Scalable preconditioners for Computational Science at extreme scale

Pasqua D'Ambra, Institute for Applied Computing (IAC-CNR), IT
and
Fabio Durastante, University of Pisa and IAC-CNR,
Salvatore Filippone, University of Rome Tor-Vergata, and IAC-CNR

13th International Conference on Large-Scale Scientific Computations

Sozopol, 7-11 June, 2021

E·C·E

Greetings and project collaborators:

- Panayot S. Vassilevski, CASC-LLNL (Livermore, CA) and Portland State University (Portland, OR), USA
- Mahantesh M. Halappanavar, PNNL (Richland, WA), USA

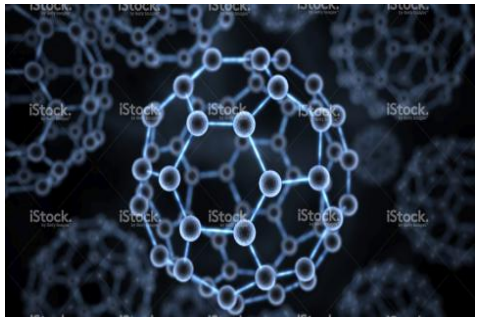
- Massimo Bernaschi (IAC-CNR), IT
- Daniele Bertaccini (Univ. of Rome Tor-Vergata and IAC-CNR), IT
- Herbert Owen (BSC), ES
- Dario Pasquini (IAC-CNR), IT

Motivation: EoCoE-II project

Energy oriented Center of Excellence: toward exascale for energy

applying cutting-edge computational methods to accelerate the transition to the production, storage and management of clean, decarbonized energy

Wind



Materials

Water

Fusion

Main aim

prepare selected numerical models to be run on (future) exascale supercomputers

Key Computational Kernel

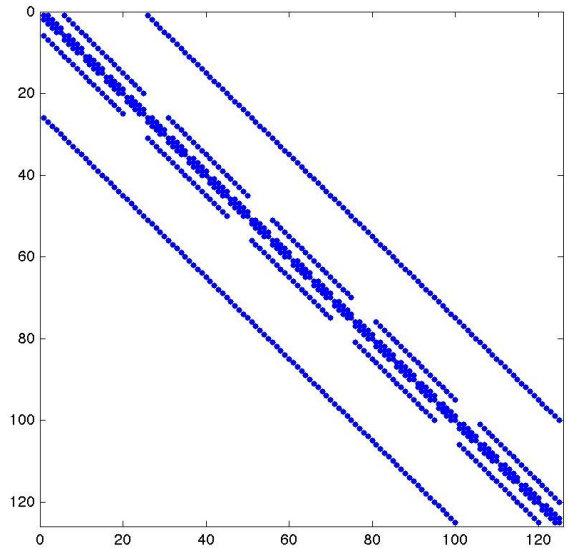
Poisson-type PDE

$$-\nabla \cdot \mathbf{K}\nabla u = f \quad \text{in } \Omega$$
$$\alpha u + \beta \frac{\partial u}{\partial \vec{n}} \quad \text{in } \partial\Omega$$

$A\mathbf{x} = \mathbf{b}, \quad A \in \mathcal{R}^{n\times n}$ (s.p.d.) $\mathbf{x}, \mathbf{b} \in \mathcal{R}^n$

$n >> 10^9$

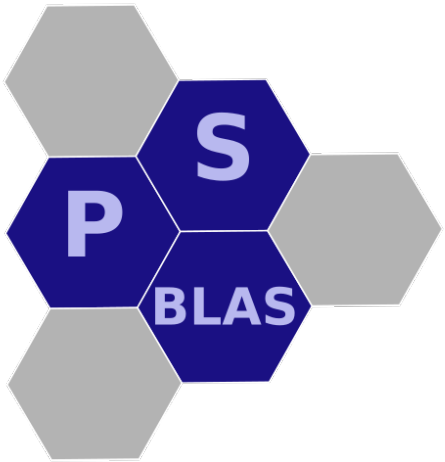
sparsity degree $= 1 - \frac{nnz}{n^2} \approx 1$

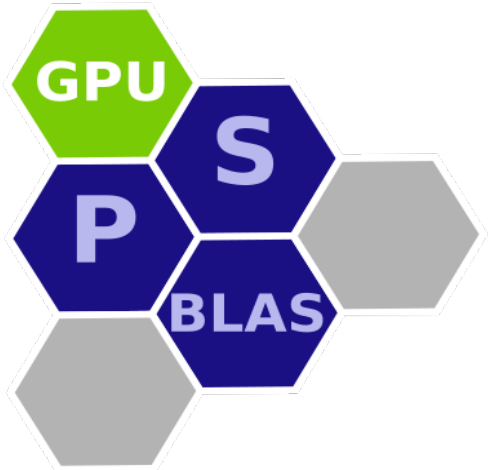


Wind applications from Barcelona Supercomputing Center (BSC)

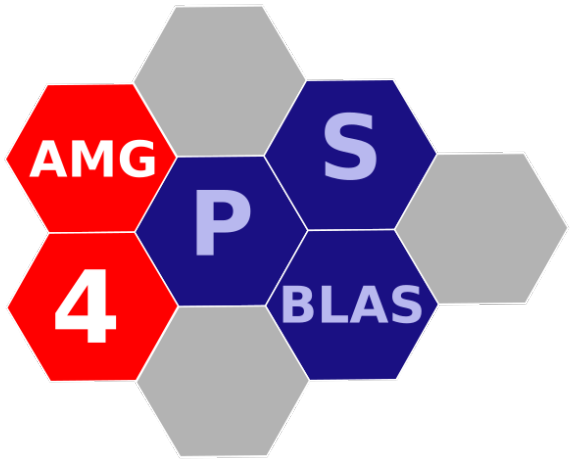
high-resolution LES models in wind farm applications require (block structured) meshes with more than ten billions ($> 10^{10}$) dofs

Parallel Sparse Computation Toolkit





Parallel Sparse Basic Linear Algebra Subroutines
+
its GPU-plugin



AMG Preconditioners for PSBLAS

Available from https://psctoolkit.github.io/
Recently selected as Excellent Innovation from EU Innovation Radar

PSBLAS

A Software development project started in the early 2000s, prompted by the work of Iain Duff et al. on standard for Sparse BLAS, ACM TOMS 23 (1997)

- parallel sparse matrix operations and data structures management, Krylov solvers (for spd and general matrices)
- general row-block matrix distribution, support infrastructure for mesh handling and sparse matrix I/O
- data allocation through graph partitioning (METIS, ParMETIS, SCOTCH)
- object oriented design in Fortran 2003/2008

S. Filippone et al., PSBLAS: A library for parallel linear algebra computation on sparse matrices, ACM TOMS, 26, 4, 2000.
S. Filippone et al., Object-Oriented Techniques for Sparse Matrix Computations in Fortran 2003. ACM TOMS, 38, 4, 2012.

PSBLAS (contd.)

- message-passing paradigm (MPI), CUDA plugin
- internal matrix representation/storage: distributed sparse matrix with native CSR/CSC/COO format, extension plugin for many other storage formats, including CUDA-enabled (DIAG, ELLPACK and variations)
- multiple tools for storage transformations
- tools and data structures for global/local index mapping (long/short integers for global/local numbering handled separately) and halo data exchange;

Memory footprint (essential for scalability)

- matrices: proportional to number of local rows/indices (may require padding for vectorization on GPUs and similar)
- vectors: proportional to local indices plus halo indices
- data exchange auxiliary storage: proportional to number of boundary plus halo indices
- global/local index mappings: proportional to local plus halo indices (may trade more memory for speed if acceptable)

AMG4PSBLAS: AMG Preconditioners for PSBLAS

A software development project started in 2004

- initially developed as a package of algebraic multigrid Schwarz preconditioners, extended to more general AMG preconditioning within EoCoE
- object-oriented design in Fortran 2003/2008, layered sw architecture on top of PSBLAS
  $\Longrightarrow$ modularity and flexibility
- clear separation between interface and implementation of methods
  $\Longrightarrow$ performance and extensibility (e.g. works transparently on GPUs)
- separated users' interface for setup of the multigrid hierarchy and setup of the smoothers and solvers to have large flexibility at each level

P. D'Ambra et al., MLD2P4: a Package of Parallel Algebraic Multilevel Domain Decomposition Preconditioners in Fortran 95, ACM TOMS, 37, 3, 2010
P. D'Ambra et al., AMG preconditioners for Linear Solvers towards Extreme Scale. Preprint arXiv:2006.16147 (to appear on SISC)

AMG Preconditioners

Example: symmetric V-cycle

procedure V-cycle$\left(k, nlev, A^k, b^k, x^k\right)$

if $(k \neq nlev)$ then

$x^k = x^k + (M^k)^{-1}\left(b^k - A^k x^k\right)$

$b^{k+1} = \left(P^{k+1}\right)^T \left(b^k - A^k x^k\right)$

$x^{k+1} = \text{V-cycle}\left(k+1, A^{k+1}, b^{k+1}, 0\right)$

$x^k = x^k + P^{k+1} x^{k+1}$

$x^k = x^k + (M^k)^{-T}\left(b^k - A^k x^k\right)$

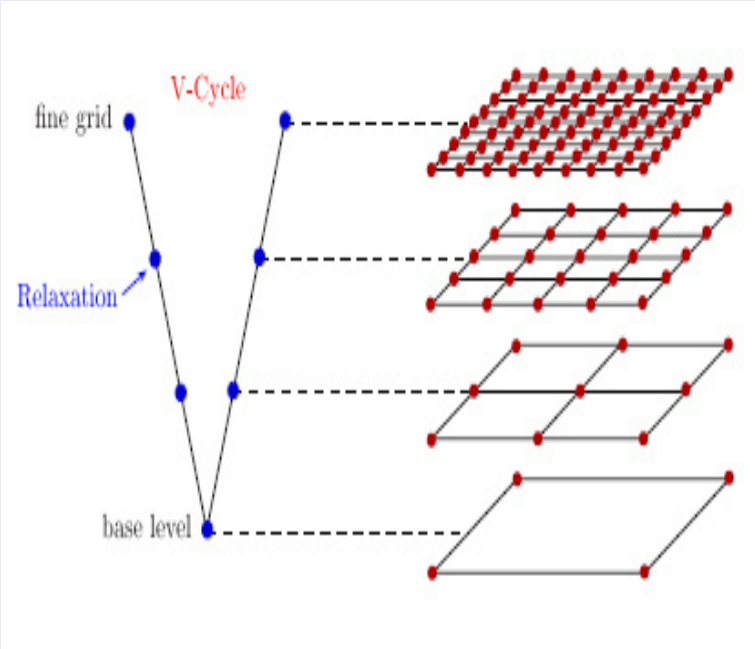
else

$x^k = \left(A^k\right)^{-1} b^k$

endif

return x^k

end



AMG methods do not explicitly use the problem geometry and rely only on matrix entries to generate coarse grids (setup phase)

Scalable (optimal) preconditioners

Solve the system:

$$B^{-1}Ax = B^{-1}b,$$

with matrix $B \approx A^{-1}$ (left preconditioner) such that:

- $\mu(B^{-1}A) \approx 1$, being independent of n (algorithmic scalability)
- the action of B^{-1} costs as little as possible, the best being $\mathcal{O}(n)$ flops (linear complexity)
- in a massively parallel computer, B^{-1} should be composed of local actions, **(implementation scalability, i.e., performance linearly proportional to the number of processors employed)**

Scalable (optimal) preconditioners

Solve the system:

$$B^{-1}Ax = B^{-1}b,$$

with matrix $B \approx A^{-1}$ (left preconditioner) such that:

- $\mu(B^{-1}A) \approx 1$, being independent of n (algorithmic scalability)
- the action of B^{-1} costs as little as possible, the best being $\mathcal{O}(n)$ flops (linear complexity)
- in a massively parallel computer, B^{-1} should be composed of local actions, **(implementation scalability, i.e., performance linearly proportional to the number of processors employed)**

MultiGrid performance parameters

- convergence rate $\rho < 1$: affects number of solver iterations
- operator complexity $opc = \frac{\sum_{k=0}^{nlev-1} nnz(A^k)}{nnz(A^0)}$: affects memory requirements and cycle time
- average stencil size $s(A^k) = nnz\_row(A^k)$: affects computation and communication both in setup and in cycle time

AMG4PSBLAS extensions for EoCoE

AMG4PSBLAS preconditioners can be obtained as combination of

setup or coarsening phase: parallel decoupled/coupled

- DVB: smoothed aggregation based on the usual strength of connection measure (Vaněk and Brezina, 1996)
- CMATCH: smoothed and unsmoothed aggregation based on compatible weighted matching (D'Ambra et al., 2013, 2016, 2018, 2021)

solve phase: **available on GPU** for many choices of smoothers & coarsest solver

- cycles: V, W, K
- smoothers: Jacobi and $l1$-Jacobi, hybrid (F/B) Gauss-Seidel and $l1$-GS, block-Jacobi / additive Schwarz with LU, ILU factorizations or sparse approximate inverses of the blocks
- coarsest-solvers: sparse LU, Jacobi and $l1$-Jacobi, hybrid (F/B) Gauss-Seidel and $l1$-GS, block-Jacobi with LU, ILU factorizations or sparse approximate inverses of the blocks, iterative preconditioned Krylov solvers

Parallel coarsening based on compatible weighted matching (CMATCH)

Let $\mathbf{w} \in \mathcal{R}^n$ smooth vector, let $P_c \in \mathcal{R}^{n \times n_c}$ and $P_f \in \mathcal{R}^{n \times n_f}$ be a prolongator and a complementary prolongator, such that:

$$\mathcal{R}^n = \mathcal{R}ange(P_c) \oplus^{\perp} \mathcal{R}ange(P_f), \quad n = n_c + n_f$$

$\mathbf{w} \in \mathcal{R}ange(P_c)$: coarse space $\qquad$ $\mathcal{R}ange(P_f)$: complementary space

$$[P_c, P_f]^T A [P_c, P_f] = \begin{pmatrix} P_c^T A P_c & P_c^T A P_f \\ P_f^T A P_c & P_f^T A P_f \end{pmatrix} = \begin{pmatrix} A_c & A_{cf} \\ A_{fc} & A_f \end{pmatrix}$$

A_c : coarse matrix $\qquad$ A_f : hierarchical complement

Parallel coarsening based on compatible weighted matching (CMATCH)

Let $\mathbf{w} \in \mathcal{R}^n$ smooth vector, let $P_c \in \mathcal{R}^{n \times n_c}$ and $P_f \in \mathcal{R}^{n \times n_f}$ be a prolongator and a complementary prolongator, such that:

$$\mathcal{R}^n = \mathcal{R}ange(P_c) \oplus^{\perp} \mathcal{R}ange(P_f), \quad n = n_c + n_f$$

$\mathbf{w} \in \mathcal{R}ange(P_c)$: coarse space $\qquad$ $\mathcal{R}ange(P_f)$: complementary space

$$[P_c, P_f]^T A [P_c, P_f] = \begin{pmatrix} P_c^T A P_c & P_c^T A P_f \\ P_f^T A P_c & P_f^T A P_f \end{pmatrix} = \begin{pmatrix} A_c & A_{cf} \\ A_{fc} & A_f \end{pmatrix}$$

A_c : coarse matrix $\qquad$ A_f : hierarchical complement

Sufficient condition for efficient coarsening

$A_f = P_f^T A P_f$ as well conditioned as possible,

i.e., convergence rate of compatible relaxation $\rho_f = \|I - M_f^{-1} A_f\|_{A_f} << 1$

Parallel coarsening based on compatible weighted matching (CMATCH)

Let $\mathbf{w} \in \mathcal{R}^n$ smooth vector, let $P_c \in \mathcal{R}^{n \times n_c}$ and $P_f \in \mathcal{R}^{n \times n_f}$ be a prolongator and a complementary prolongator, such that:

$$\mathcal{R}^n = \mathcal{R}ange(P_c) \oplus^{\perp} \mathcal{R}ange(P_f), \quad n = n_c + n_f$$

$\mathbf{w} \in \mathcal{R}ange(P_c)$: coarse space $\qquad$ $\mathcal{R}ange(P_f)$: complementary space

$$[P_c, P_f]^T A [P_c, P_f] = \begin{pmatrix} P_c^T A P_c & P_c^T A P_f \\ P_f^T A P_c & P_f^T A P_f \end{pmatrix} = \begin{pmatrix} A_c & A_{cf} \\ A_{fc} & A_f \end{pmatrix}$$

A_c : coarse matrix $\qquad$ A_f : hierarchical complement

Sufficient condition for efficient coarsening

$A_f = P_f^T A P_f$ as well conditioned as possible,

i.e., convergence rate of compatible relaxation $\rho_f = \|I - M_f^{-1} A_f\|_{A_f} << 1$

Our idea (D'Ambra et al., 2013, 2016, 2018)

build P_c (and P_f) by dofs **aggregation based on matching in the weighted (adjacency) graph of A**, to make A_f **as diagonally-dominant as possible**

CMATCH (cont'd)

Input: A matrix, $\mathbf{w}$ (smooth) vector,
maxsize maximum coarsest size
Output: hierarchy of coarse matrices A^k

1. $A^1 = A$, $k = 1$, $\mathbf{w^1} = \mathbf{w}$
2. **while** size$(A^k) >$ *maxsize*
   - 2.1 **apply** parallel matching-based pairwise aggregation to the graph of A^k with weigths depending on $\mathbf{w}^k$
   - 2.2 **build** P_c^k, $R_c^k = (P_c^k)^T$ and $A_c^k = R_c^k A^k P_c^k$
   - 2.3 $A^{k+1} = A_c^k$, $\mathbf{w}_c = R_c^k \mathbf{w}^k$
   - 2.4 $k = k + 1$

   endwhile

CMATCH (cont'd)

Input: A matrix, $\mathbf{w}$ (smooth) vector,
maxsize maximum coarsest size
Output: hierarchy of coarse matrices A^k

1. $A^1 = A$, $k = 1$, $\mathbf{w^1} = \mathbf{w}$
2. **while** size$(A^k) >$ *maxsize*
   - 2.1 **apply** parallel matching-based pairwise aggregation to the graph of A^k with weigths depending on $\mathbf{w}^k$
   - 2.2 **build** P_c^k, $R_c^k = (P_c^k)^T$ and $A_c^k = R_c^k A^k P_c^k$
   - 2.3 $A^{k+1} = A_c^k$, $\mathbf{w}_c = R_c^k \mathbf{w}^k$
   - 2.4 $k = k + 1$

   endwhile

Increasing Coarsening Ratio for Reducing Complexity

Consecutive levels based on pairwise aggregation can be combined,
e.g., double pairwise can be obtained by:

$$\overline{P}^k = P^{2k-1}P^{2k}, \quad \overline{R}^k = (\overline{P}^k)^T, \quad \overline{A}^k = A^{2k}, \quad k = 1, \ldots \lceil nl/2 \rceil$$

CMATCH (cont'd)

Approximation algorithms & parallel software

efficient (sub-optimal) algorithms (Catalyürek et al. 2012, Manne et al. 2014)

- quality guarantee of the computed matching, generally $1/2-$approximation to a maximum weight matching
- linear-time $\mathcal{O}(nnz)$ complexity
- available software in source form (MatchBox-P by Halappanavar et al.)

Main advantages of CMATCH

- a completely automatic procedure applicable to general s.p.d. systems, independent of any heuristics or a priori information on the near kernel of A
- well-balanced coarse matrices among parallel processes, no need for special treatment of process-boundary dofs accounting for inter-processes coupling
- significant flexibility in the choice of the size of aggregates, almost arbitrarily aggressive coarsening
- possible improving in V-cycle convergence, by smoothing of matching-based prolongators as in classic smoothed aggregation

Test Case

Poisson equation

$$-\Delta u = 1 \quad \text{on unit cube, with DBC}$$

- 7-point finite-difference discretization
- cartesian grid with uniform refinement along the coordinates for increasing mesh size

Solver/preconditioner settings

- AMG as preconditioner of Flexible CG, stopped when $\|\mathbf{r}^k\|_2/\|\mathbf{b}\|_2 \leq 10^{-6}$, or $itmax = 500$

  KCMATCH K-cycle with 2 inner iterations, CMATCH building aggregates of max size 8, unsmoothed prolongators

  VSCMATCH V-cycle, CMATCH building aggregates of max size 8, smoothed prolongators

  VSDVB V-cycle for decoupled classic smoothed aggregation

- 1 sweep of forward/backward Hybrid Gauss-Seidel smoother, parallel CG preconditioned with Block-Jacobi and ILU(0) at the coarsest level
- coarsest matrix size $n_c \leq 200np$, with np number of cores

Experimental environment & Comparison

Piz Daint - Swiss National Supercomputing Center by PRACE

- Cray Model XC40/Cray XC50 architecture with 5704 hybrid compute nodes (Intel Xeon E5-2690 v3 with Nvidia Tesla P100 accelerator)
- Cray Aries routing and communications ASIC with Dragonfly network topology
- GNU compiler rel. 8, Cray MPI 7, Cray-libsci 20.09.1
- PSBLAS 3.7, AMG4PSBLAS 1.0

Hypre: Scalable Linear Solvers and Multigrid Methods by LLNL

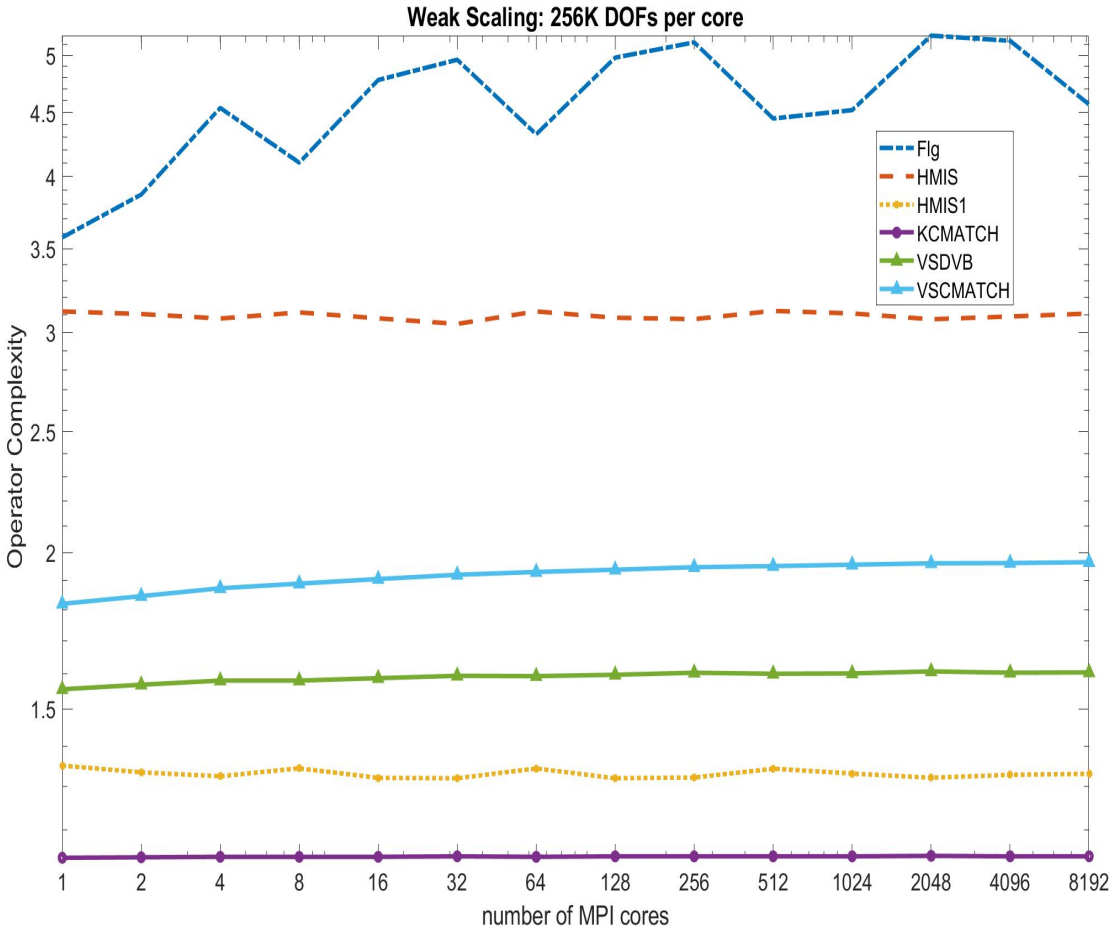
- BoomerAMG as preconditioner of CG, stopped when $\|\mathbf{r}^k\|_2/\|\mathbf{b}\|_2 \leq 10^{-6}$, or $itmax = 500$
- V-cycle with 1 sweep of forward/backward Hybrid Gauss-Seidel smoother, LU factorization at the coarsest level
- 3 coarsening schemes: hybrid RS/CLJP (**Flg**), Hybrid Maximal Independent Set (**HMIS**), HMIS with first level of aggressive coarsening (**HMIS1**); default parameters for coarsest matrix size $1 \leq n_c \leq 9$, coupled with modified (long-range) classical interpolation

Weak scaling (256K dofs per core): algorithmic scalability

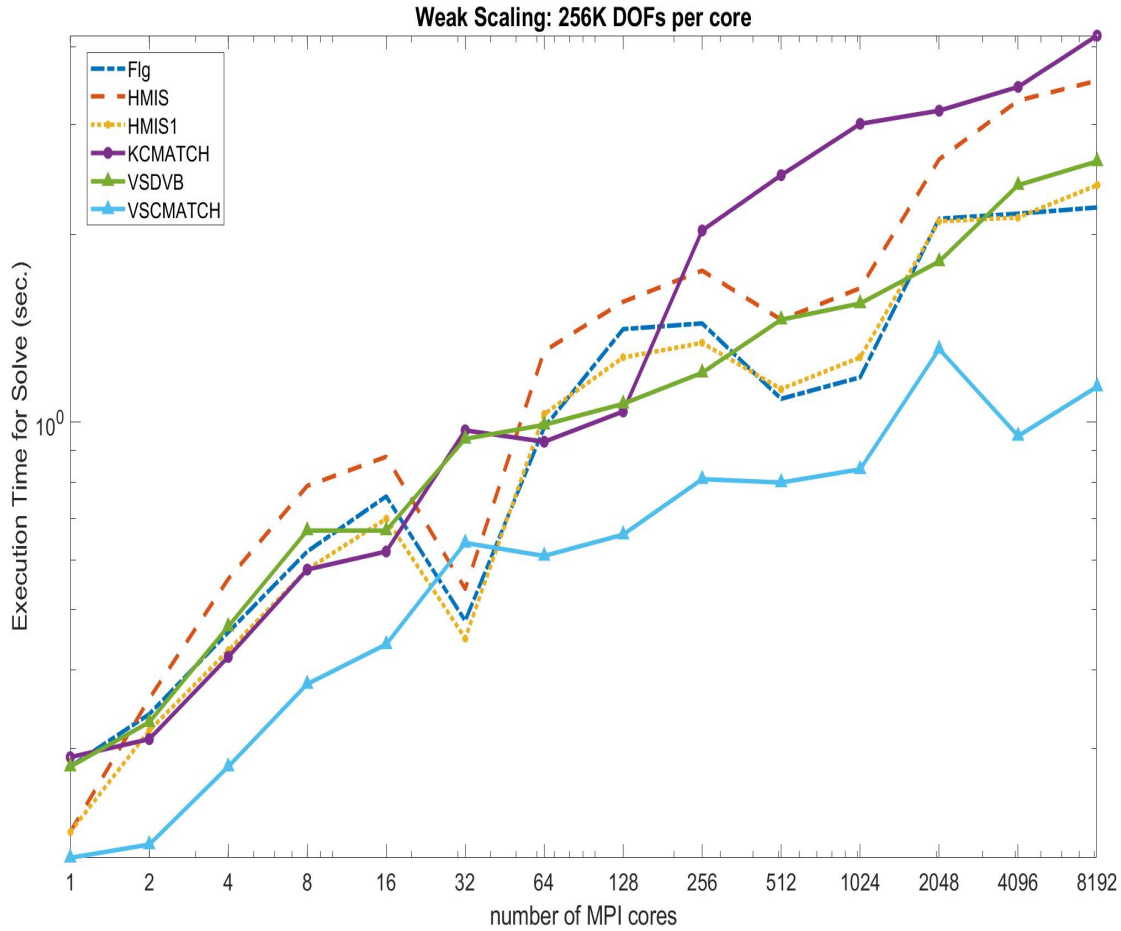
		AMG4PSBLAS			Hypre		
np	$n/10^6$	KCMATCH	VSCMATCH	VSDVB	Flg	HMIS	HMIS1
1	0.256	12	7	11	6	6	12
2	0.512	12	7	12	7	9	15
2^2	1.036	12	7	13	7	12	17
2^3	2.048	12	7	14	8	13	17
2^4	4.075	12	8	14	8	14	20
2^5	8.049	13	9	15	8	14	20
2^6	16.384	12	8	15	9	16	22
2^7	32.604	12	8	15	10	18	25
2^8	63.917	13	9	16	10	20	27
2^9	131,072	14	8	18	11	22	29
2^{10}	256,000	15	8	17	12	25	32
2^{11}	511,335	16	12	21	13	29	37
2^{12}	1024,192	15	8	26	13	35	40
2^{13}	2097,152	16	9	27	14	37	44

Table: Number of iterations for solve

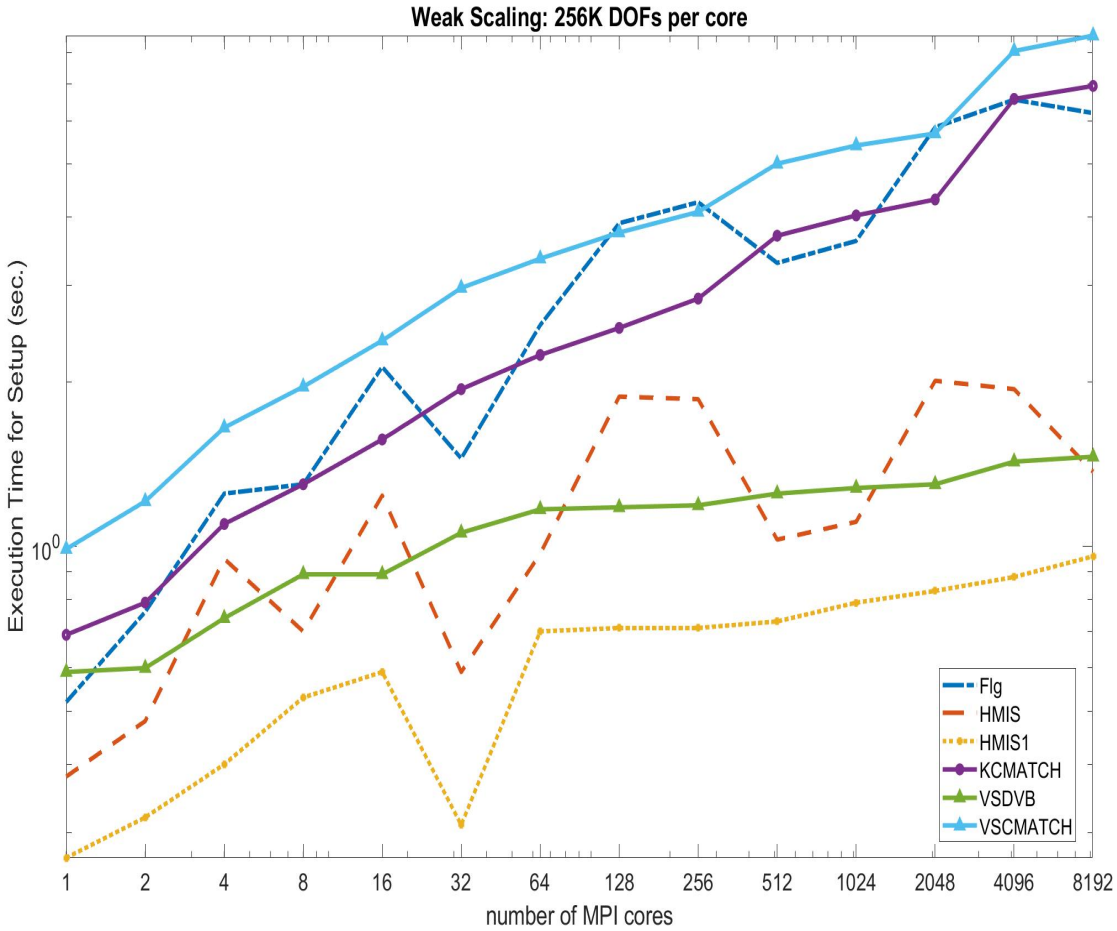
Weak scaling (256K dofs per core): operator complexity



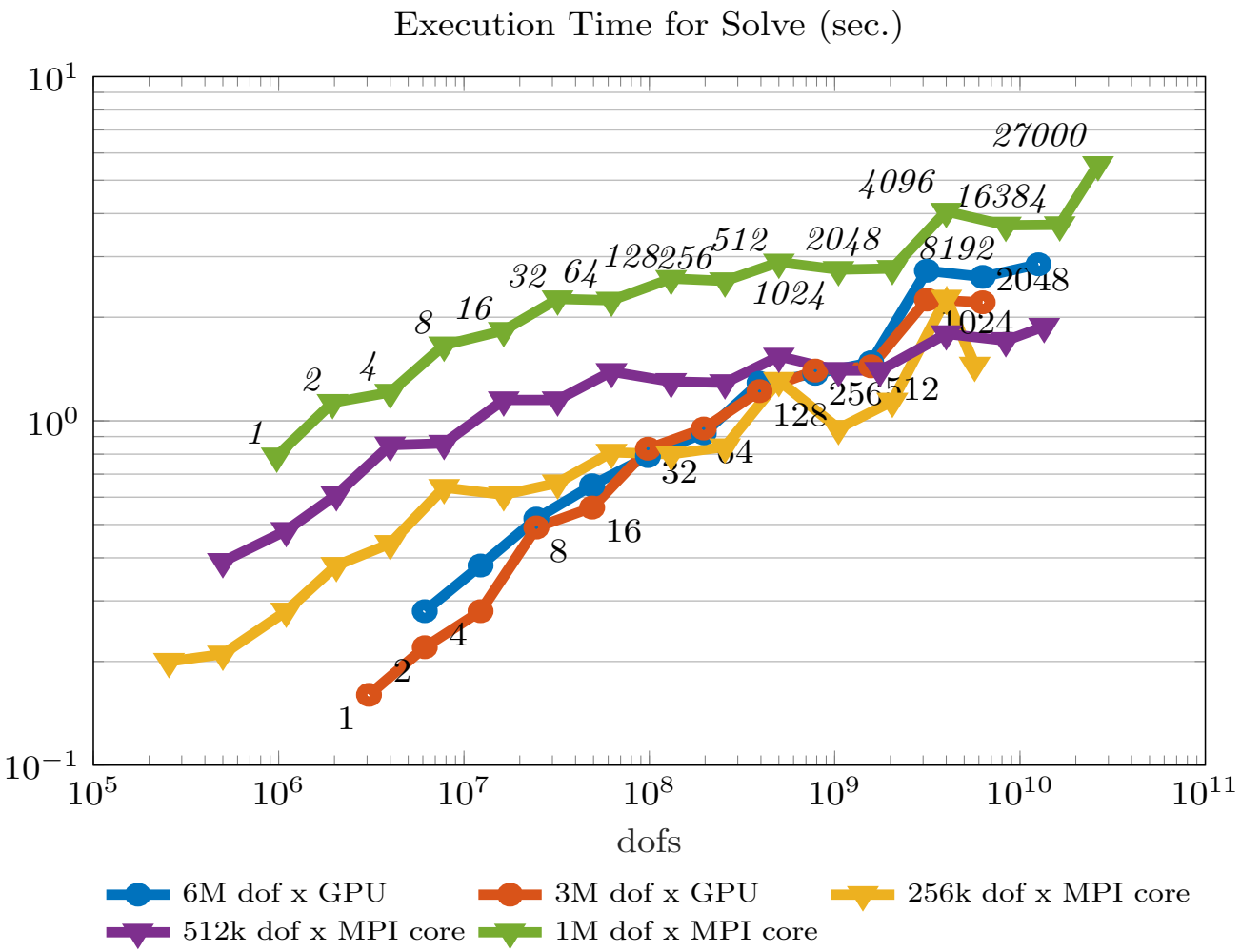
Weak scaling (256K dofs per core): solve time



Weak scaling (256K dofs per core): setup time



Results at extreme scale: MPI vs hybrid MPI-CUDA



Concluding Remarks and Work in Progress

- PSCToolkit is a new software framework addressing scalability, flexibility and robusteness for high-performance scientific computing at extreme scale
- the new parallel coarsening algorithm based on compatible weighted matching, used in conjunction with smoothed prolongators and highly parallel smoothers, shows algorithmic and implementation scalability
- we solve systems with size larger than 10^{10} on current pre-exascale computers, embedding hybrid CPU-GPU nodes, in less than 3 seconds
- scalability results and comparison with available software demonstrates the validity of our approaches both in terms of algorithms and in terms of software development
- integration and testing within very large scale wind simulations and hydrologycal applications, in collaborations with BSC and JSC, is work in progress

Thanks for Your Attention

This work was performed with support of the European Union's Horizon 2020 research and innovation programme under grant agreement N. 824158