A fast parallel Poisson solver for Scrape-Off-Layer

Kab Seok Kang kskang@ipp.mpg.de

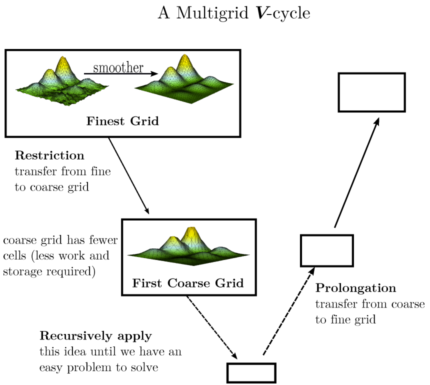
High Level Support Team (HLST)
Department of Computational Plasma Physics
Max-Planck-Institut für Plasmaphysik, EURATOM Association,
Boltzmannstraße 2, D-85748 Garching, Germany

OUTLINE

- Fast parallel solver
  - Multigrid and DDM
  - Parallelization issues
  - Modern HPC
- Model problems
  - Poisson problem
  - Hexagonal domain
  - Scrape-off-Layer
- Numerical experiments
  - Helios
  - Reduced core and OpenMP/MPI hybrid
  - Scaling properties
- Acknowledgements

Multigrid method: Idea

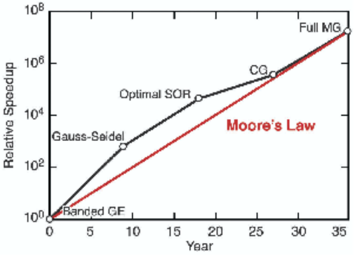
– Motivation: Simple iterative method reduces well high frequency error and low frequency error is well approximated by coarser level problem



Multigrid method: Properties

– Well-known and well-analyzed fast solver and preconditioner
– The required number of iterations is fixed for many cases

Method	Storage	Flops
GE (banded)	n^5	n^7
Gauss-Seidel	n^3	$n^5 \log n$
Optimal SOR	n^3	$n^4 \log n$
CG	n^3	$n^{3.5} \log n$
Full MG	n^3	n^3



– Smoothing operators: On each level
– Prolongations and restrictions: Intergrid transfer operators
– Lowest solver: On lowest level, CGM, GMRES, Direct solver,

Domain Decomposition Method: Idea

– Divide sub-domains and solve problems only on it
  $\rightarrow$ Naturally fit to distributed computers
– Overlapping DDM: Schwarz method
  - Schwarz method: Solve local problems
    on each sub-domain with Dirichlet BC
  - Multipicative method: Alternatively solve
  - Additive method: Solve local problems on the same time.
    Simple and mainly use as a precondtioner
– Nonoverlapping methods: Use conditions on boundaries
  of the sub-domains
  - Neumann-Neumann and Dirichlet-Dirichlet
  $\rightarrow$ Good for discontinuous or many parts problems
  - BDD, BDDC, FETI, FETI-DP
– Can be used any discretization method, FEM, FVM, and DG.

Domain Decomposition Method: Properties

– One-level DDM: Depends on the number of subdomains
  - Large δ: Good condition number, more cost of the data communication
  - Small δ: Minimal data communication cost, similar the block Jacobi iteration, not efficient preconditioner

– Two level DDM: Not depending on the number of subdomains, Only on the ratio of the fine and coarse level meshes
  - Overlapping: Need to solve the coarse level problem
  - Nonoverlapping: BDDC and FETI-DP
  - Solve local and global coarser problem
  - Local (Dirichlet and Neumann): Need to communicate boundary data with neighborhood subdomains
  - Global coarser: Contributions from and uses by all Size of system: according to the number of subdomains (the number of cores) $\rightarrow$ direct, CGM, Multigrid

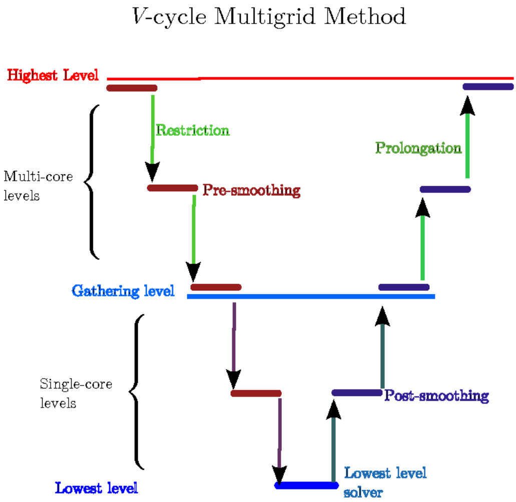
Issues

– Lower levels: needs more data communication time in comparison to computing time
  → Bottleneck on the parallel computers
  → Use V-cycle scheme as a solver and as a preconditioner
– Gauss-Seidel smoother: Prefered, but hard to parallelize
  → Localize: Perform the Gauss-Seidel iteration exclusively on each core, no data communication between cores in one Gauss-Seidel cycle
– Lowest level solver: depends on the problem
  - Problem size: can be less than the number of cores
  - Single core version is better than parallel version
  → Same as global coarser problem of two-level nonoverlapping DDM
  - Bigger problem size: Need more iterations for iterative methods, such as CGM, GMRES

Parallel multigrid method with reduced cores

- Reduce the number of execution cores on a certain coarser level $\rightarrow$ Use only one core
- Gather data to one core, solve, and scatter to all core $\rightarrow$ Only one core is busy and others idle
- Gather data on each core and solve on every core $\rightarrow$ Don't need scattering step
- Use `MPI_Allreduce` :
  Combine `MPI_Reduce` and `MPI_Bcast`
  $\rightarrow$ Better performance depending on the MPI implementation

Gathering data algorithm



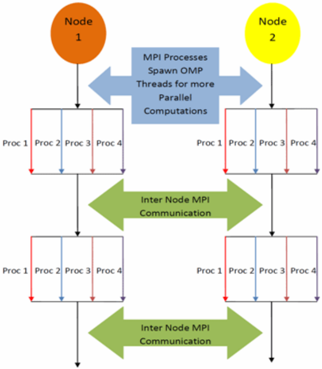
Domain Decomposition Method as a lowest solver

– DDM: Can be used as a lowest solver of the multigrid method
– Standard problem: the size of the lowest problem might be
as small as possible
$\rightarrow$ Reduced core algorithm is better
– Many problems have restrictions on the lowest level:
- irregular shape domain, nonsymmetric problem, ...
- Need more iterations for iterative method and
not fit for direct method
$\rightarrow$ DDM might be the better than other method

OpenMP/MPI hybrid: Implementation

– OpenMP: a standard for shared-memory systems

- Launch one process per node → Need launch time
- Have each process fork one thread (or maybe more) per core
- Share data using shared memory
- Can't share data with a different processor (node) (except maybe via file I/O)



– Hybridization: Each MPI process to launch multiple OpenMP threads that can share local memory

Trends of HPC

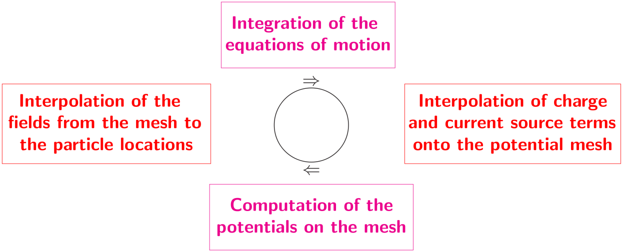
– 100 million to 1 billion cores
– Clock rates of 1 to 2 GHz (reduced energy usage):
  ARM-based (Mont-Blanc project in EU)
– Multi-threaded, fine-grained concurrrency of 10- to 100- way concurrency per core (computational accelerator):
  GPU (OpenACC), MIC (OepnMP), ...
– Hundreds of cores per die: multicore, multisocket
– Active power management: Max 20MW for the computer
– New design: 3D packaging of dies for stacks of four to ten dies, each including DRAM, cores, and networking

Multicore-multisocket CPU and accelerator

– Intel Xeon E5-2692 12C 2.2GHz: Tianhe-2(#1)
  $\rightarrow$ 2 Sockets (12 cores) = 24 cores per node
– Intel Xeon E5-2680 8C 2.7GHz: Stampede (#6), SuperMUC(#9)
  $\rightarrow$ 2 Sockets (8 cores) = 16 cores per node
– Opteron 6274 16C 2.200GHz: Titan - Cray XK7(#2),
  $\rightarrow$ 16 cores per node
– Power BQC 16C 1.60 GHz: Sequoia - BlueGeneQ(#3), Mira(#5), JUQUEEN(#7), VULCAN(#8)
  $\rightarrow$ 16 cores per node
– SPARC64 VIIIfx 2.0GHz: K computer(#4)
  $\rightarrow$ 8 cores per node
– Accelerator:
- GPU: NVIDIA K20x(#2), NVIDIA 2050 (Tianhe-1A, #10)
- MIC: Xeon Phi31S1P(Tianhe-2,#1), Xeon PhiSE10P(Stampede,#6)

Model problem

Schematic diagram of the PIC method



– Computing potentials on each time step: The second order PDE problem on a domain with Dirichlet boundary condition

$$\begin{cases} (A - \nabla \cdot B\nabla)\, u = f, & \text{in } \Omega \\ u = 0, & \text{on } \partial\Omega \end{cases}$$

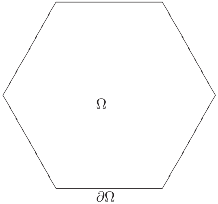
Purpose

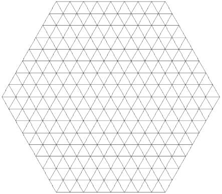
– Solve the 2nd order PDE in Plasma Physics simulation codes for Tokamak experiments
– Solution is sought at each time step $\rightarrow$ less than 0.1 sec

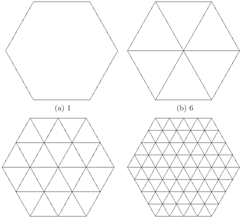
Tokamak	ASDEX	JET	ITER	DEMO
DoF	2M	8M	32M	?

–Hexagonal domain: For GEMT project (gyrofluid and reduced MHD and gyrokinetic models)
–Scrape-off-Layer: Prediction of plasma particle and energy loads to the plasma facing components (PFC), estimation of corresponding PFC erosion rates and impurity and dust generation rates

Discretization and parallelization



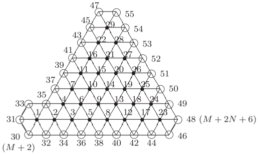




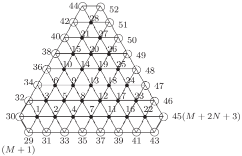
(a) 1 (b) 6 (c) 24 (d) 96

– Linear Finite element method or Finite volume method
– Triangulation with regular triangles
– Divide a regular hexagonal domain with regular triangular sub-domains
– Limited number of cores: 1, 6, 24, 96, 384, ...
– Determine where the boundary nodes of the sub-domain are included.

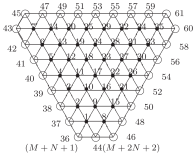
Communications



Type I: 0,6,9,12,15,18,21,24, ...



Type II: 1, 2, 3, 4, 5, 8, 11, ...

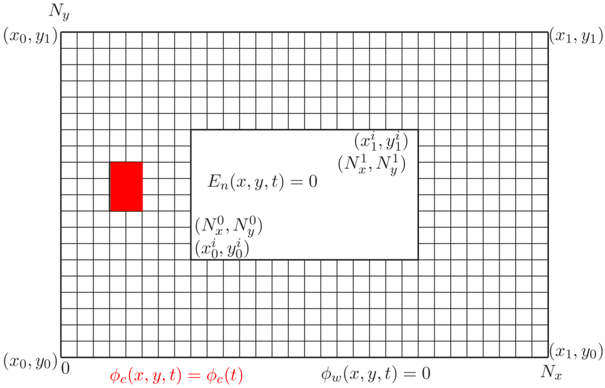


Type III: 7, 10, 13, 16, 19, 22, 25, ...

- Consisted by Real (•) and Ghost (◯) nodes.
- Classify three types of sub-domains.
- Need **five steps** for data communication for matrix-vector

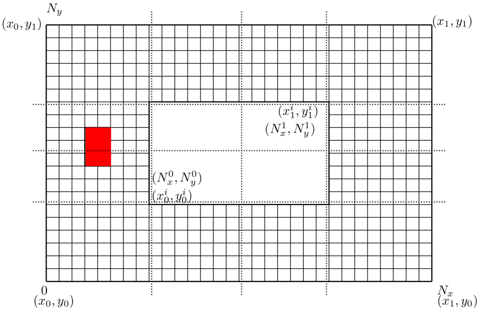
Scarape-off-Layer: domain

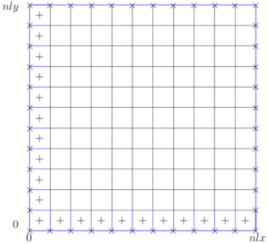
$$-\left[\frac{\partial}{\partial x}\epsilon(x,y)\frac{\partial}{\partial x}+\frac{\partial}{\partial y}\epsilon(x,y)\frac{\partial}{\partial y}\right]\phi(x,y,t)=\rho(x,y,t)$$



Scarape-off-Layer: parallelization

4×4 subdomains, use real and ghost nodes and cells





HELIOS

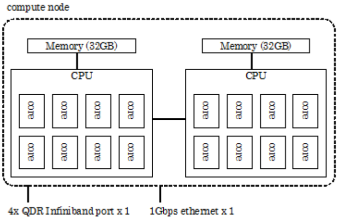
– IFERC: The International Fusion Energy Centre is located at Rokkasho, Japan
  - EU(F4E)–Japan Broader Approach collaboration

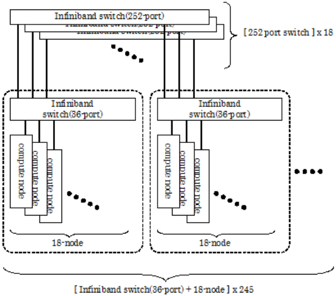
– The Computaional Simulation Centre (CSC): To exploit large-scale and high performance fusion simulations

– HELIOS: 4410 Bullx B510 Blades, 70,000 cores
  - 1.3 Pflops peak performance
  - No. 20 in Top500, June 2013
  - Xeon E5-2680 8C 2.7 GHz per node
  - Interconnection: Infiniband
  - Upgrade with MIC accelerator

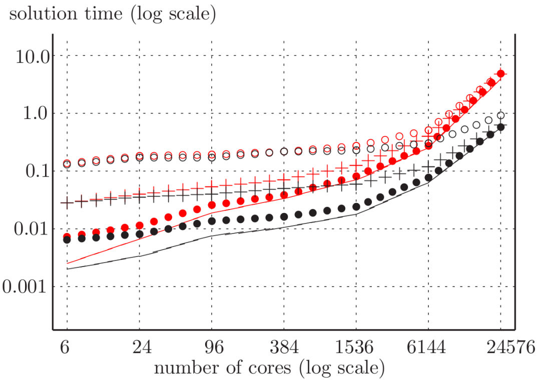
Architecture of node

– Two sockets (8 cores) per node





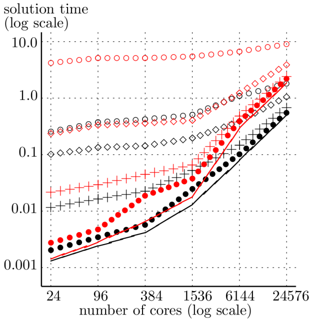
MG with gathering data



— : 2.2K DoF/core
● ● ●: 8.5K DoF/core
+ + +: 33K DoF/core
○ ○ ○: 132K DoF/core
with gathering in black
without gathering in red

DDM: # of iterations and weak scaling

	# cores	24	96	384	1536	6144	24576
	levels	4	5	6	7	8	9
	CGM	55	110	213	411	802	1591
1/8	PCGMG	5	5	5	5	5	5
	FETIDP	12	15	16	16	16	16
	BDDC	7	8	8	8	8	8
1/16	FETIDP	14	17	19	20	19	19
	BDDC	8	9	10	10	10	9
1/32	FETIDP	16	20	22	23	23	23
	BDDC	9	11	11	11	11	11
1/64	FETIDP	18	23	24	26	26	26
	BDDC	10	13	13	13	13	13
	levels	8	9	10	11	12	13
	CGM	802	1591	3056	5614	10965	22000
1/128	PCGMG	5	5	5	5	5	5
	FETIDP	12	15	16	16	16	16
	BDDC	7	8	8	8	8	8

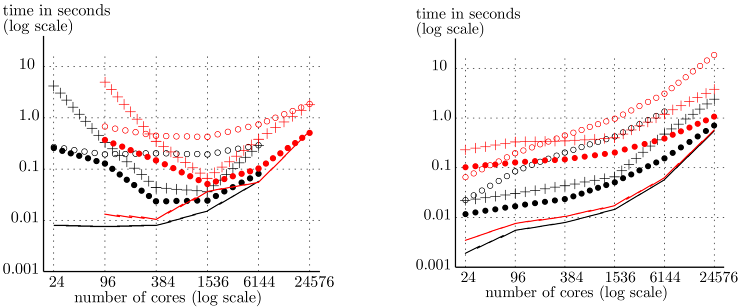


FETI-DP (black), BDDC (red)

32 DoF (solid), 500 DoF (+), 8K DoF (○)

128 DoF (●) 2K DoF (◇) per core

Comparison as a lowest solvers



Strong scaling

Weak scaling

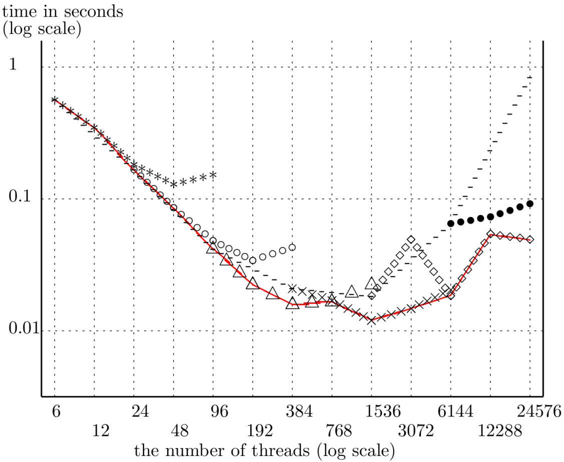
500K DoF (black), 2M DoF (red)

590 DoF (black), 2200 DoF (red) per core

PCGM (solid), FETI-DP (●), BDDC(+), CGM (○)

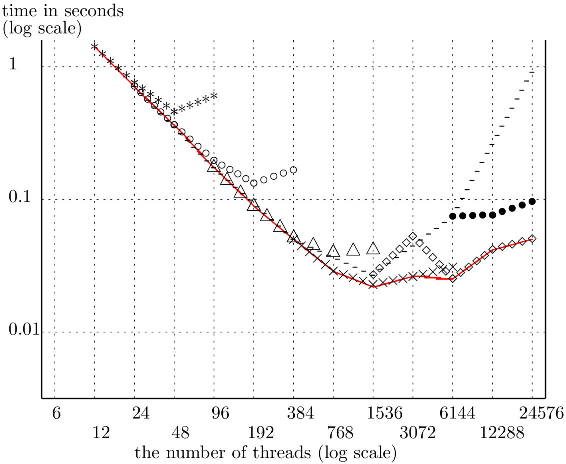
- FETI-DP and BDDC : better than CGM when the number of cores is large

Strong scaling: OpenMP/MPI (3.1M DoF)



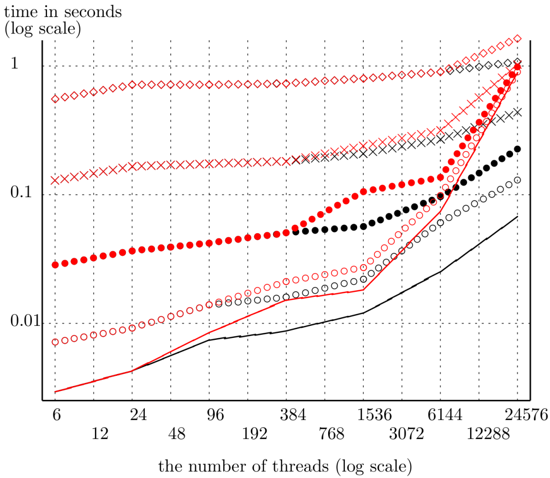
– –: pure MPI, the best in red

Strong scaling: OpenMP/MPI (12M DoF)



– –: pure MPI, the best in red

Weak scaling



The best (black)

Pure MPI(red)

2.2K DoF (solid)

8.5K DoF (○)

3.1M DoF (●)

13M DoF (×)

50M DoF (◇)

per core

Conclusion and future works

– Multigrid method with gathering data has been made performance improvement
– Multigrid methgod is the fastest solver in comparison FETI-DP, BDDC, and CGM
– FETI-DP is better scaling property than CGM
  $\rightarrow$ Might be lowest solver for SOL domain
– Small number of Dof per core and large number of MPI tasks
  $\rightarrow$ Improved the performance by using hybrid OpenMP/MPI

$*$ Future work

– Implement SOL-domain
– Analyze the performances of Multigrid, FETI-Dp, BDDC and OpenMP/MPI hybridization

Acknowledgements

This work was carried out using the HELIOS supercomputer system at Computational Simulation Centre of International Fusion Energy Research Centre (IFERC-CSC), Aomori, Japan, under the Broader Approach collaboration between Euratom and Japan, implemented by Fusion for Energy and JAEA. I would like to thank R. Hatzky and other HLST team members, B. Scott, and D. Tskhakaya for helpful discussions.

Thank you for your attention!!