF (1024 subdomains)
- 1068 processors - 17,088 cores (half 1.6 Ghz with 2x2MB of cache, half 2.93 Ghz with 2x4MB of cache)
- (Synchronous) OSM and asynchronous OSM
- Compute optimal parameters using CMA-ES
- In each subdomain solve linear system directly

<table>
<thead>
<tr>
<th></th>
<th>iter</th>
<th>time</th>
<th>upt min</th>
<th>upt max</th>
<th>time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>case I (256)</td>
<td>1722</td>
<td>43</td>
<td>1030</td>
<td>1917</td>
<td>40</td>
</tr>
<tr>
<td>case II (512)</td>
<td>3379</td>
<td>777</td>
<td>2257</td>
<td>4438</td>
<td>591</td>
</tr>
<tr>
<td>case III (1024)</td>
<td>8331</td>
<td>3888</td>
<td>5251</td>
<td>13274</td>
<td>863</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>iter</th>
<th>time</th>
<th>upt min</th>
<th>upt max</th>
<th>time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>case I (256)</td>
<td>575</td>
<td>14</td>
<td>309</td>
<td>1334</td>
<td>13</td>
</tr>
<tr>
<td>case II (512)</td>
<td>938</td>
<td>214</td>
<td>627</td>
<td>2714</td>
<td>176</td>
</tr>
<tr>
<td>case III (1024)</td>
<td>1850</td>
<td>863</td>
<td>811</td>
<td>4820</td>
<td>352</td>
</tr>
</tbody>
</table>
Two-level RAS. 3D example. Weak scaling.

Each subdomain about 40K unknowns. 64, 256 and 4096 subdomains. Balanced load.

[Glusa, Boman, Chow, Rajamanickam, S., 2020]
Repeating

- Overlap is worth considering
- Asynchronous Optimized Schwarz and two-level RAS work well
- and they scale well
- No communication bottleneck, no synchronization!
Asynchronous parallel methods for fixed point problems

Long history mostly from the 1980’s and 1990’s

Very selected references:

Papers: [Chazan, Miranker, 1969], [Robert, 1976], [Baudet, 1978],
[El Tarazi, 1982], [Bertsekas, 1983], [El Baz, Miellou, Spiteri, 1996],
[Üresin, Dubois, 1989]

Books: [Bertsekas, Tsitsiklis, 1989], [Bahi, Contassot-Vivier, Couturier, 2008]

Surveys: [Frommer, S., 2000], [Spiteri, 2020]

All theory is based on product spaces (subdomains or group of variables, including the overlap case).
Essentially (linear and nonlinear) block Jacobi. Inherently slow. Asynchronous BJ faster but still slow.
Asynchronous parallel methods for fixed point problems

Long history mostly from the 1980’s and 1990’s

Very selected references:

Papers: [Chazan, Miranker, 1969], [Robert, 1976], [Baudet, 1978],
[El Tarazi, 1982], [Bertsekas, 1983], [El Baz, Miellou, Spiteri, 1996],
[Uresin, Dubois, 1989]

Books: [Bertsekas, Tsitsiklis, 1989], [Bahi, Contassot-Vivier, Couturier, 2008]

Surveys: [Frommer, S., 2000], [Spiteri, 2020]

All theory is based on product spaces (subdomains or group of variables, including the overlap case).
Essentially (linear and nonlinear) block Jacobi. Inherently slow. Asynchronous BJ faster but still slow.

What is different now?
Now, OSM fast, AOSM fast.
Mathematical Models: Asynchronous iterations for $x = \mathcal{T}x$

For each time stamp $k \in \mathbb{N}$, let $I^k \subseteq \{1, \ldots, q\}$ (the set of variables written at time stamp $k$) and $(s_1(k), \ldots, s_q(k)) \in \mathbb{N}_0^q$ where $s_j(k)$ is the tag of variable $j$ available when computation starts ending in a variable $i$ written at time stamp $k$, such that (typical three assumptions)

$$s_j(k) < k \text{ for } j \in \{1, \ldots, q\}, \ k \in \mathbb{N}$$

(only read variables already computed)

$$\lim_{k \to \infty} s_j(k) = \infty \text{ for } j \in \{1, \ldots, q\}$$

(no information is stale forever)

$$|\{k \in \mathbb{N} : i \in I^k\}| = \infty \text{ for } i \in \{1, \ldots, q\}$$

(each variable is eventually updated)
Mathematical Models: Asynchronous iterations for $x = T x$

Given an initial vector $x^0 \in E = E_1 \times \ldots \times E_q$, the iteration

$$x_i^k = \begin{cases} T_i(x_1^{s_1(k)}, \ldots, x_q^{s_q(k)}) & \text{for } i \in I^k \\ x_i^{k-1} & \text{for } i \notin I^k, \end{cases}$$

is termed an \textit{asynchronous iteration} (with strategy $I^k$, $k \in \mathbb{N}$ and delays $d_i(k) = k - s_i(k)$, $i = 1, \ldots q$, $k \in \mathbb{N}$).

For \textit{bounded delays}, there exist $d$ such that $d_i(k) \leq d$ for all $i, k$. 
Typical convergence theorem

For a fixed point iteration $x(k+1) = T x(k)$, if $\|T\| < 1$, for some operator norm conformal with the product space, and with the typical assumptions, asynchronous iteration converges to the unique fixed point.

e.g., [El Tarazi, 1982], [Bertsekas, 1983]

Notes: No convergence rate (and no iteration counts!)
In other theorems, condition is $\rho(|T|) < 1$.
We used these theorems to show convergence for AOSM in some settings and for two-level asynchronous RAS
At each time stamp $k$,

$$x_i^k = \begin{cases} 
  T_i(x_1^{s_1(k)}, \ldots, x_q^{s_q(k)}) & \text{with probability } p_i \\
  x_i^{k-1} & \text{with probability } 1 - p_i
\end{cases}$$

[Strikwerda, LAA, 2002] where he also had $s_i(k)$ as random variables

Of course $\sum_{i=1}^q p_i = 1$

Strikwerda proved that $\mathbb{E}(\|x_k - x^*\|) \rightarrow 0$

for $T = B$, $\rho(B) < 1$

and in fact $\mathbb{E}(\|x_k - x^*\|) = O(R^{-k})$ for some real number $R$

(radius of analyticity of a matrix $M(z) = I - z(I - P + s(z)PB)$, $P = \text{diag}(p_i)$, $s(z)$ related to randomized $s_i(k)$)

Note: This is analysis of “classical” asynchronous iterations, not a new randomized method

They do propose a new algorithm where probabilities are used. Essentially Asynchronous Randomized (point) Jacobi ($\equiv$ Randomized Gauss-Seidel). Let $A = D - B$, $D = \text{diag}(A)$, $H = D^{-1}A$, $c = D^{-1}b$

$$\text{for } m = 1, 2, \ldots \text{ do}$$

choose index $i$ with probability $p_i$

$$x_i^{m+1} = \sum_{j=1}^{n} h_{ij} x_j^m + c_i, \quad x_{\ell}^{m+1} = x_{\ell}^m \text{ for } \ell \neq i$$

end for

Note: $m$ here counts relaxations, not iterations.
Computational model here: 1. Bounded delays $k - s_i(k) \leq d$.
2. Atomic write: only one component is updated for every time stamp.

**Theorem.** If $\|H\|_\infty$ small enough so that $\|H\|_\infty < n/2d$ (or given $A$, the delay $d$ small enough), and if the probabilities are uniform, then

$$\mathbb{E}(\|x_m - x_*\|^2_A) \leq \beta \alpha^{m/(d_0 + d)} \|x_0 - x_*\|^2_A$$

where $\beta$, $\alpha$ functions of $\lambda_{\text{max}}(H)$, $d$, $n$, $\|H\|_\infty$, and $\kappa(H)$, and

$$d_0 = \left\lfloor \frac{\log(1/2)}{\log(1 - \lambda_{\text{max}}/n)} \right\rfloor$$
Challenge


We want to do the same for blocks and for $A$ nonsymmetric. What conditions on $A$? What norm to use?

Definition

Given a permutation and partition $\pi$ into $q$ sets of \{1, 2, \ldots, n\}. We define the $n \times n$ matrix $S_i$ with the columns of $I$ corresponding to the set $\pi_i$. Let $S = [S_1, \ldots, S_q]$, it is a complete sketching. Let $A_{ij} = S_i^T A S_j$. Assume that $A_{ii}$ is nonsingular, $i = 1, \ldots, n$. $A$ is called (strictly) block (column) diagonally dominant (BDD) in the sense of Robert [1969] if

$$\sum_{i=1}^{q} \|A_{ii}^{-1}A_{ij}\| < 1, \text{ for } j = 1, \ldots, q$$

That is, if $D = \text{diag}(A_{ii})$, $H = D^{-1}A$, $A$ BDD, then

$$\max_j \sum_{i=1}^{q} \|H_{ij}\| < 1$$
New Definitions

Let $u > 0$, $u \in \mathbb{R}^q$. $A$ is called generalized (strictly) block (column) diagonally dominant (GBDD) if

$$\frac{1}{u_j} \sum_{i=1}^{q} \|A^{-1}_{ii} A_{ij}\| u_i < 1, \text{ for } j = 1, \ldots, q$$

That is, if $D = \text{diag}(A_{ii})$, $H = D^{-1} A$, $A$ GBDD iff

$$\|H\|_{S,u} := \max_j \frac{1}{u_j} \sum_{i=1}^{q} \|H_{ij}\| u_i < 1$$

This matrix norm is induced from the (block weighted $\ell_1$) vector norm

$$\|v\|_{S,u} = \sum_{i=1}^{q} u_i \|S_i^T v\|$$
Theorem. [Frommer, S., 2022] Fix a permutation and partition \( \pi \) into \( q \) sets, and corresponding matrix \( S \). Let \( u > 0, u \in \mathbb{R}^q \). Let \( A \) be generalized BDD w.r.t. \( \pi, u \). Let \( A = D - B, D = \text{blockdiag}(A), H = D^{-1}A, c = D^{-1}b, \hat{r}_k = c - Hx_k \). Assume that 

\[
\|H\|_{S,u} = \max_{j=1}^{q} \rho_j < 1,
\]

where \( \rho_j \) denotes the weighted block column sum

\[
\rho_j = \frac{1}{u_j} \sum_{i=1}^{q} u_i \|H_{ij}\|, \quad j = 1, \ldots, q.
\]

Let \( \alpha = \min_{j} \rho_j (1 - \rho_j) \). Then,

\[
\mathbb{E}(\|\hat{r}^k\|_{S,u}) \leq (1 - \alpha)^{[k/d]} \|\hat{r}^0\|_{S,u}.
\]

\( k \) here are block relaxations - time stamps. \( d \) the bound on the delay. Asynchronous iterations.
Conclusions

- Asynchronous iterations work. They can work very fast, especially with overlap.
- After 40 years, we have now provable linear convergence rate for large classes of matrices.

Papers and reports can be found at: math.temple.edu/szyld