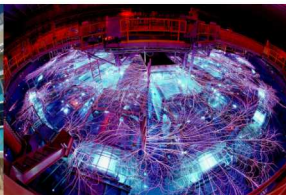


This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

Exceptional Service to the National Energy



SAND2019-1936C



Asynchronous Domain Decomposition Solvers

SIAM CSE19

Christian Glusa

Center for Computing Research, Sandia National Laboratories

February 25, 2019



Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract number DE-AC05-04OR21400.

Synchronous iterative methods:

- Alternate computation and communication
→ High network peak usage
- Are limited by global synchronization at scale, in particular under load imbalance due to
 - heterogeneous systems,
 - unbalanced local solve costs,
 - (transient) variability in communication cost

Asynchronous methods:

- Spread out communication in time,
- Have a chance of absorbing load imbalance more easily.
- Sacrifice deterministic behavior.
- Fault resilience

Can asynchronous linear solvers be used at scale?

Restricted Additive Schwarz iteration (RAS)

Solve

$$\begin{cases} -\Delta u &= f & \text{in } \Omega \subset \mathbb{R}^d, \\ u &= 0 & \text{on } \partial\Omega. \end{cases}$$

for $k = 1, 2, \dots$ **do**

Residual $r^{(k)} = f + \Delta u^{(k-1)}$

Solve in parallel

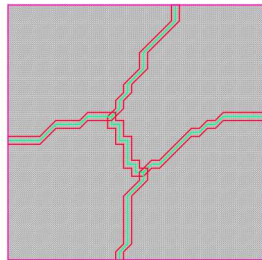
$$\begin{cases} -\Delta d_p^{(k)} &= r^{(k)} & \text{in } \Omega_p, \\ d_p^{(k)} &= 0 & \text{on } \partial\Omega_p. \end{cases}$$

Update

$$u^{(k)} = u^{(k-1)} + \sum_p \theta_p d_p^{(k)}$$

- $\{\theta_p\}$ is a partition of unity.
- Careful choice of θ_p eliminates need for communication in the update step.

Overlapping partition $\Omega = \cup_p \Omega_p$



Algebraic version of RAS

Solve

$$\mathbf{A}\vec{u} = \vec{f}.$$

For $k = 1, 2, \dots$

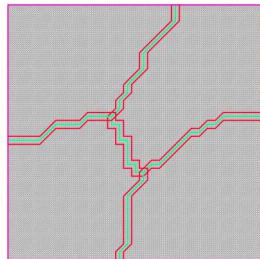
$$\vec{u}^{(k)} = \vec{u}^{(k-1)} + \sum_p \mathbf{R}_p^T \mathbf{D}_p \mathbf{A}_p^{-1} \mathbf{R}_p \left(\vec{f} - \mathbf{A} \vec{u}^{(k-1)} \right)$$

- $\mathbf{R}_p$ is the restriction to subdomain p ,
 $\mathbf{A}_p = \mathbf{R}_p \mathbf{A} \mathbf{R}_p^T$.
- The diagonal matrices $\mathbf{D}_p$ satisfy discrete partition of unity property

$$\mathbf{I} = \sum_p \mathbf{R}_p^T \mathbf{D}_p \mathbf{R}_p.$$

- Similar to Additive Schwarz, but does not require damping to work as an iterative method.
- Unsymmetric iteration.

Overlapping partition $\Omega = \cup_p \Omega_p$



$$\underbrace{\vec{u}^{(k)} = \vec{u}^{(k-1)} + \sum_p \mathbf{R}_p^T \mathbf{D}_p}_{\text{update}} \underbrace{\mathbf{A}_p^{-1}}_{\text{local solve}} \underbrace{\mathbf{R}_p \left(\vec{f} - \mathbf{A} \vec{u}^{(k-1)} \right)}_{\text{residual}}$$

- Computation of the residual requires neighborhood communication (halo values).
- Local solve and update are communication free.

Synchronous residual computation at iteration k :

$$\mathbf{R}_p \left(\vec{f} - \mathbf{A} \vec{u}^{(k-1)} \right) = \vec{r}_{p,\text{local}}^{(k-1)} + \sum_q \vec{r}_{q,\text{halo}}^{(k-1)}$$

Asynchronous residual computation at time t :

$$\mathbf{R}_p \left(\vec{f} - \mathbf{A} \vec{u}^{(t)} \right) = \vec{r}_{p,\text{local}}^{(t)} + \sum_q \vec{r}_{q,p,\text{halo}}^{(t)}$$

$\vec{r}_{q,p,\text{halo}}^{(t)}$ can be out of date since it is computed on subdomain q and needs to be sent across the network.

- Asynchronous iteration is not mathematically equivalent to the synchronous one.
- Special convergence theory is required. (See work by Szyld, Frommer, etc)

Realizing asynchronous communication: One-sided MPI

- Allows MPI ranks to directly read/write exposed memory regions on other ranks.
- If the hardware (network interfaces) and MPI implementation support it, can be significantly faster than two-sided communication.

<code>MPI_Win_create</code>	Expose a memory region.
<code>MPI_Put</code>	Write into a window.
<code>MPI_Get</code>	Read from a window.
<code>MPI_Win_lock /</code> <code>MPI_Win_unlock</code>	Locks/unlocks a window, exclusive or shared access.
<code>MPI_Win_lock_all /</code> <code>MPI_Win_unlock_all</code>	Locks/unlocks a window on all processes.
<code>MPI_Win_flush</code>	Complete all outstanding operations.
<code>:</code>	

After testing the different options, we settled on `MPI_Win_lock_all / MPI_Win_unlock` and `MPI_Put`.

Convergence detection in synchronous iterations: $\|\vec{r}\| < \varepsilon$.

For asynchronous methods:

- No global iteration count, no synchronized global residual
- Norms and inner products introduce synchronization points.

Our ad-hoc approach for convergence detection:

- Compute norm of local residual vector $\|\vec{r}_p\|$
- Save to window on rank 0
- Rank 0 sums contributions and exposes result in a different window.
- Subdomains retrieve approximate residual norm and terminate if small enough.

Drawbacks:

- Local residual norms are not necessarily monotonically decreasing.
- Might iterate longer than necessary when rank 0 is busy.

- No GMRES acceleration (inner products and norms are synchronization points)
 - Need to use the best possible fixed point iteration.
- One-level methods are not scalable
 - Asynchronous coarse grid solve for global information exchange

Subdomain problem in RAS

$$\begin{cases} -\Delta d_p^{(k)} = r^{(k)} & \text{in } \Omega_p, \\ d_p^{(k)} = 0 & \text{on } \partial\Omega_p. \end{cases}$$

The optimal choice would be

$$\begin{cases} -\Delta d_p^{(k)} = r^{(k)} & \text{in } \Omega_p, \\ d_p^{(k)} = 0 & \text{on } \partial\Omega_p \cap \partial\Omega, \\ \frac{\partial d_p^{(k)}}{\partial \vec{n}} - \Lambda_{\Omega_p^c} \left(d_p^{(k)} \right) = 0 & \text{on } \partial\Omega_p \setminus \partial\Omega, \end{cases}$$

where $\Lambda_{\Omega_p^c}$ is the Poincaré-Steklov or Dirichlet-to-Neumann operator with respect to $\Omega_p^c = \Omega \setminus \Omega_p$. One can show convergence in at most $P - 1$ iterations!

oRASO

But: evaluation of $\Lambda_{\Omega_p^\epsilon}$ is as expensive as the original problem.
Lowest order approximation

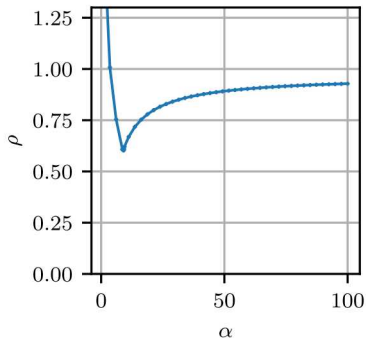
$$\Lambda_{\Omega_p^\epsilon}(d) \approx \alpha d, \quad \alpha > 0,$$

i.e. Robin condition, leads to *optimized Restricted Additive Schwarz iteration* of 0^{th} order (oRASO).

- Replace subdomain matrices $\mathbf{A}_p \rightarrow \mathbf{A}_p + \alpha \mathbf{M}_p^{\partial\Omega_p}$, where $\mathbf{M}_p^{\partial\Omega_p}$ is the surface mass matrix.
- $\alpha \rightarrow \infty$ recovers usual RAS, $\alpha \rightarrow 0$ blows up.

How could we determine the optimal value of α efficiently?

- Fourier-type analysis (e.g. Gander [2], Garay & Szyld),
- Extrapolation from a coarser version of the problem,
- Optimization algorithms



Numerical Experiments with asynchronous oRASO

- 2D Finite Element Poisson on unstructured mesh
- 64 subdomains
- # DoFs: $\approx 261,000$
- Subdomain problem solved using direct factorization
- 5 samples per value of α
- Synchronous communication: non-blocking two-sided MPI
- Asynchronous communication: one-sided MPI
- Is the asynchronous method converging?
- How does the asynchronous iteration compare to the synchronous one?
Observed contraction factor:

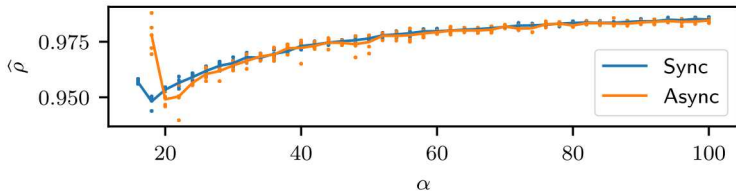
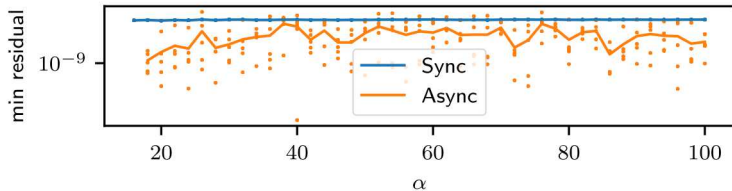
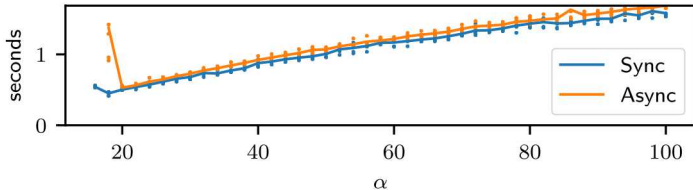
$$\hat{\rho} = \left(\frac{\text{Final residual}}{\text{Initial residual}} \right)^{\frac{\text{Average time per synchronous iteration}}{\text{Solve time}}}$$

(Reduces to contraction per iteration in synchronous case.)

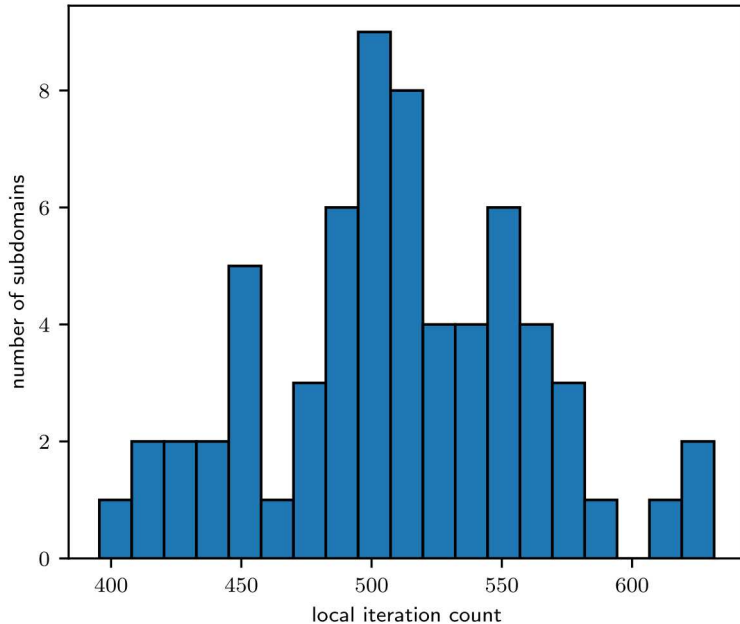
- Is the dependence on α different from the synchronous case?

(See also results in Magoulès, Szyld, and Venet [3])

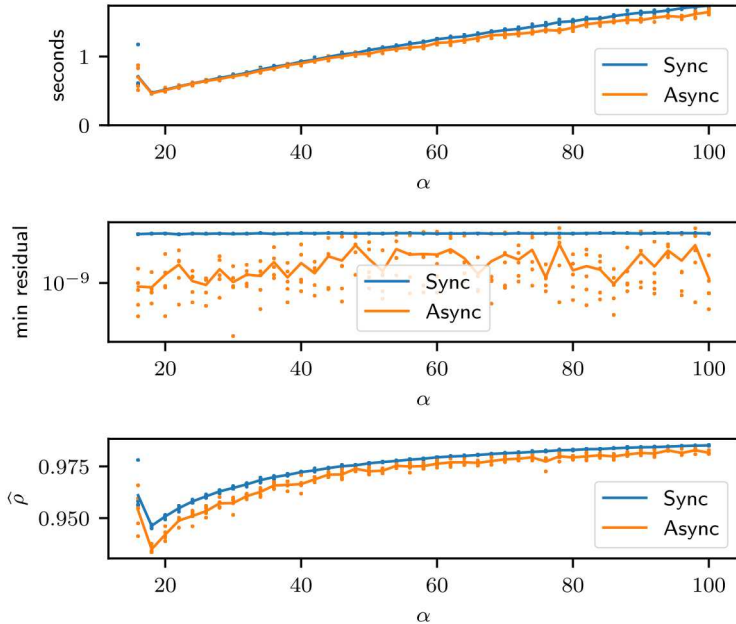
Numerical Experiments with oRASO, sync vs async



Local iteration counts for asynchronous oRASO



Numerical Experiments with oRAS0, one subdomain 50% bigger



Synchronous Coarse Grid correction

- To achieve scalability, a mechanism of global information exchange is required.
- $\mathbf{A}_0$ coarser version of $\mathbf{A}$, and $\mathbf{R}_0$ restriction to coarse DoFs.
- Synchronous additive coarse grid correction to standard RAS:

$$\vec{u}^{(k)} = \vec{u}^{(k-1)} + \left(\frac{1}{2} \sum_p \mathbf{R}_p^T \mathbf{D}_p \mathbf{A}_p^{-1} \mathbf{R}_p + \frac{1}{2} \mathbf{R}_0^T \mathbf{A}_0^{-1} \mathbf{R}_0 \right) (\vec{b} - \mathbf{A} \vec{u}^{(k-1)})$$

(Similar for oRAS0, but lack of a strategy for selection of α , use RAS for now.)

- Communication pattern

$$\underbrace{\frac{1}{2} \mathbf{R}_0^T}_{\text{prolongation}} \underbrace{\mathbf{A}_0^{-1}}_{\text{coarse grid solve}} \underbrace{\mathbf{R}_0 (\vec{b} - \mathbf{A} \vec{u}^{(k-1)})}_{\text{residual}}$$

- Residual *and* prolongation require communication between *all* subdomains and coarse grid.
- Coarse grid solve can be distributed itself and internally require communication, but we keep it synchronous.

Asynchronous coarse grid cannot be handled in the same way as the subdomains.

Asynchronous Coarse Grid correction

On coarse grid

```

while not converged do
  if RHSisReady0[p]  $\forall$  subdomains p then
    Solve  $A_0 x = rhs_0$ 
    for subdomain p do
      RHSisReady0[p]  $\leftarrow$  False
      canWriteRHSp  $\leftarrow$  True
      solutionp  $\leftarrow$  x
      solutionIsReadyp  $\leftarrow$  True

```

On subdomain p

```

while not converged do
  Compute residual asynchronously
  if canWriteRHSp then
    Compute local part of coarse grid residual rhsp
    rhs0  $\leftarrow$  rhs0 + rhsp
    canWriteRHSp  $\leftarrow$  False
    RHSisReady0[p]  $\leftarrow$  True
  Solve local problem
  if solutionIsReadyp then
    Update local iterate with solutionp and
    solution of local problem (weights (1/2, 1/2))
    solutionIsReadyp  $\leftarrow$  False
  else
    Update local iterate with solution of local
    problem (weight 1)

```

Types of variables:

window on remote rank

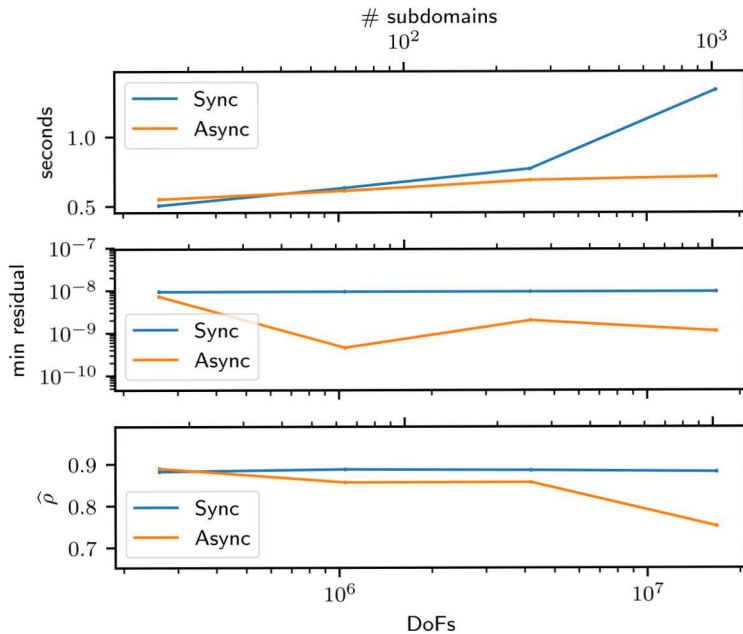
window on local rank

local variable

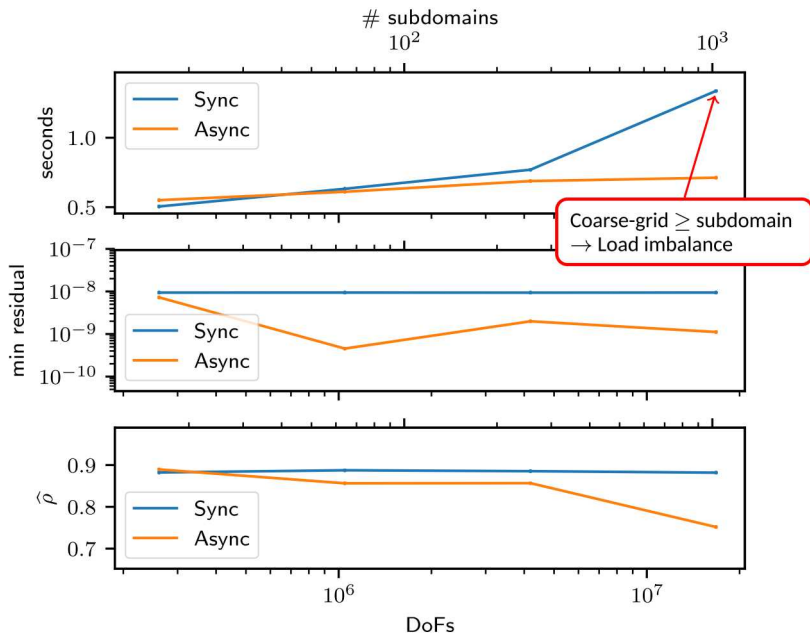
- 2D Finite Element Poisson
- Weak scaling experiment:

subdomains:	16	64	256	1024
unknowns:	256k	1M	4M	16M
- # DoFs per subdomain: $\approx 20,000$
- # DoFs for coarse grid: $\approx 16 \times \text{\#subdomains}$
- Subdomain and coarse grid solves using direct factorization.
- Coarse grid solve on a single MPI rank.
- Haswell partition on Cori @ NERSC, 1 rank/core

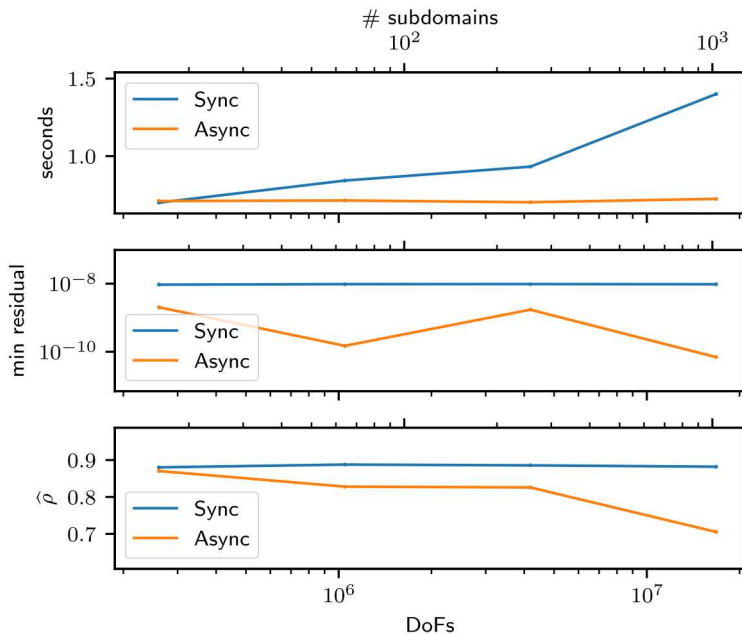
2D, RAS w/ additive coarse grid correction, sync vs async



2D, RAS w/ additive coarse grid correction, sync vs async



2D, RAS w/ additive coarse grid correction, one subdomain 50% bigger



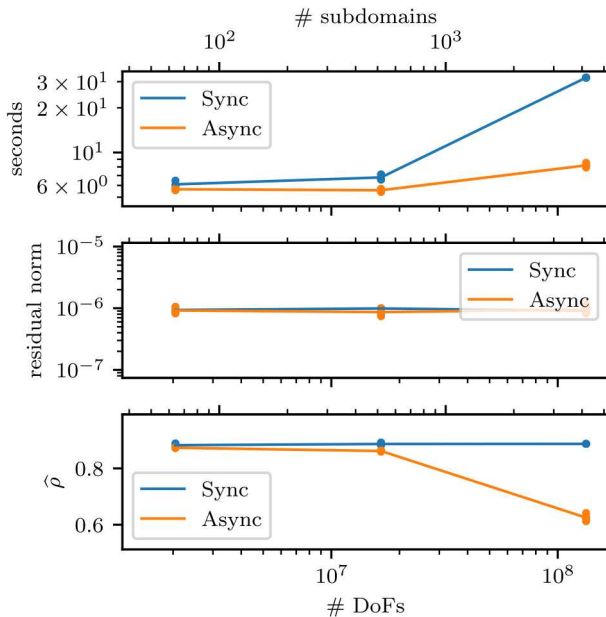
3D Numerical Experiments, RAS w/ additive coarse grid

- 3D Poisson equation on a box, structured partitioning.
- Weak scaling experiment:

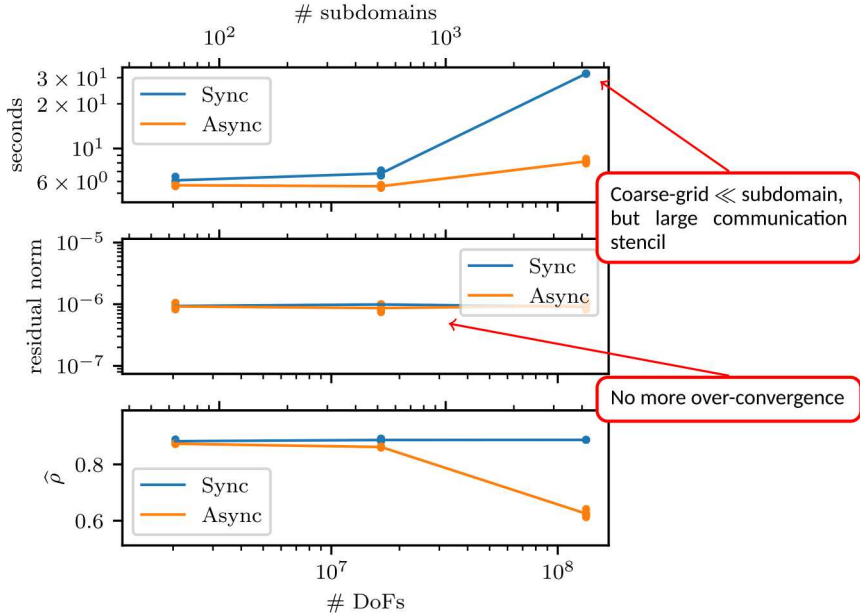
subdomains:	64	512	4096	32768
unknowns:	2M	16M	128M	1B
- # DoFs per subdomain: $\approx 40k$
- # DoFs for coarse grid: $\approx$ one per subdomain
- Subdomain solves using conjugate gradients with incomplete Cholesky preconditioning, relative tolerance $1/10$
- Coarse solve using a single multigrid V-cycle.
- Coarse solve is distributed if the coarse problem is large.
- Haswell partition on Cori @ NERSC, 1 rank/core

Built-in possibility of load imbalance: local iterative solves

3D, RAS w/ additive coarse grid correction

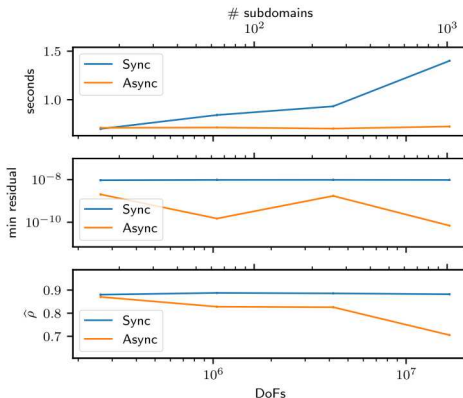
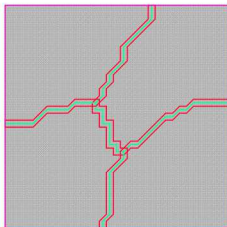


3D, RAS w/ additive coarse grid correction



Conclusion and Outlook

- Asynchronous solvers can compete with their synchronous equivalents.
- Load imbalance is absorbed more easily.
- Need to pick a strategy for selection of optimization parameter α if we want to use oRAS0 (and higher order variants).
- Try asynchronous solvers on more interesting problems in real applications.



References I



Philippe Chevalier and Frédéric Nataf. “Symmetrized method with optimized second-order conditions for the Helmholtz equation”. In: *Contemporary Mathematics* 218 (1998), pp. 400–407.



Martin J Gander. “Optimized Schwarz Methods”. In: *SIAM Journal on Numerical Analysis* 44.2 (2006), pp. 699–731.



Frédéric Magoulès, Daniel B. Szyld, and Cédric Venet. *Asynchronous Optimized Schwarz Methods with and without Overlap*. Tech. rep. 15-06-29. Department of Mathematics, Temple University, June 2015.



Frédéric Nataf. “Absorbing boundary conditions in block Gauss–Seidel methods for convection problems”. In: *Mathematical Models and Methods in Applied Sciences* 6.04 (1996), pp. 481–502.



Frédéric Nataf and Francis Nier. “Convergence rate of some domain decomposition methods for overlapping and nonoverlapping subdomains”. In: *Numerische Mathematik* 75.3 (1997), pp. 357–377.