Communication/Synchronization Avoiding/Reducing in Krylov Iterative Solvers on Multicore/Manycore Clusters
Early Case Studies on Oakforest-PACS with Intel Xeon Phi（抜粋）

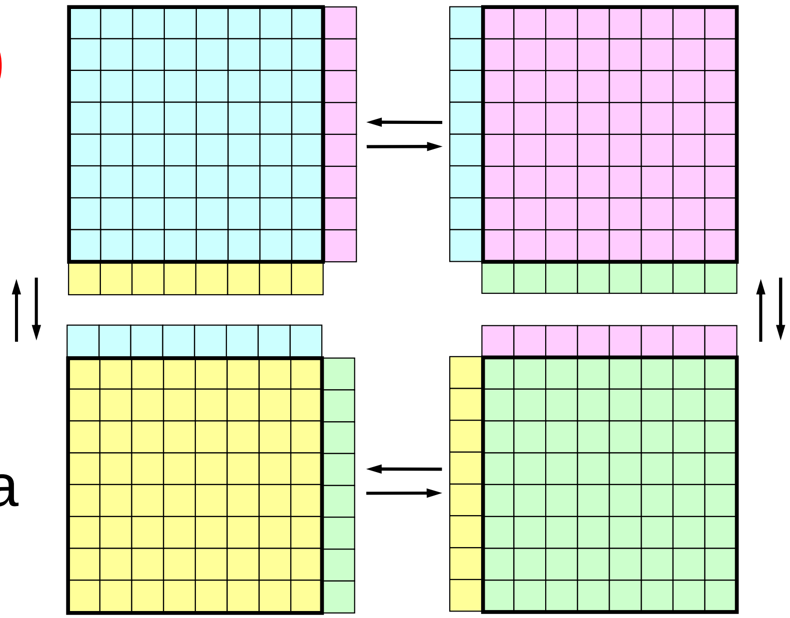
Kengo Nakajima, Toshihiro Hanawa

Information Technology Center, The University of Tokyo

Berkeley Lab Computing Sciences Seminar
January 20, 2017

- Communication/Synchronization Avoiding/Reducing in Krylov Iterative Solvers
- Pipelined CG: Background
- Pipelined CG: Results
- Communication-Computation Overlapping
- Summary
- **Preliminary Study on OFP**

Parallel (Krylov) Iterative Solvers



- Both of convergence (robustness) and efficiency (single/parallel) are important
- Communications needed
  - SpMV (P2P communications, MPI_Isend/Irecv/Waitall): Local Data Structure with HALO
  - Dot-Products (MPI_Allreduce)
    - ✓ effect of latency
  - Preconditioning (up to algorithm)
- Remedy for Robust Parallel ILU Preconditioner
  - Additive Schwartz Domain Decomposition
  - HID (Hierarchical Interface Decomposition, based on global nested dissection) [Henon & Saad 2007], ext. HID [KN 2010]

Communication/Synchronization Avoiding/Reducing/Hiding

for Parallel Preconditioned Krylov Iterative Methods

- SpMV
  - Overlapping of Computations & Communications
  - Matrix Powers Kernel
- Dot Products
  - Pipelined Methods
  - Gropp's Algorith,

Algorithm 1 Preconditioned CG

$r_0 := b - Ax_0;\quad u_0 := M^{-1}r_0;\quad p_0 := u_0$
for $i = 0, \ldots$ **do**
  $\underline{s := Ap_i}$
  $\alpha := \underline{(r_i, u_i) \,/\, (s, p_i)}$
  $x_{i+1} := x_i + \alpha p_i$
  $r_{i+1} := r_i - \alpha s$
  $u_{i+1} := M^{-1} r_{i+1}$
  $\beta := \underline{(r_{i+1}, u_{i+1}) \,/\, (r_i, u_i)}$
  $p_{i+1} := u_{i+1} + \beta p_i$
end for

Communication Avoiding/Reducing Algorithms for Sparse Linear Solvers utilizing Matrix Powers Kernel

- Matrix Powers Kernel: Ax, A^2x, A^3x ...
- Krylov Iterative Method without Preconditioning
  - Demmel, Hoemmen, Mohiyuddin etc. (UC Berkeley)
- *s-step* method
  - Just one P2P communication for each Mat-Vec during *s* iterations. Convergence may become unstable for large *s*.
- Communication Avoiding ILU0 (CA-ILU0) [Moufawad & Grigori, 2013]
  - First attempt to CA preconditioning
  - Nested dissection reordering for limited geometries (2D FDM)
- Generally, it is difficult to apply Matrix Powers Kernel to preconditioned iterative solvers

- Communication/Synchronization Avoiding/Reducing in Krylov Iterative Solvers
- **Pipelined CG: Background**
- Pipelined CG: Results
- Communication-Computation Overlapping
- Summary

Hiding Overhead by Collective Comm. in Krylov Iterative Solvers

- Dot Products in Krylov Iterative Solvers
  - MPI_Allreduce: Collective Communications
  - Large overhead with many nodes
- Pipelined CG [Ghysels et al. 2014]
  - Utilization of asynchronous collective communications (e.g. MPI_Iallreduce) supported in MPI-3 for hiding such overhead.
  - Algorithm is kept, but order of computations is changed
  - [Reference] P. Ghysels et al., Hiding global synchronization latency in the preconditioned Conjugate Gradient algorithm, Parallel Computing 40, 2014
  - When I visited LBNL in September 2013, Dr. Ghysels asked me to evaluate his idea in my parallel multigrid solvers

4 Algorithms [Ghysels et al. 2014]

- Alg.1 Original Preconditioned CG
- Alg.2 Chronopoulos/Gear
  - 2 dot products are combined in a single reduction
- Alg.3 Pipelined CG (MPI_Iallreduce)
- Alg.4 Gropp's asynchronous CG (MPI_Iallreduce)

- Algorithm itself is not different from the original one
  - Recurrence Relations: 漸化式
  - Order of computation changed -> Rounding errors are propagated differently
    - Convergence may be affected (not happened in my case)
    - update of r= b-Ax needed at every 50 iterations (original paper)

Original Preconditioned CG (Alg.1)

Original Preconditioned CG (Alg.1)

$r_0 := b - Ax_0; \quad u_0 := M^{-1} r_0; \quad p_0 := u_0$
for $i = 0, \ldots$ **do**
  $s := Ap_i$
  $\alpha := (r_i, u_i) \,/\, (s, p_i)$
  $x_{i+1} := x_i + \alpha p_i$
  $r_{i+1} := r_i - \alpha s$
  $u_{i+1} := M^{-1} r_{i+1}$
  $\beta := (r_{i+1}, u_{i+1}) \,/\, (r_i, u_i)$
  $p_{i+1} := u_{i+1} + \beta p_i$
end for

$$
\begin{aligned}
s_i &= Ap_i \\
x_{i+1} &= x_i + \alpha_i p_i \\
r_{i+1} &= b - Ax_{i+1} = b - Ax_i - \alpha_i Ap_i \\
&= r_i - \alpha_i Ap_i = r_i - \alpha_i s_i
\end{aligned}
$$

Chronopoulos/Gear CG (Alg.2)

2 dot products are combined into a single reduction

s-step algorithm with s=1

Chronopoulos/Gear CG (Alg.2)

$r_0 := b - Ax_0; \quad u_0 := M^{-1}r_0; \quad w_0 := Au_0$

$\alpha_0 := (r_0, u_0) / (w_0, u_0); \quad \beta_0 := 0; \quad \gamma_0 := (r_0, u_0)$

for $i = 0, \ldots$ **do**

$\quad p_i := u_i + \beta_i p_{i-1}$

$\quad s_i := w_i + \beta_i s_{i-1}$

$\quad x_{i+1} := x_i + \alpha_i p_i$

$\quad r_{i+1} := r_i - \alpha_i s_i$

$\quad u_{i+1} := M^{-1} r_{i+1}$

$\quad w_{i+1} := Au_{i+1}$

$\quad \gamma_{i+1} := (r_{i+1}, u_{i+1})$

$\quad \delta := (w_{i+1}, u_{i+1})$

$\quad \beta_{i+1} := \gamma_{i+1} / \gamma_i$

$\quad \alpha_{i+1} := \gamma_{i+1} / (\delta - \beta_{i+1}\gamma_{i+1}/\alpha_i)$

end for

$$\underline{s_i = Au_i + \beta_i s_{i-1}}, \; p_i = u_i + \beta_i p_i \Leftrightarrow s_i = Ap_i$$

$$x_{i+1} = x_i + \alpha_i p_i$$

$$r_{i+1} = b - Ax_{i+1} = b - Ax_i - \alpha_i Ap_i$$

$$= r_i - \alpha_i Ap_i = r_i - \alpha_i s_i$$

- 2 dot products combined

- $s_i = Ap_i$ is not computed explicitly: by recurrence

Pipelined Chronopoulos/Gear (No Preconditioning)

Pipelined Chronopoulos/Gear (No Precond.)

$r_0 := b - Ax_0; \quad w_0 := Ar_0$
for $i = 0, \ldots$ **do**
  $\gamma_i := (r_i, r_i)$
  $\delta := (w_i, r_i)$
  $q_i := Aw_i$
  if $i > 0$ **then**
    $\beta_i := \gamma_i / \gamma_{i-1}; \quad \alpha_i := \gamma_i / (\delta - \beta_i \gamma_i / \alpha_{i-1})$
  else
    $\beta_i := 0; \quad \alpha_i := \gamma_i / \delta$
  end if
  $z_i := q_i + \beta_i z_{i-1}$
  $s_i := w_i + \beta_i s_{i-1}$
  $p_i := r_i + \beta_i p_{i-1}$
  $x_{i+1} := x_i + \alpha_i p_i$
  $r_{i+1} := r_i - \alpha_i s_i$
  $w_{i+1} := w_i - \alpha_i z_i$
end for

- Global synchronization of dot products are overlapped with SpMV

$$u_i = r_i, \; w_i = Au_i = Ar_i$$

$$Ar_{i+1} = Ar_i - \alpha_i As_i \Rightarrow w_{i+1} = w_i - \alpha_i As_i$$

$$As_i = Aw_i + \beta_i As_{i-1} \Rightarrow z_i \left(= As_i = A^2 p_i\right) = Aw_i + \beta_i z_{i-1}$$

$$q_i = Aw_i = A^2 r_i$$

$$= r_i - \alpha_i Ap_i = r_i - \alpha_i s_i$$

Preconditioned Pipelined CG (Alg.3)

Preconditioned Pipelined CG (Alg.3)

$r_0 := b - Ax_0; \quad u_0 := M^{-1}r_0; \quad w_0 := Au_0$

for $i = 0, \ldots$ **do**

$\gamma_i := (r_i, u_i)$

$\delta := (w_i, u_i)$

$m_i := M^{-1}w_i$

$n_i := Am_i$

if $i > 0$ **then**

$\beta_i := \gamma_i/\gamma_{i-1}; \quad \alpha_i := \gamma_i/\left(\delta - \beta_i\gamma_i/\alpha_{i-1}\right)$

else

$\beta_i := 0; \quad \alpha_i := \gamma_i/\delta$

end if

$z_i := n_i + \beta_i z_{i-1}$

$q_i := m_i + \beta_i q_{i-1}$

$s_i := w_i + \beta_i s_{i-1}$

$p_i := u_i + \beta_i p_{i-1}$

$x_{i+1} := x_i + \alpha_i p_i$

$r_{i+1} := r_i - \alpha_i s_i$

$u_{i+1} := u_i - \alpha_i q_i$

$w_{i+1} := w_i - \alpha_i z_i$

end for

- Global synchronization of dot products are overlapped with SpMV and Preconditioning

$$\gamma_i = (u_i, u_i)_M = (Mu_i, u_i) = (r_i, u_i)$$

$$\delta_i = \left(M^{-1}Au_i, u_i\right)_M = (Au_i, u_i) = (w_i, u_i)$$

$$M^{-1}r_{i+1} = M^{-1}r_i - \alpha_i M^{-1}s_i \Rightarrow u_{i+1} = u_i - \alpha_i q_i \quad \left(q_i = M^{-1}s_i\right)$$

$$M^{-1}s_{i+1} = M^{-1}w_i + \beta_i M^{-1}s_i \Rightarrow q_{i+1} = M^{-1}w_i + \beta_i q_i$$

$$Au_{i+1} = Au_i - \alpha_i Aq_i \Rightarrow w_{i+1} = w_i - \alpha_i Aq_i$$

$$Aq_i = AM^{-1}w_i + \beta_i Aq_{i-1} \Rightarrow z_i = Am_i + \beta_i z_{i-1}$$

$$\left(m_i = M^{-1}w_i = M^{-1}Au_i = M^{-1}AM^{-1}r_i,\ z_i = Aq_i\right)$$

Gropp's Asynchronous CG (Alg.4) Smaller Computations than Alg.3

Gropp's Asynchronous CG (Alg.4)

$r_0 := b - Ax_0; \quad u_0 := M^{-1} r_0; \quad p_0 := u_0; \quad s_0 := Ap_0; \quad \gamma_0 := (r_0, u_0)$

for $i = 0, \ldots$ **do**

$\quad \delta := (p_i, s_i)$

$\quad q_i := M^{-1} s_i$

$\quad \alpha_i := \gamma_i / \delta$

$\quad x_{i+1} := x_i + \alpha_i p_i$

$\quad r_{i+1} := r_i - \alpha_i s_i$

$\quad u_{i+1} := u_i - \alpha_i q_i$

$\quad \gamma_{i+1} := (r_{i+1}, u_{i+1})$

$\quad w_{i+1} := A u_{i+1}$

$\quad \beta_{i+1} := \gamma_{i+1} / \gamma_i$

$\quad p_{i+1} := u_{i+1} + \beta_{i+1} p_i$

$\quad s_{i+1} := w_{i+1} + \beta_{i+1} s_i$

end for

- Definition of δ is different from that of Alg.3
- Global synchronization of dot products are overlapped with SpMV and Preconditioning

W. Gropp, Update on Libraries for Blue Waters.
http://jointlab-pc.ncsa.illinois.edu/events/workshop3/pdf/presentations/Gropp-Update-on-Libraries.pdf
Presentation Material (not a paper, article)

Implementation (Alg.4)

```
!C
!C +---------------+
!C | Delta= (p,s)  |
!C +---------------+
!C===
      DL0= 0.d0
!$omp parallel do private(i) reduction(+:DL0)
      do i= 1, 3*N
        DL0= DL0 + P(i)*S(i)
      enddo
      call MPI_Iallreduce (DL0, Delta, 1, MPI_DOUBLE_PRECISION,            &
     &                    MPI_SUM, MPI_COMM_WORLD, req1, ierr)
!C===

!C
!C +-----------------+
!C | {q}= [Minv]{s}  |
!C +-----------------+
!C===

（前処理：省略）

!C===

      call MPI_Wait (req1, sta1, ierr)
```

Gropp's Asynchronous CG (Alg.4)

$r_0 := b - Ax_0;\quad u_0 := M^{-1}r_0;\quad p_0 := u_0;\quad s_0 := Ap_0;\quad \gamma_0 := (r_0, u_0)$
for $i = 0, \ldots$ **do**
    $\boxed{\begin{array}{l}\delta := (p_i, s_i)\\ q_i := M^{-1}s_i\\ \alpha_i := \gamma_i/\delta\end{array}}$
    $x_{i+1} := x_i + \alpha_i p_i$
    $r_{i+1} := r_i - \alpha_i s_i$
    $u_{i+1} := u_i - \alpha_i q_i$
    $\gamma_{i+1} := (r_{i+1}, u_{i+1})$
    $w_{i+1} := Au_{i+1}$
    $\beta_{i+1} := \gamma_{i+1}/\gamma_i$
    $p_{i+1} := u_{i+1} + \beta_{i+1}p_i$
    $s_{i+1} := w_{i+1} + \beta_{i+1}s_i$
end for

Pipelined CR (Conjugate Residuals)

Preconditioned Pipelined CR

$r_0 := b - Ax_0; \quad u_0 := M^{-1} r_0; \quad w_0 := Au_0$
for $i = 0, \ldots$ **do**
  $m_i := M^{-1} w_i$
  $\gamma_i := (w_i, u_i)$
  $\delta := (m_i, w_i)$
  $n_i := Am_i$
  if $i > 0$ **then**
    $\beta_i := \gamma_i / \gamma_{i-1}; \quad \alpha_i := \gamma_i / \left(\delta - \beta_i \gamma_i / \alpha_{i-1}\right)$
  else
    $\beta_i := 0; \quad \alpha_i := \gamma_i / \delta$
  end if
  $z_i := n_i + \beta_i z_{i-1}$
  $q_i := m_i + \beta_i q_{i-1}$
  $p_i := u_i + \beta_i p_{i-1}$
  $x_{i+1} := x_i + \alpha_i p_i$
  $u_{i+1} := u_i - \alpha_i q_i$
  $w_{i+1} := w_i - \alpha_i z_i$
end for

- Global synchronization of dot products are overlapped with SpMV

Amount of Computations

DAXPY could very smaller than (sophisticated) preconditioning

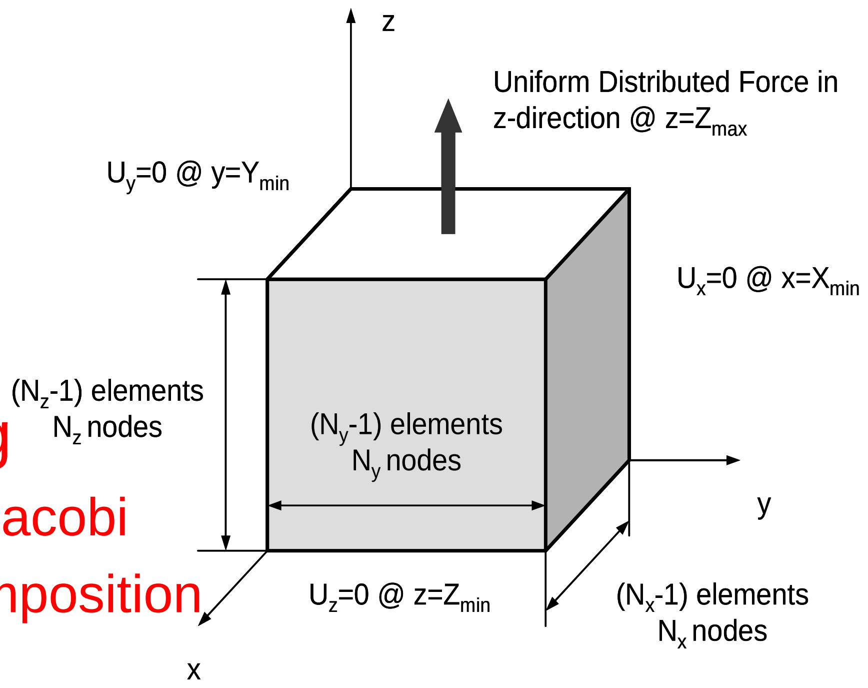
		SpMV	Precond.	Dot Prod.	DAXPY
1	Original CG	1	1	2+1	3
2	Chronopoulos/ Gear	1	1	2+1	4
3	Pipelined CG	1	1	2+1	8
4	Grppp's Algorithm	1	1	2+1	5
	Pipelined CR	1	1	2+1	6

+1 for
residual norm

- Communication/Synchronization Avoiding/Reducing in Krylov Iterative Solvers
- Pipelined CG: Background
- **Pipelined CG: Results**
- Communication-Computation Overlapping
- Summary

GeoFEM/Cube

- Parallel FEM Code (& Benchmarks)
- 3D-Static-Elastic-Linear (Solid Mechanics)
- Performance of Parallel Precond. Iterative Solvers
  - 3D Tri-linear Elements
  - SPD matrices: CG solver
  - Fortran90+MPI+OpenMP
  - Distributed Data Structure
  - Localized SGS Preconditioning
    - Symmetric Gauss-Seidel, Block Jacobi
    - Additive Schwartz Domain Decomposition
  - Reordering by CM-RCM: RCM
  - MPI, OpenMP, OpenMP/MPI Hybrid

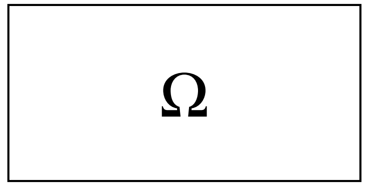


Overlapped Additive Schwartz Domain Decomposition Method

Stabilization of Localized Preconditioning: ASDD

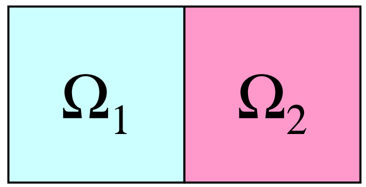
Global Operation

$$Mz = r$$



Local Operation

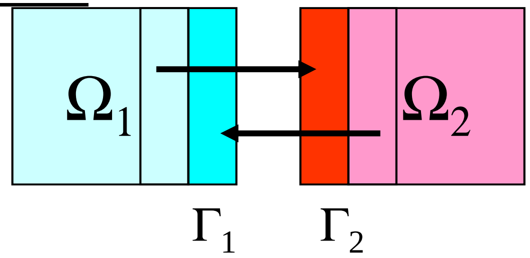
$$z_{\Omega_1} = M_{\Omega_1}^{-1} r_{\Omega_1}, \quad z_{\Omega_2} = M_{\Omega_2}^{-1} r_{\Omega_2}$$



Global Nesting Correction: Repeating -> Stable

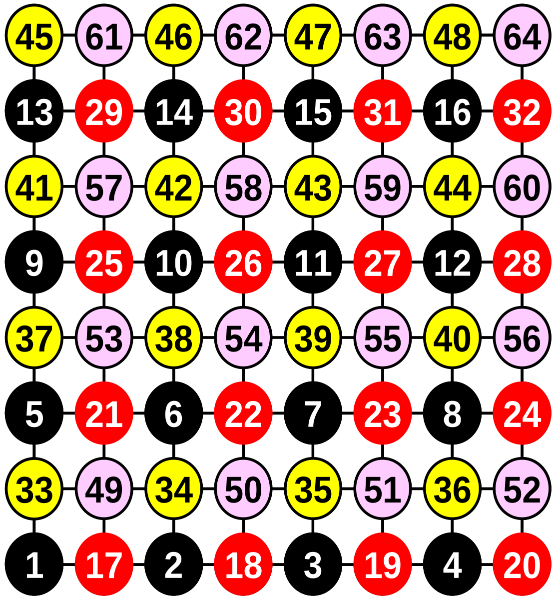
$$z_{\Omega_1}^{n} = z_{\Omega_1}^{n-1} + M_{\Omega_1}^{-1}\left(r_{\Omega_1} - M_{\Omega_1} z_{\Omega_1}^{n-1} - M_{\Gamma_1} z_{\Gamma_1}^{n-1}\right)$$

$$z_{\Omega_2}^{n} = z_{\Omega_2}^{n-1} + M_{\Omega_2}^{-1}\left(r_{\Omega_2} - M_{\Omega_2} z_{\Omega_2}^{n-1} - M_{\Gamma_2} z_{\Gamma_2}^{n-1}\right)$$

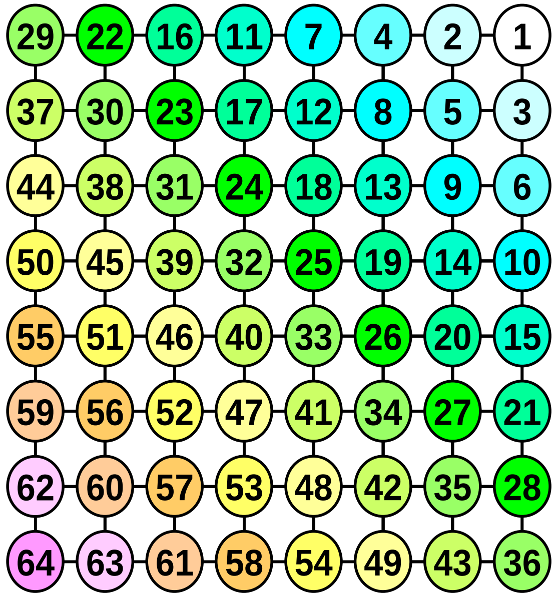


Reordering for avoiding data dependency in IC/ILU computations on each MPI process

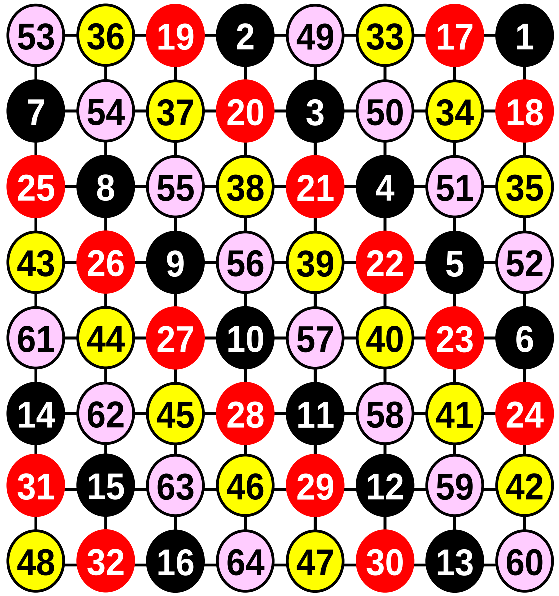
Elements in “same color” are independent: to be parallelized by OpenMP on each MPI process.



MC (Color#=4)
Multicoloring



RCM
Reverse Cuthill-Mckee



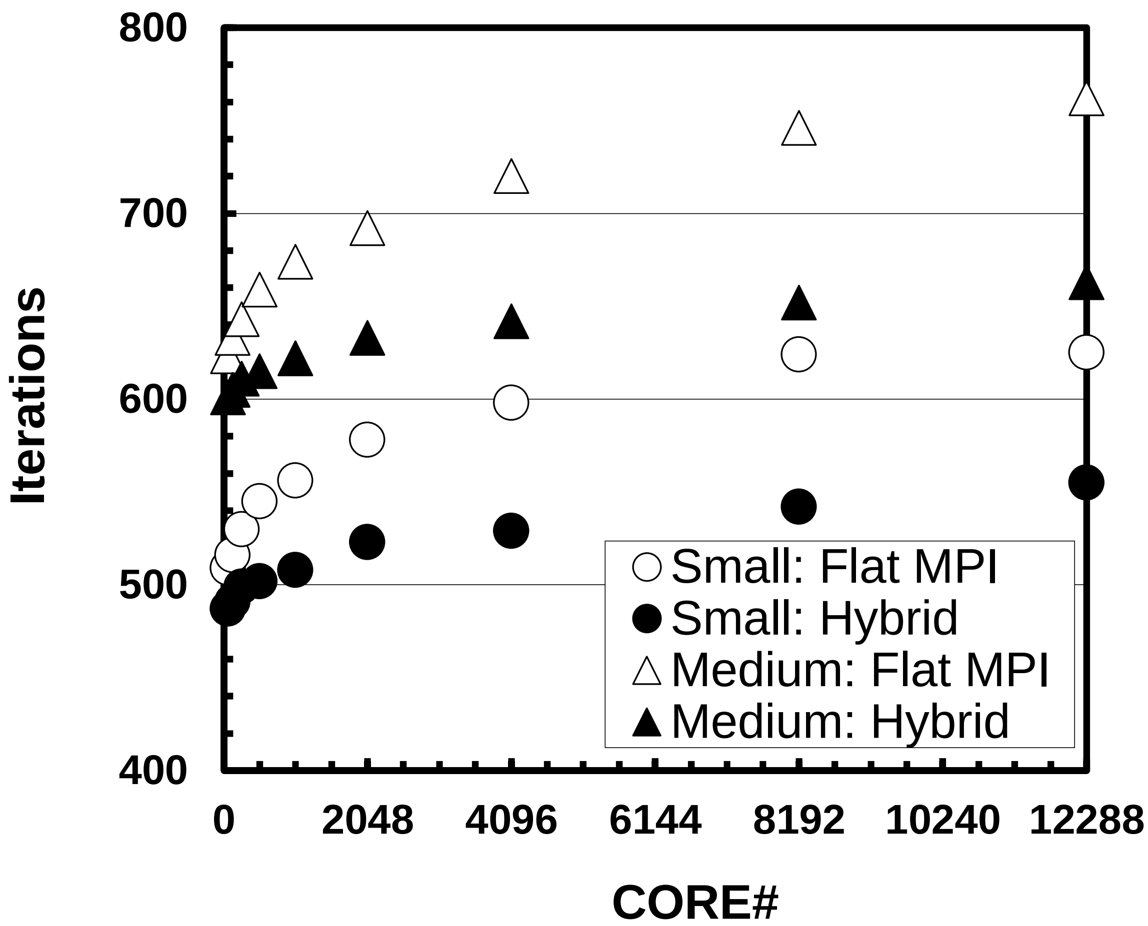
CM-RCM (Color#=4)
Cyclic MC + RCM

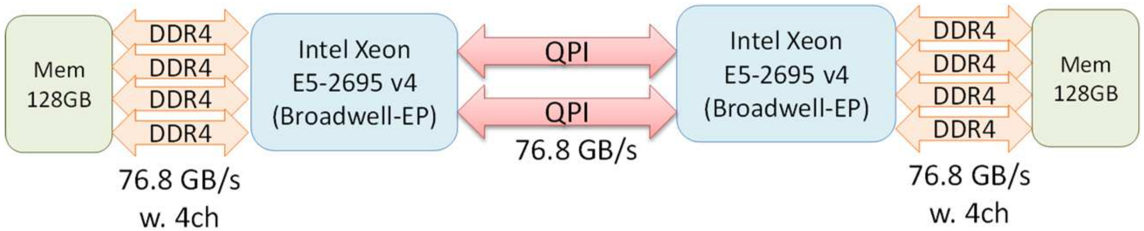
Results on Reedbush-U (RDB)

- 4 Types of Algorithms
  - Alg.1 Original Preconditioned CG
  - Alg.2 Chronopoulos/Gear
  - Alg.3 Pipelined CG (MPI_Allreduce, MPI_Iallreduce,)
  - Alg.4 Gropp's asynchronous CG (MPI_Allreduce, MPI_Iallreduce)
- Flat MPI, OpenMP/MPI Hybrid with Reordering
- Platform
  - Integrated Supercomputer System for Data Analyses & Scientific Simulations (Reedbush-U)
  - Intel Broadwell-EP 18 cores x 2 sockets x 420 nodes
  - Intel Fortran + Intel MPI
  - 16 of 18 cores/socket, up to 384 nodes (= 768 sockets, 12,288 cores)

Results: Number of Iterations

- Strong Scaling
- Small
  - 256×128×144 nodes (=4,718,592)
  - 14,155,776 DOF
  - at 384 nodes (12,288 cores)
    - 8×8×6=384 nodes/core
    - 1,152 DOF/core
- Medium
  - 256×128×288 nodes (=9,437,184)
  - 28,311,552自由度

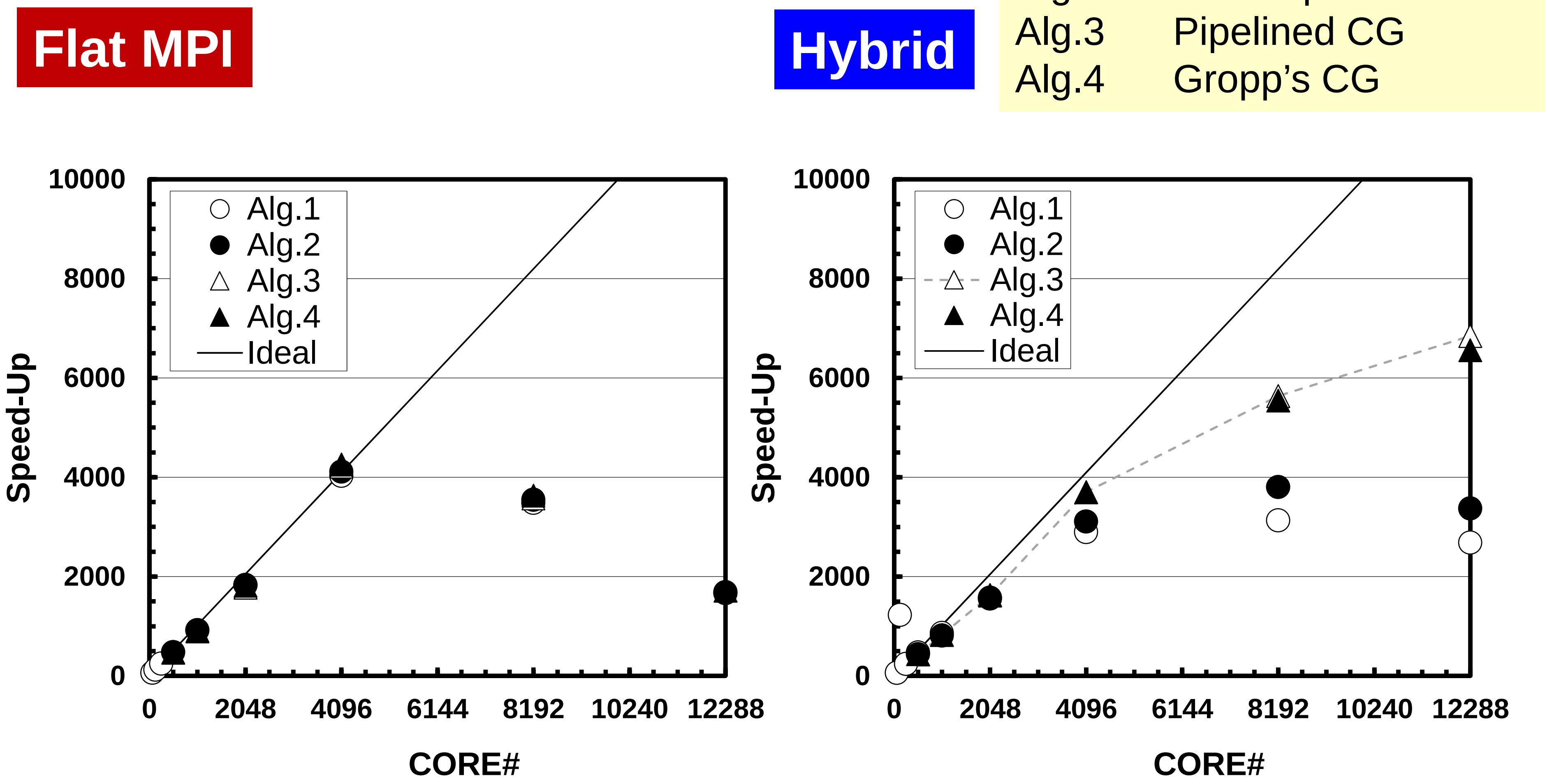




Results: Speed-Up: Small

Performance of 2 nodes of Flat MPI = 64.0 (4 sockets, 64 cores)

Alg.1	Original PCG
Alg.2	Chronopoulos/Gear
Alg.3	Pipelined CG
Alg.4	Gropp's CG

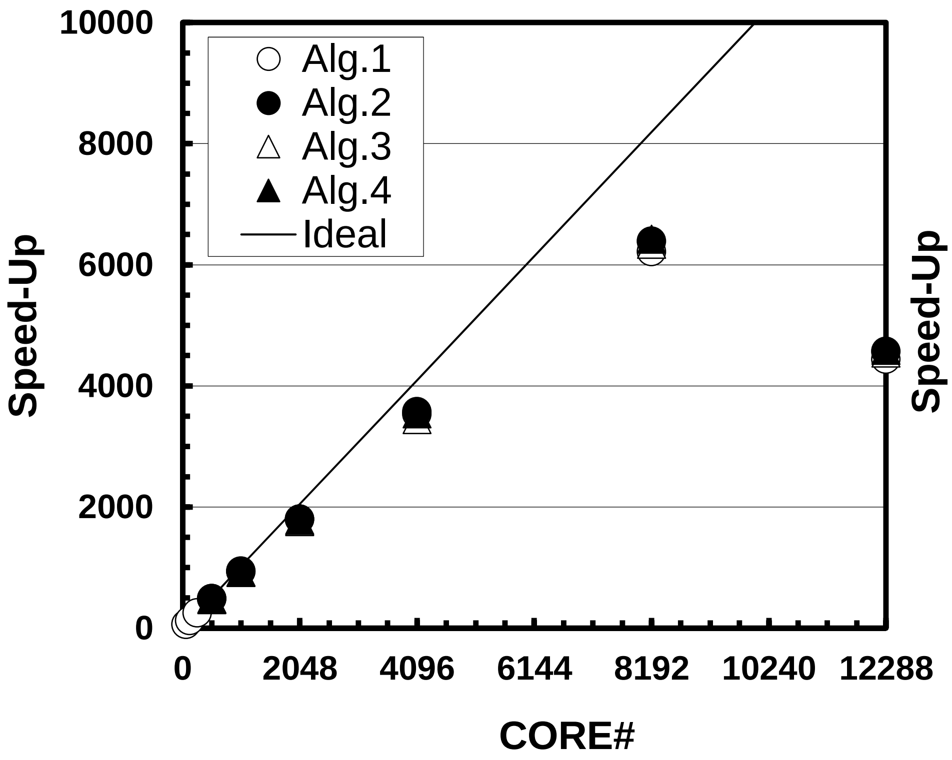


Results: Speed-Up: Medium

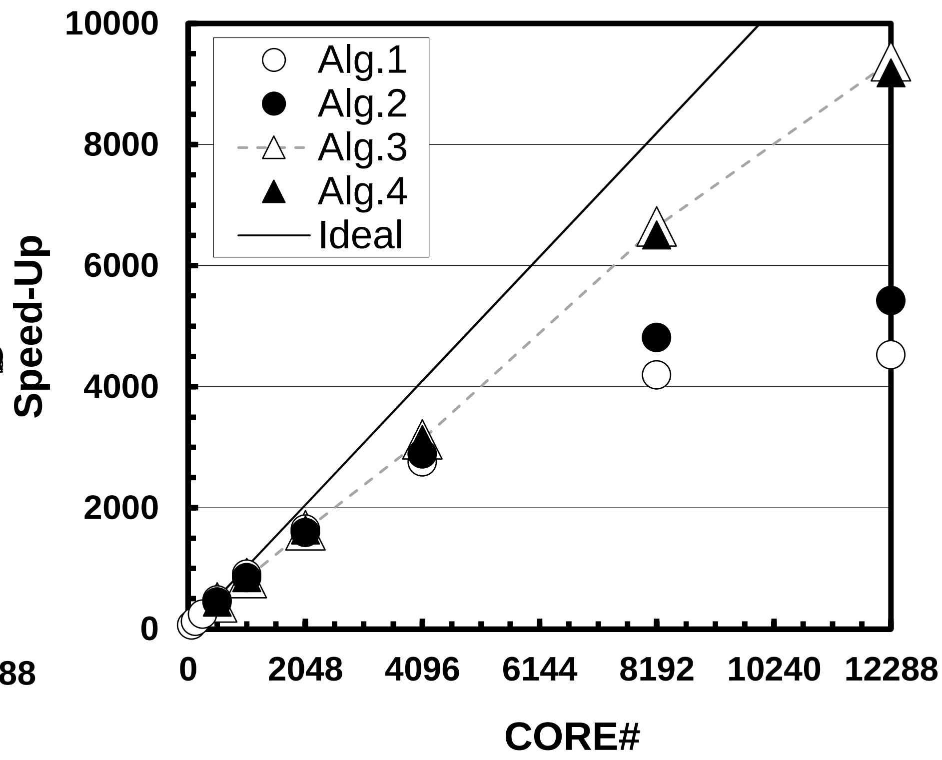
Performance of 2 nodes of Flat MPI = 64.0 (4 sockets, 64 cores)

Alg.1	Original PCG
Alg.2	Chronopoulos/Gear
Alg.3	Pipelined CG
Alg.4	Gropp's CG

Flat MPI



Hybrid

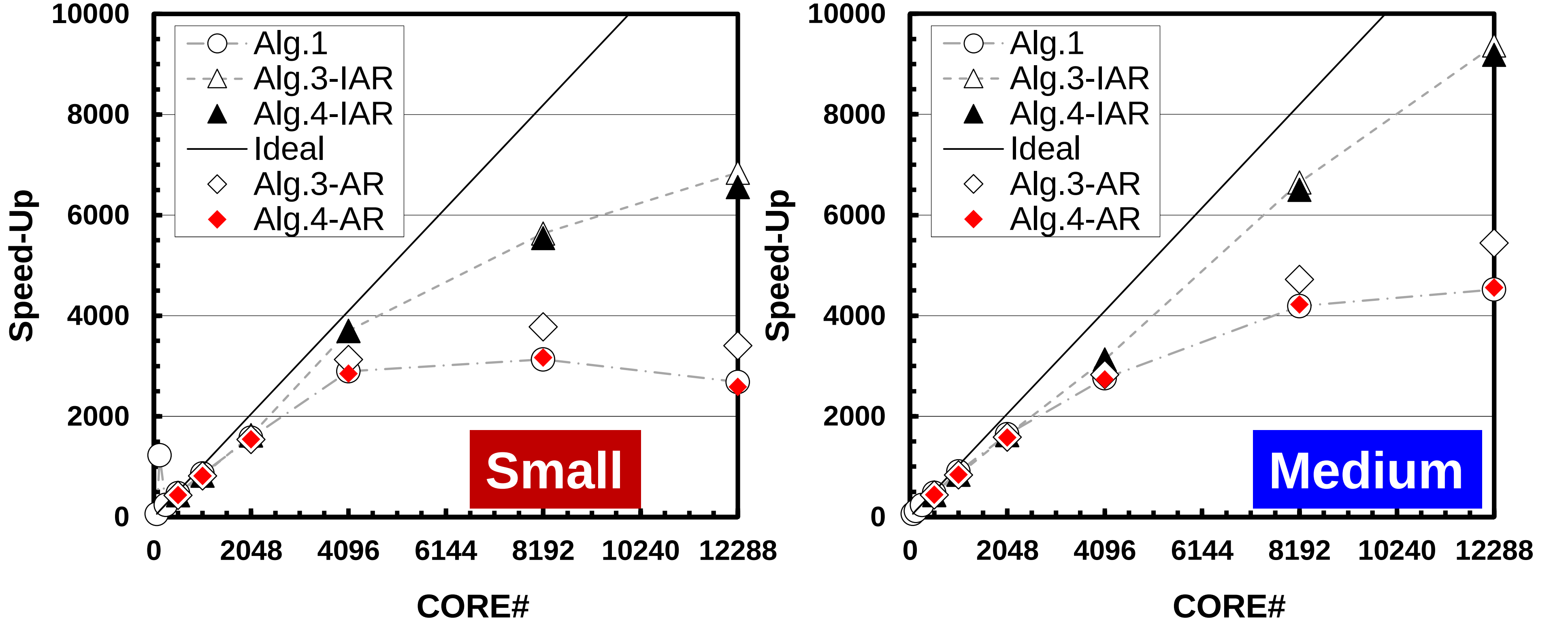


Allreduce vs. Iallreduce for Hybrid

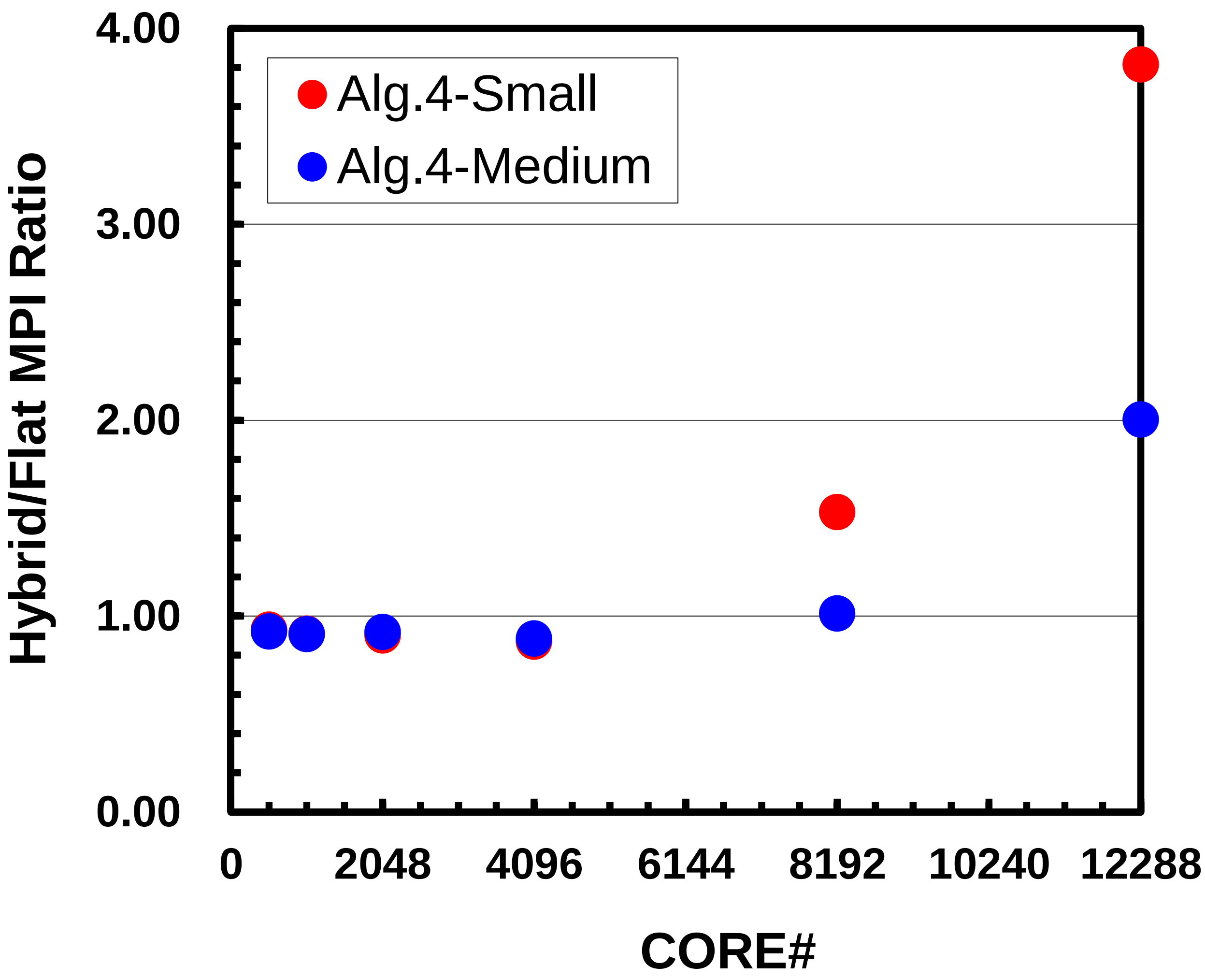
Performance of 2 nodes of Flat MPI = 64.0
(4 sockets, 64 cores)

IAR: MPI_Iallreduce
AR : MPI_Allreduce

Alg.1 Original PCG
Alg.3 Pipelined CG
Alg.4 Gropp's CG

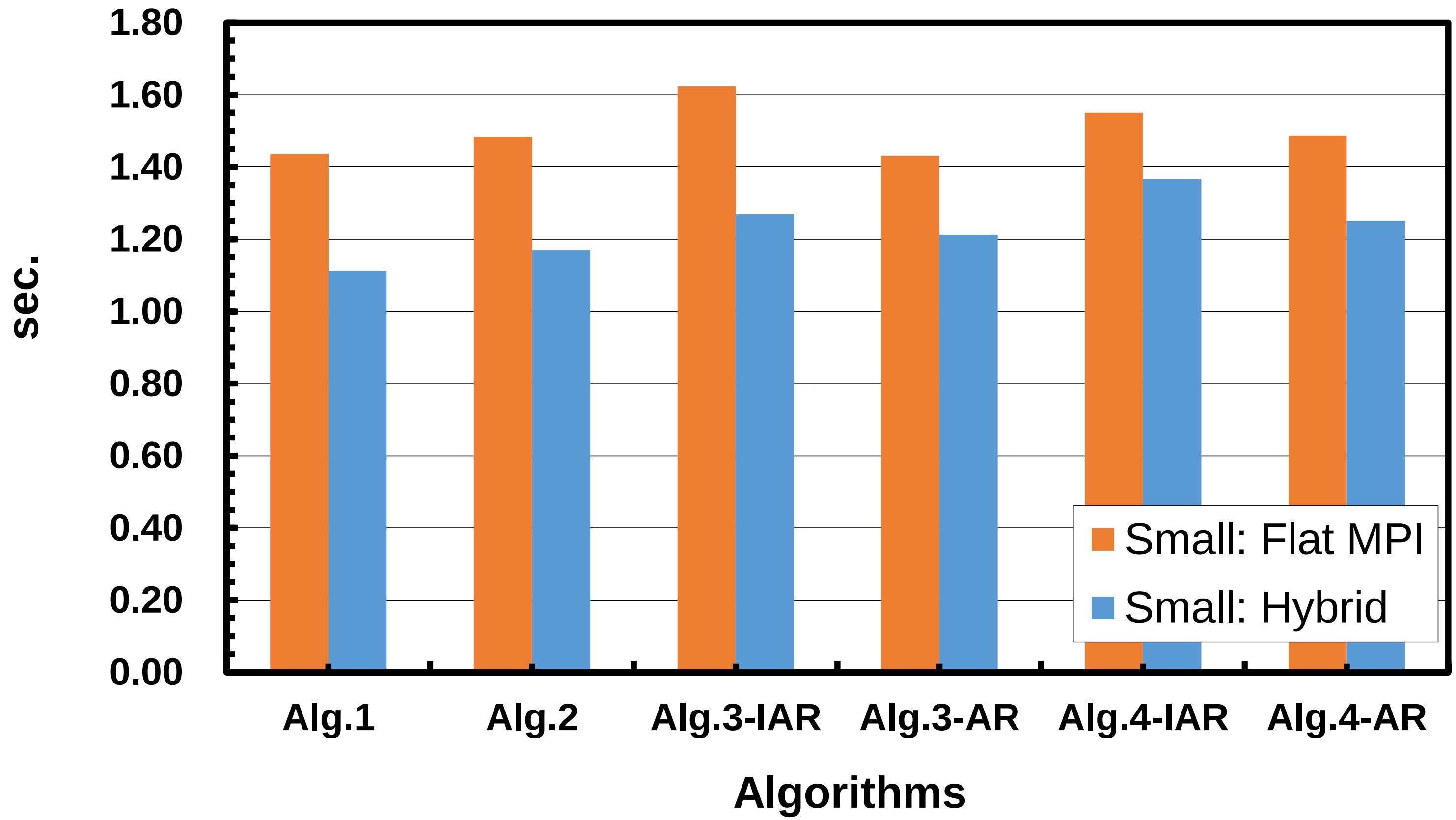


Hybrid vs. Flat MPI for Alg.4



Results on 768 nodes (12,288 cores) of Fujitsu FX10 (Oakleaf-FX)

MPI-3 is not optimized
FX100 has special HW for communication



Summary

- Pipelined CG, Gropp's CG
  - Effect of hiding collective communication by MPI_Iallreduce is significant, especially for strong scaling
  - Alg.3 ~ Alg.4
  - Future works
    - Pipelined CR should be also evaluated
      - Dr. Ghysels's recommendation
    - Application to Multigrid, HID (Hierarchical Interface Decomposition)
    - Evaluation on FX100, Oakforest-PACS (KNL Cluster) with OmniPATH
- Loop Scheduling for OpenMP
  - Effect is significant on FX10
  - Detailed profiling needed
  - Good target for AT (already done)