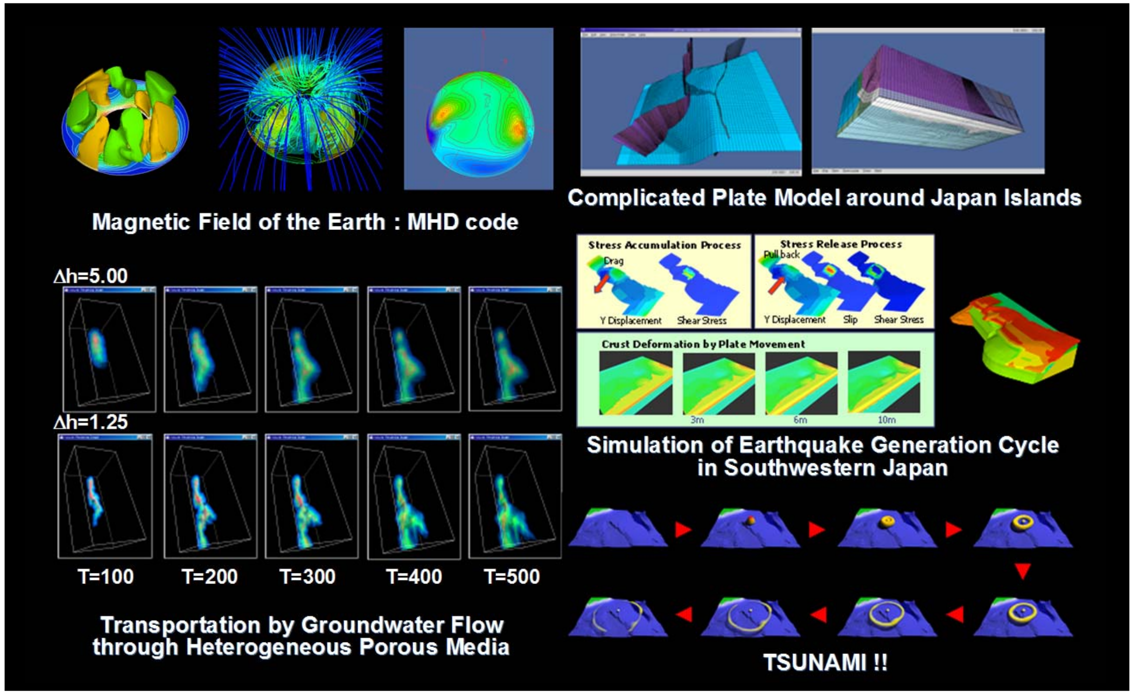
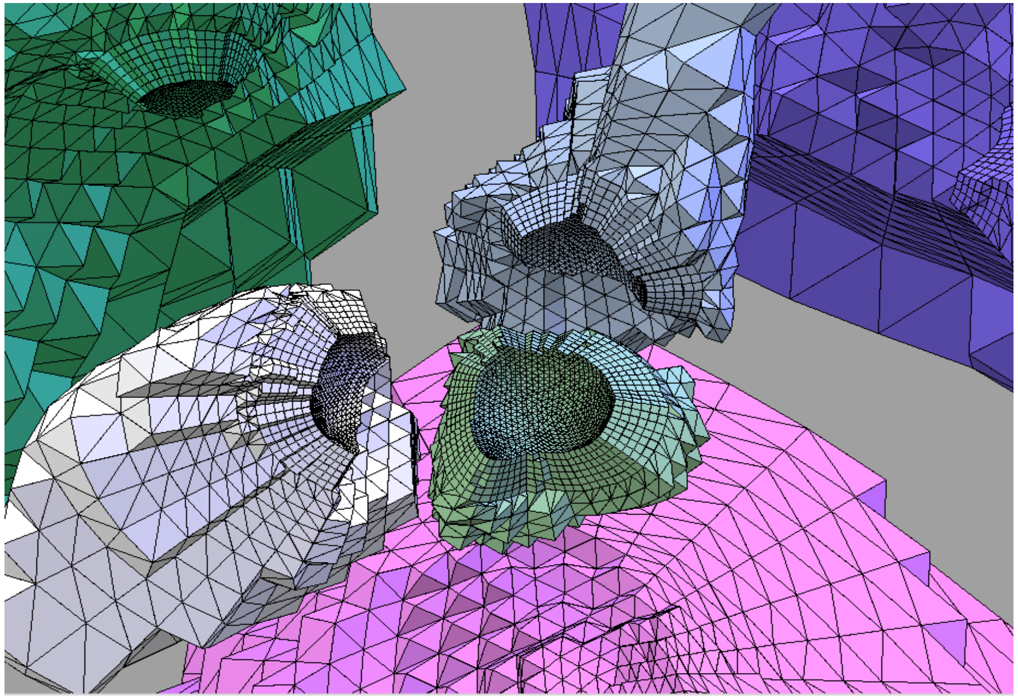
Parallel Preconditioning Method for Ill-Conditioned Problems

Kengo Nakajima
Information Technology Center
The University of Tokyo

Large-scale Simulations by Parallel FEM Procedures

- Unstructured grid with irregular data structure
- Large-scale sparse matrices
- Preconditioned parallel iterative solvers
- “Real-world” ill-conditioned problems



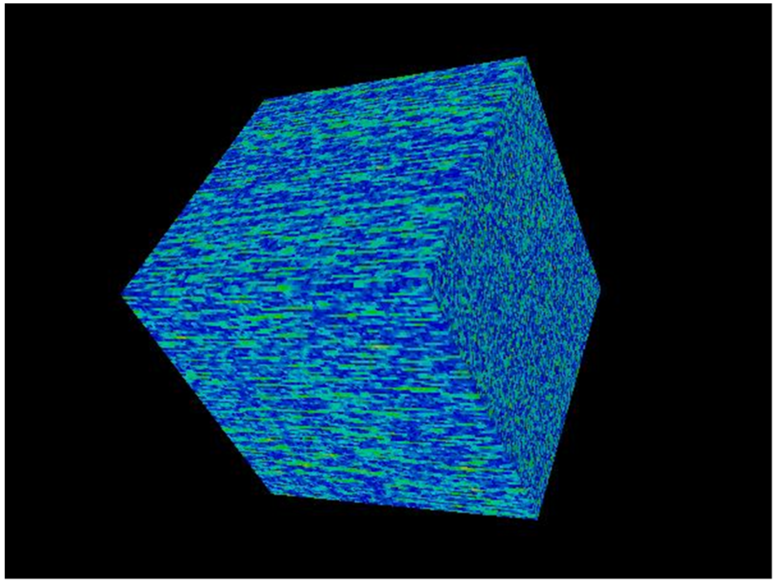


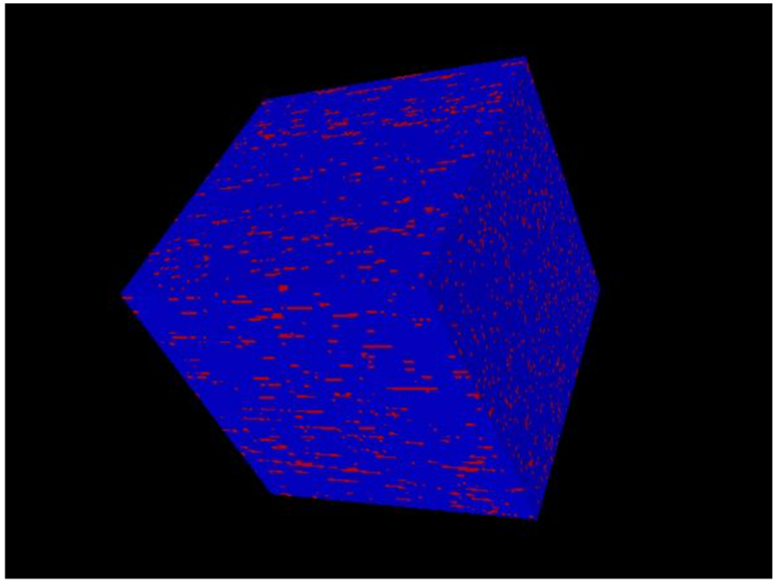
What are ill-conditioned problems ?

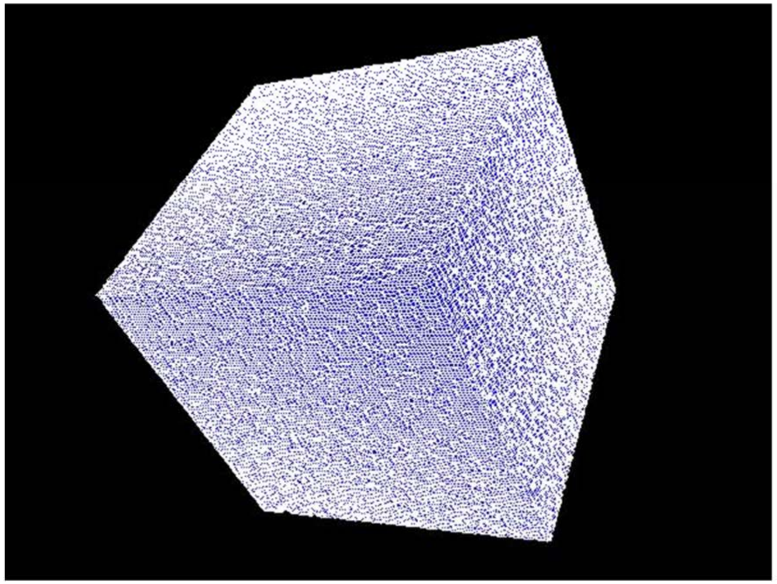
- Various ill-conditioned problems
  - For example, matrices derived from coupled NS equations are ill-conditioned even if meshes are uniform.
- In this work, we are focusing on 3D solid mechanics applications with:
  - heterogeneity
  - contact conditions
  - BILU/BIC
- Ideas can be extended to other fields.

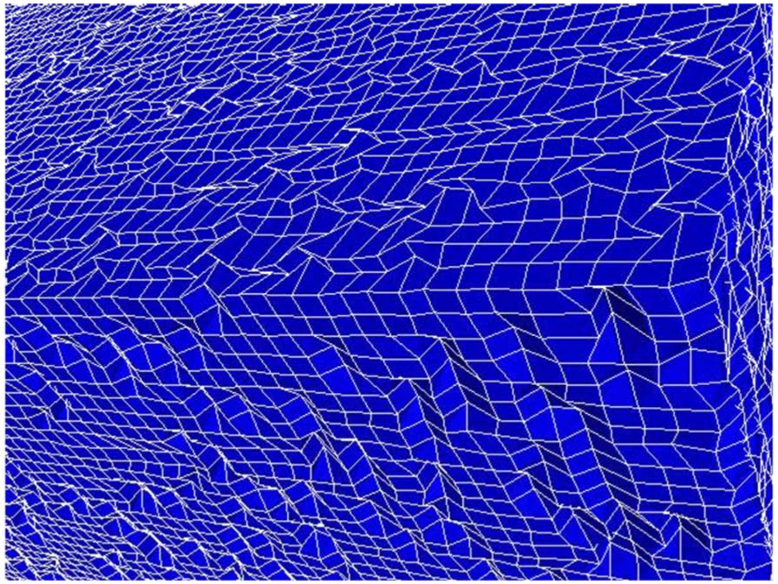
Ill-Conditioned Problems

Heterogeneous Fields, Distorted Meshes

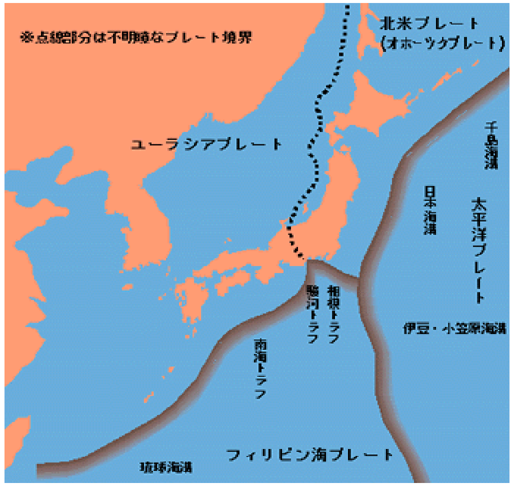


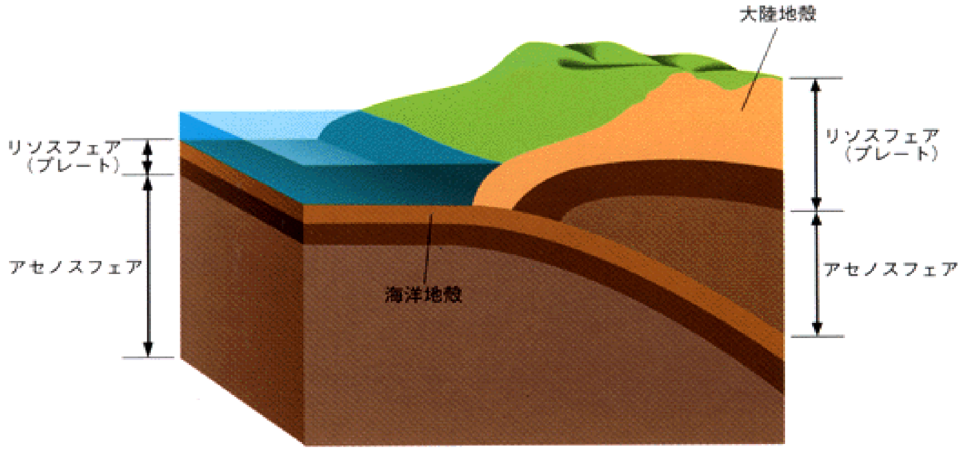


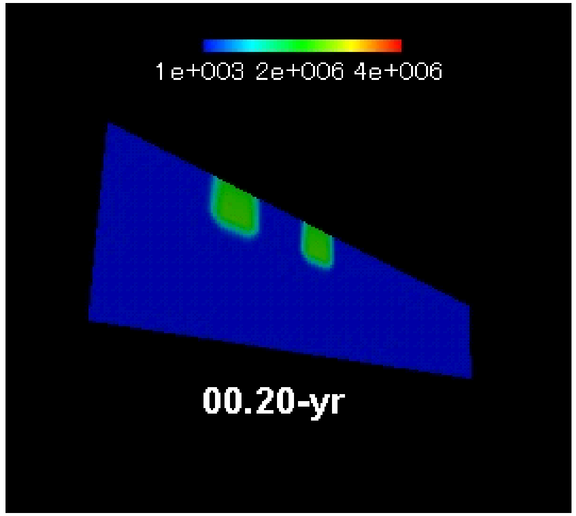




Contact Problems in Simulations of Earthquake Generation Cycle







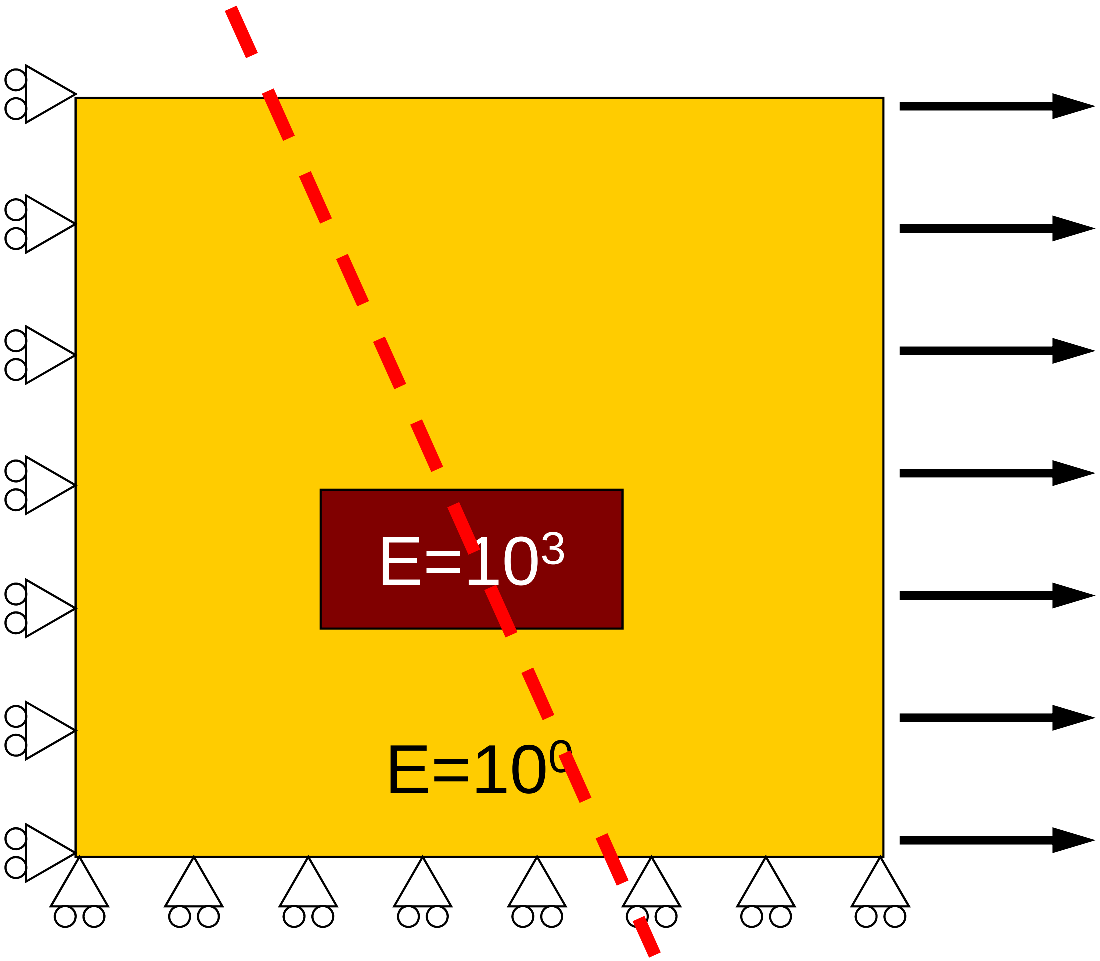
Preconditioning Methods (of Krylov Iterative Solvers) for Real-World Applications

- are the most critical issues in scientific computing
- are based on
  - Global Information: condition number, matrix properties etc.
  - Local Information: properties of elements (shape, size …)
- require knowledge of
  - background physics
  - applications

Technical Issues of “Parallel” Preconditioners in FEM

- Block Jacobi type Localized Preconditioners
- Simple problems can easily converge by simple preconditioners with excellent parallel efficiency.
- Difficult (ill-conditioned) problems cannot easily converge
  - Effect of domain decomposition on convergence is significant, especially for ill-conditioned problems.
    - Block Jacobi-type localized preconditioiners
    - More domains, more iterations
  - There are some remedies (e.g. deep fill-ins, deep overlapping), but they are not efficient.
  - ASDD does not work well for really ill-conditioned problems.

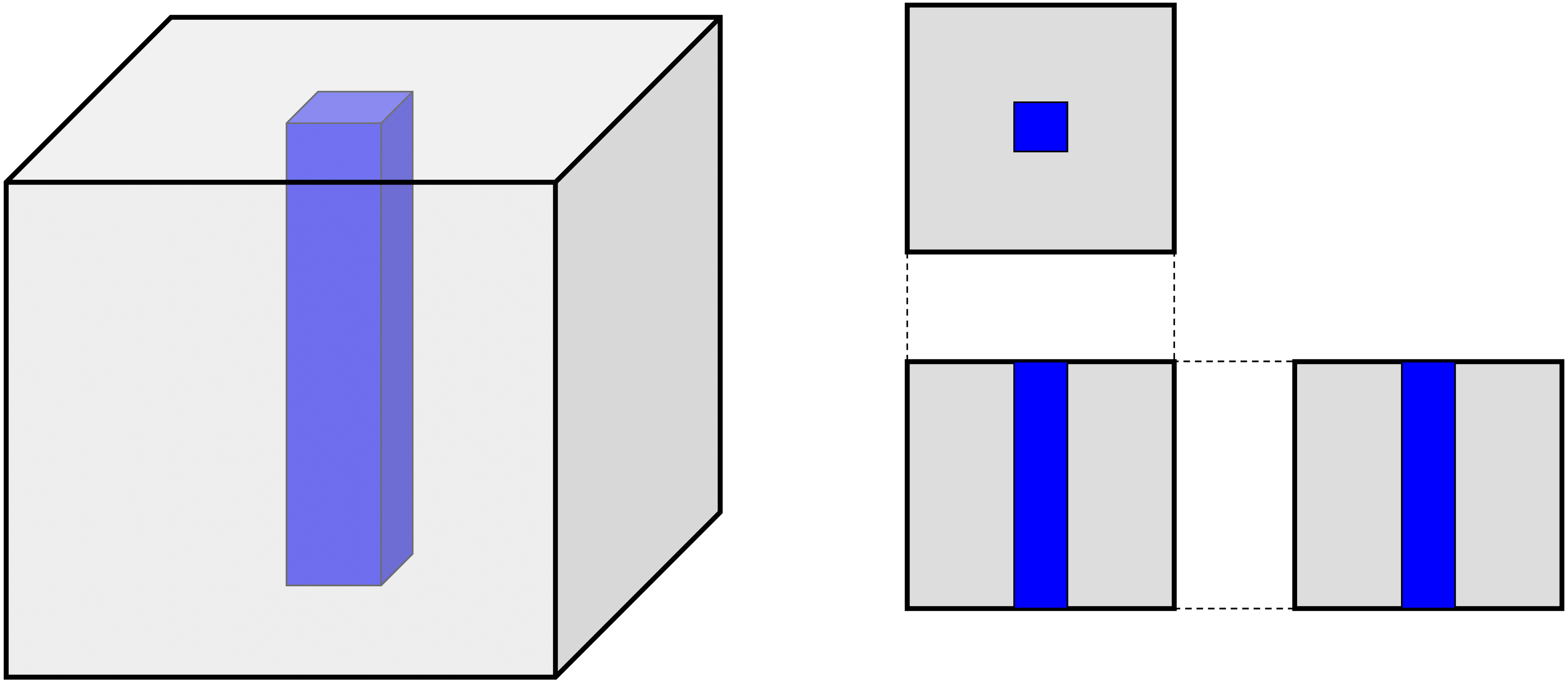
Technical Issues of "Parallel" Preconditioners for Iterative Solvers



3D Solid Mechanics
E: Young's Modulus

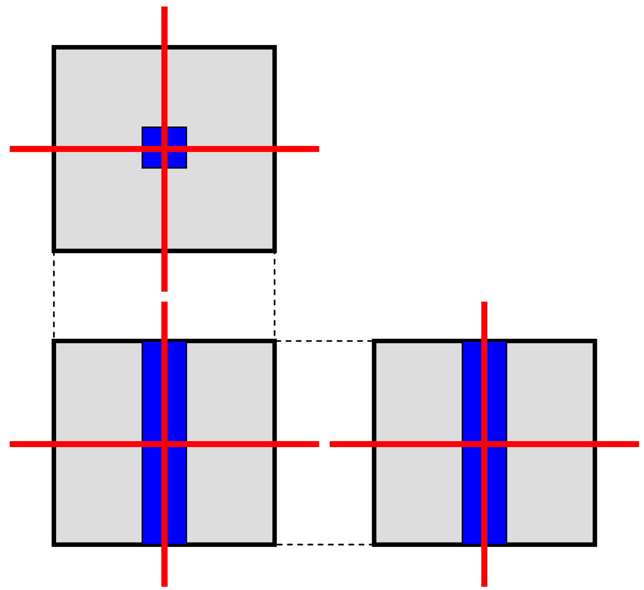
- If domain boundaries are on "stronger" elements, convergence is very bad.

3D Linear Elastic Problem with 20^3 Tri-Linear Hexahedral Elements

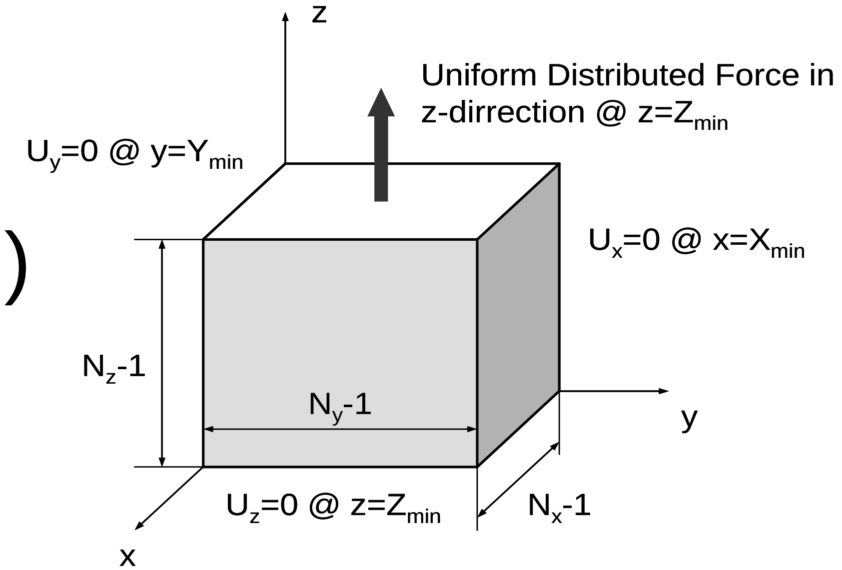


Number of Iterations for Convergence BILU(0)-GPBiCG with 8 domains

- : ν= 0.25
- : E=1.00



- 1-processor
  - : E=10^0 , 31 iterations
  - : E=10^{+3} , 84 iterations
    - Harder, More ill-conditioned

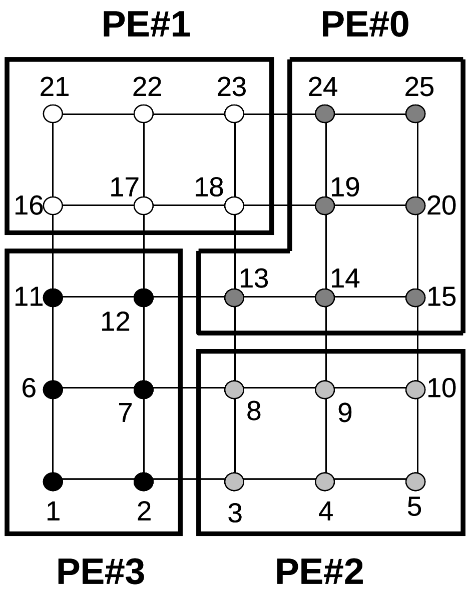


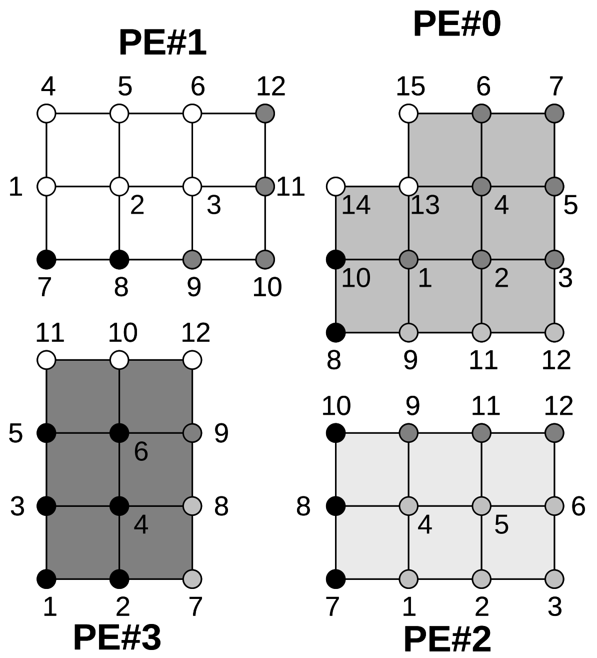
- 8-processors (MPI, no-overlapping)
  - : E=10^0 , 52 iter's (×1.68)
  - : E=10^{+3}, 158 iter's (×1.88)

Remedies: Domain Decomposition

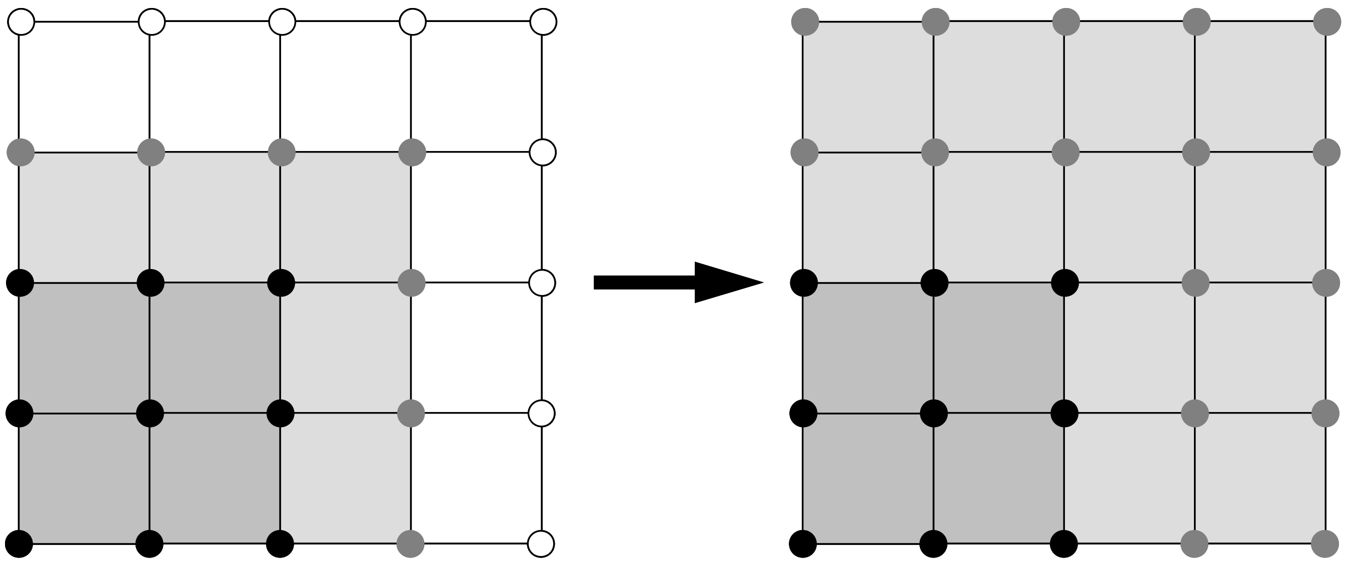
- Avoid “Strong Elements”
  - not practical
- Extended Depth of Overlapped Elements
  - Selective Fill-ins, Selective Overlapping [KN 2007]
    - adaptive preconditioning/domain decomposition methods which utilize features of FEM procedures
- PHIDAL/HID (Hierarchical Interface Decomposition) [Henon & Saad 2007]
- Extended HID [KN 2009]

Extension of Depth of Overlapping





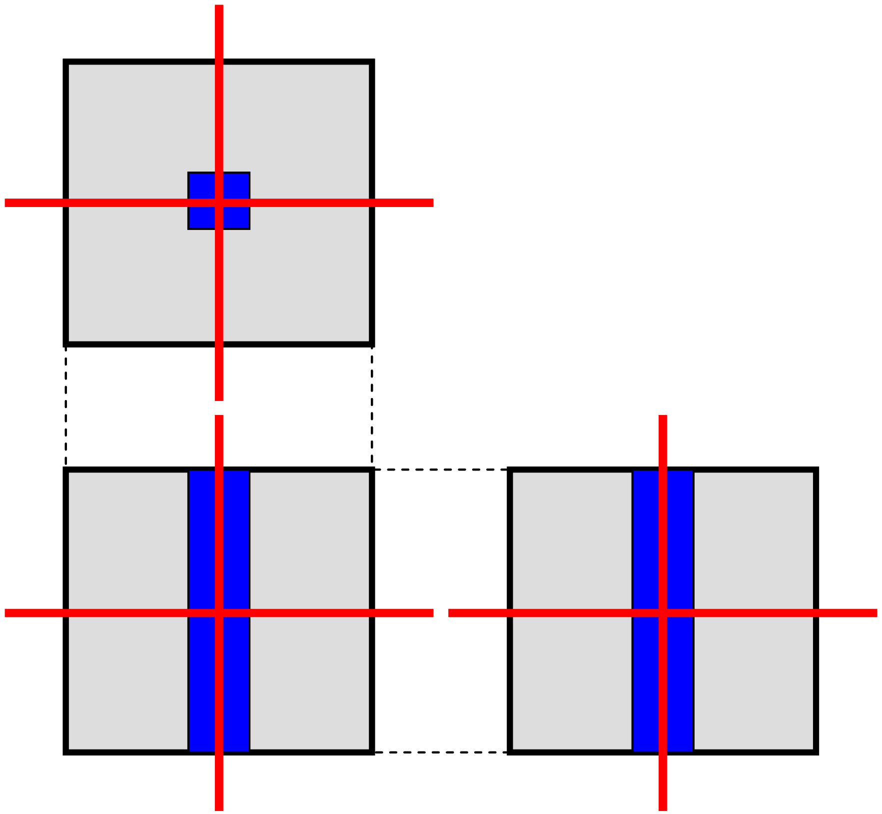
Cost for computation and communication may increase



●: Internal Nodes, ●: External Nodes
■: Overlapped Elements

Number of Iterations for Convergence BILU(0)-GPBiCG, 8-domains (PE's)

Effect of Extended Depth of Overlapping



Depth of Overlap	$E=10^0$	$E=10^3$
0	52	158
1	33	103
2	32	100
3	32	97
4	31	82
Single Domain	31	84

Final goal of this work

- Development of robust and efficient parallel preconditioning method

- Construction of strategies for optimum selection of preconditioners, partitioning, and related methods/parameters, such as:
  - Selective Fill-ins
  - Selective Overlapping/HID

How to get to the final goal ?

- Utilization of both of:
  - global information obtained from derived coefficient matrices
  - very local information, such as information of each mesh in finite-element applications.

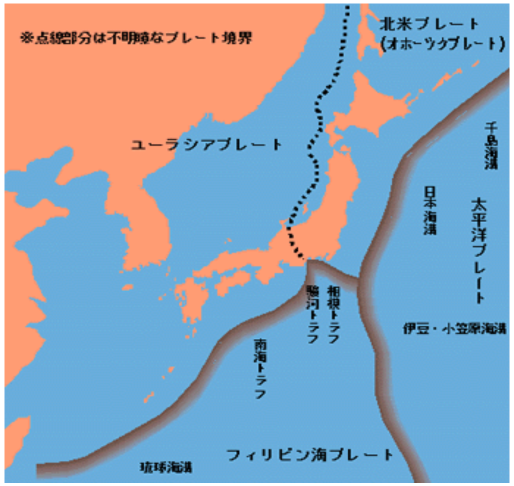
- Usually, this type of work mainly focuses on features of derived coefficient matrices (e.g. ILUT)
  - In real applications, convergence of parallel iterative solvers is often affected by local heterogeneity and/or discontinuity of the field, as shown in this presentation.

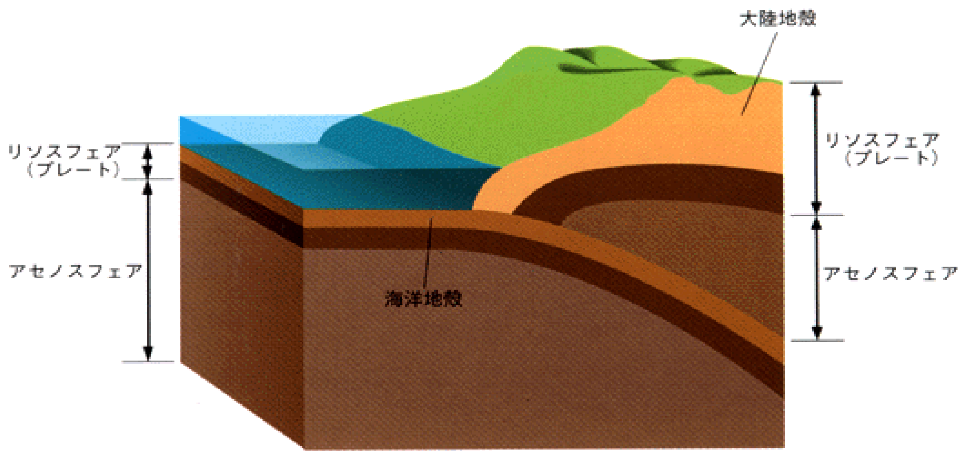
Overview

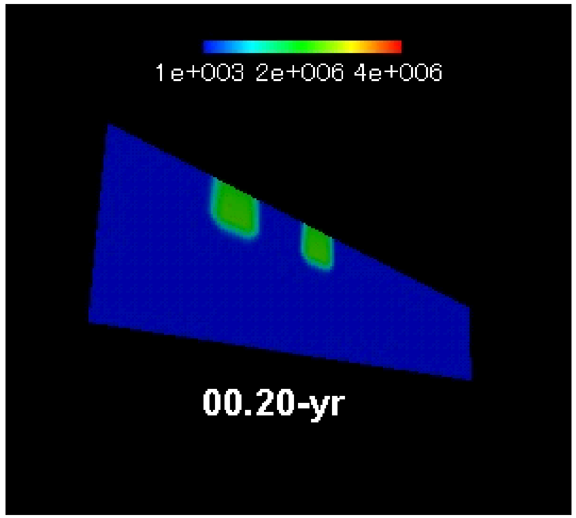
- Background
  - Selective Blocking
  - More General Problems
    - Extension of Overlapped Zones
- Preconditioning/Partitioning Methods
  - Target Application
  - Selective Fill-ins, Selective Overlapping
- HID
  - Hierarchical Interface Decomposition
- Extended HID
- Fields with Heterogeneity

Initial Motivation: Contact Problems in Simulations of Earthquake Generation Cycle

- Quasi-static stress accum. process at plate boundaries
- Non-linear contact problems with Newton-Raphson iter's
- Ill-conditioned linear equations due to penalty constraint by ALM (Augmented Lagrangean).
- Parallel FEM with domain decomposition (GeoFEM)

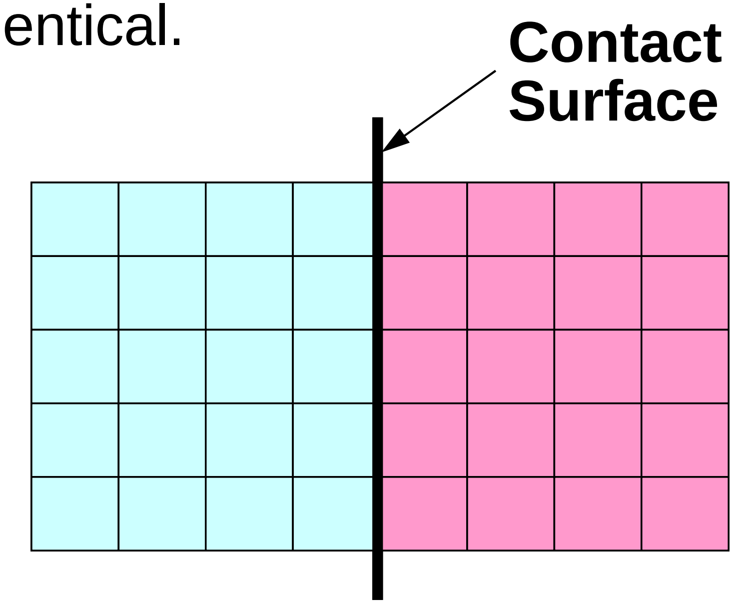






Contact Problems in Simulations of Earthquake Generation Cycle

- Assumptions (GeoFEM: http://geofem.tokyo.rist.or.jp/)
  - Infinitesimal deformation, static contact relationship.
    - Location of nodes is in each "contact pair" is identical.
    - “Consistent” node number and position
- Large-scale problems
  - Parallel preconditioned iterative solvers
- Special preconditioning : ***Selective Blocking.***
  - provides robust and smooth convergence in 3D solid mechanics simulations for geophysics with contact.
- Special partitioning

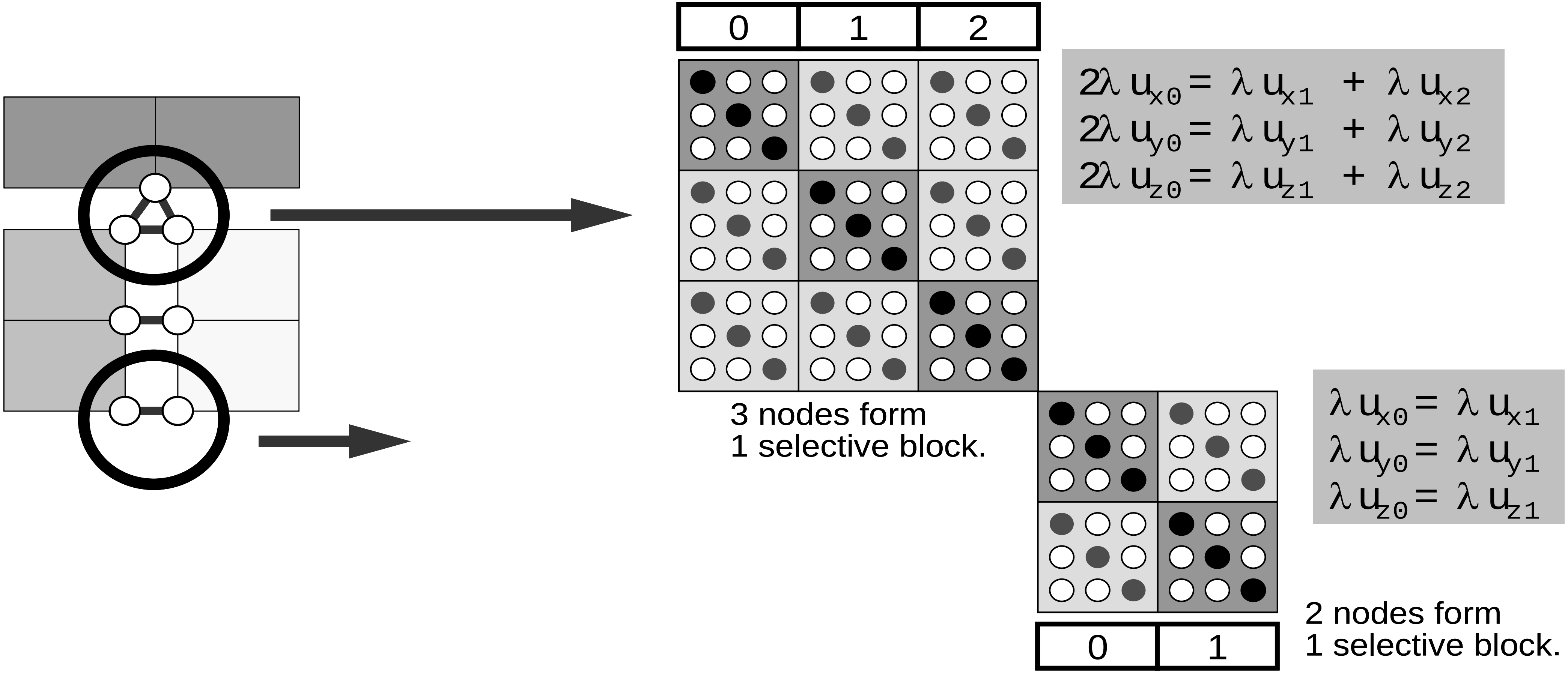


Selective Blocking [KN 2001]

Special Method for Contact Problem

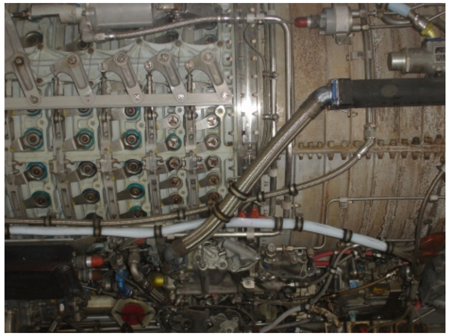
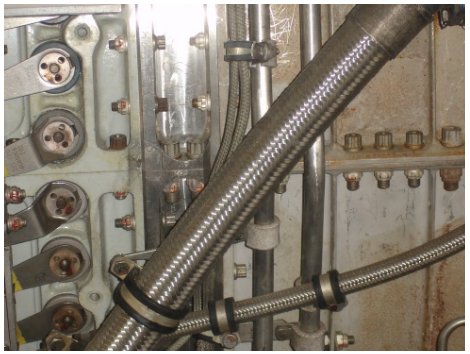
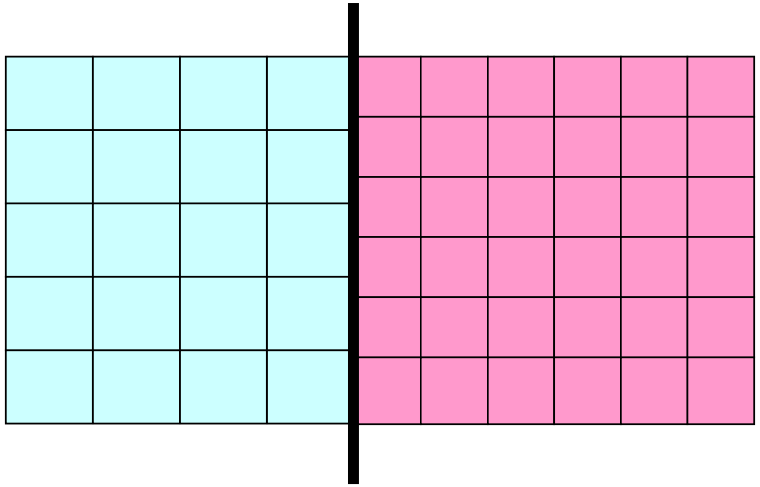
Strongly coupled nodes are put into the same diagonal block.

Full LU factorization for each block.

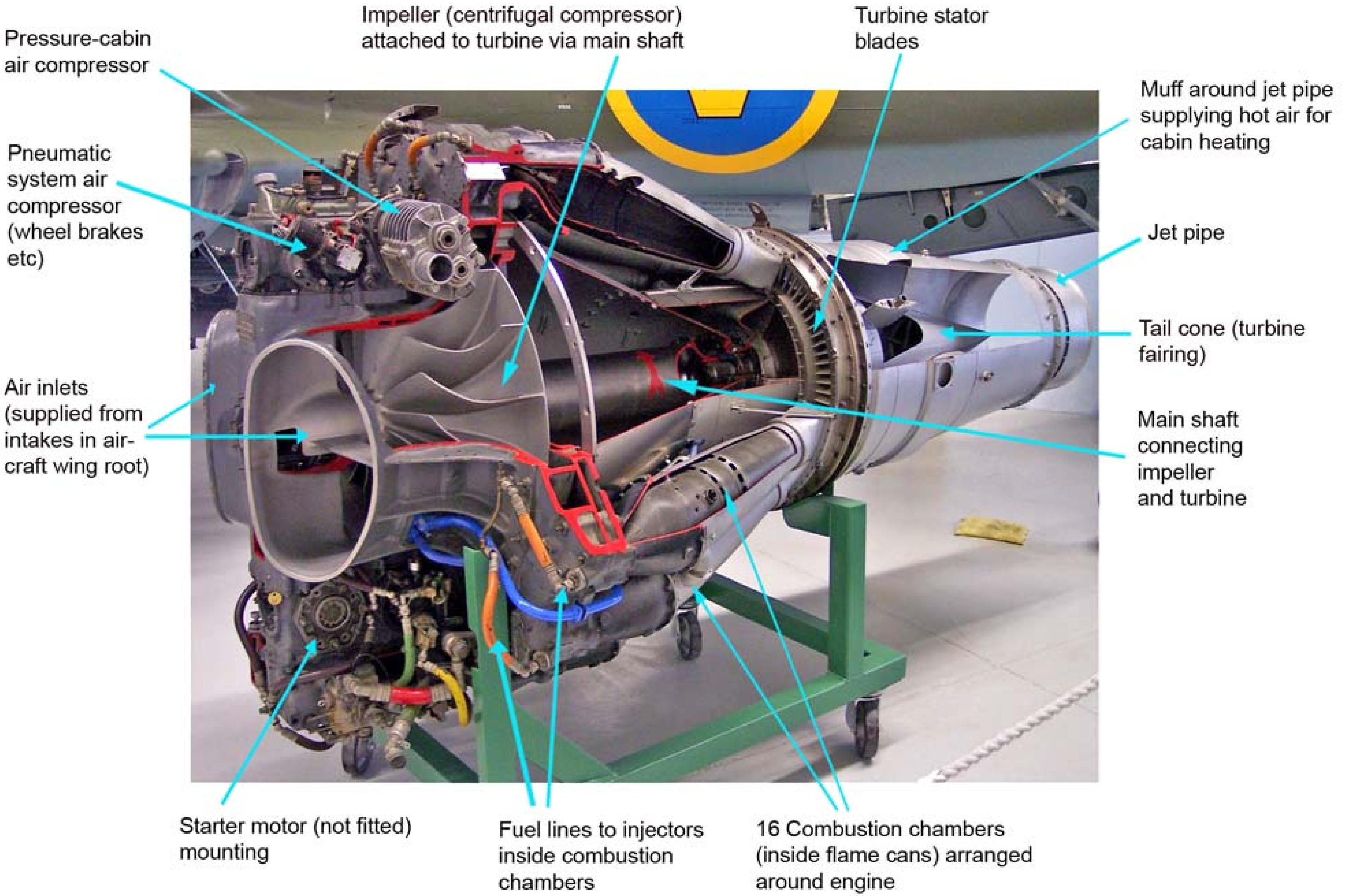


More General Problems

- Moving boundaries due to large slip conditions
- Inconsistent node number (and location) at boundary surfaces
  - Assembly structure for machine parts.
    - where meshes for each part are separately generated.
  - Commercial FEM codes (e.g. ABAQUS, NASTRAN) can treat problems for this type of "inconsistent" cases. (single PE, direct method for linear equations).

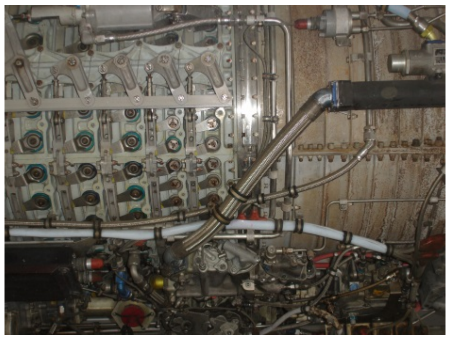
Example of Assembly Structure
Jet Engine

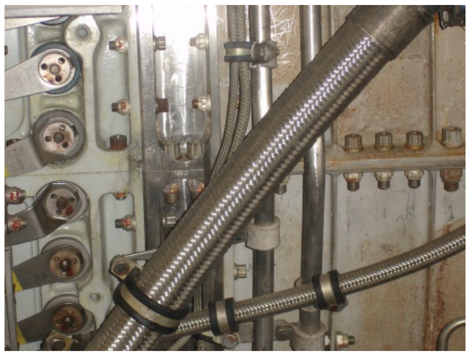


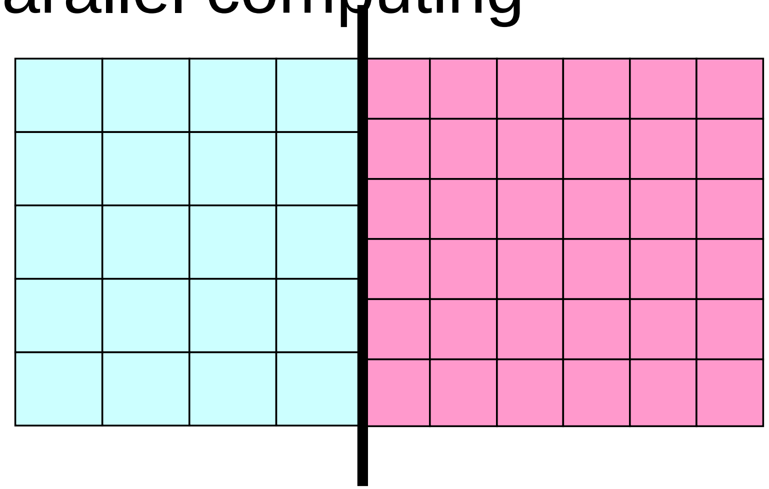
More General Problems

Inconsistent Number of Nodes at Boundary Surfaces

- Difficult to apply “selective blocking”
  - Size of each “selective block” may be too large for full LU factorization
- Difficult to apply “special partitioning”
- Remedy
  - Higher-order fill-in’s
  - Extension of overlapped zones for parallel computing

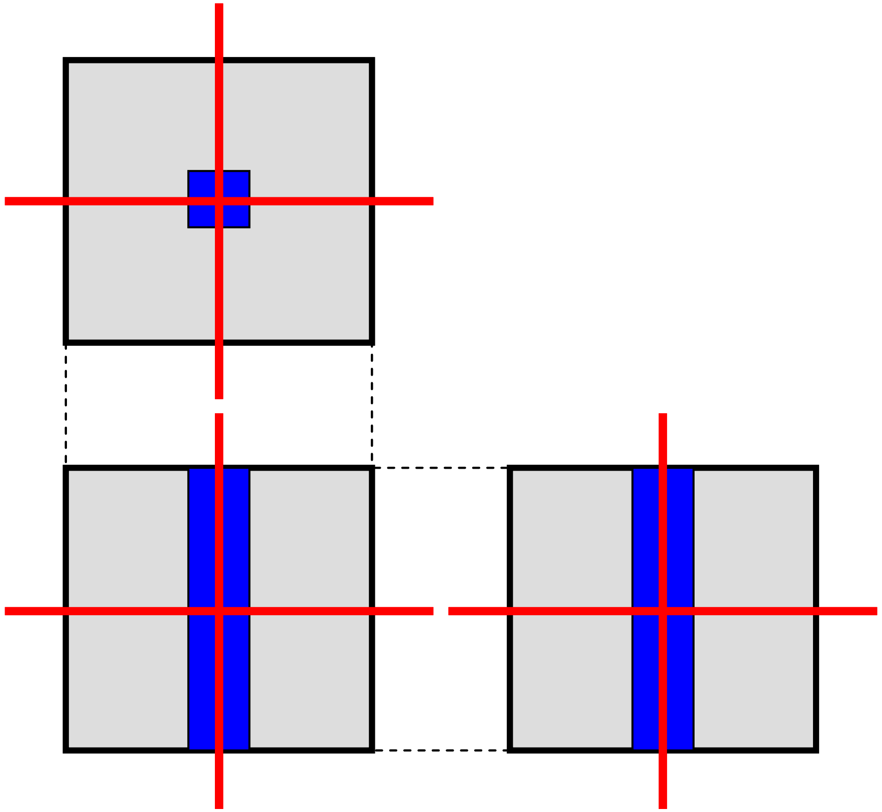






Number of Iterations for Convergence BILU(0)-GPBiCG, 8-domains (PE’s)

Effect of Extended Depth of Overlapping



Depth of Overlap	$E=10^0$	$E=10^3$
0	52	158
1	33	103
2	32	100
3	32	97
4	31	82
Single Domain	31	84

- Background
  - Selective Blocking
  - More General Problems
    - Extension of Overlapped Zones
- **Preconditioning/Partitioning Methods**
  - **Target Application**
  - **Selective Fill-ins, Selective Overlapping**
- HID
  - Hierarchical Interface Decomposition
- Extended HID
- Fields with Heterogeneity

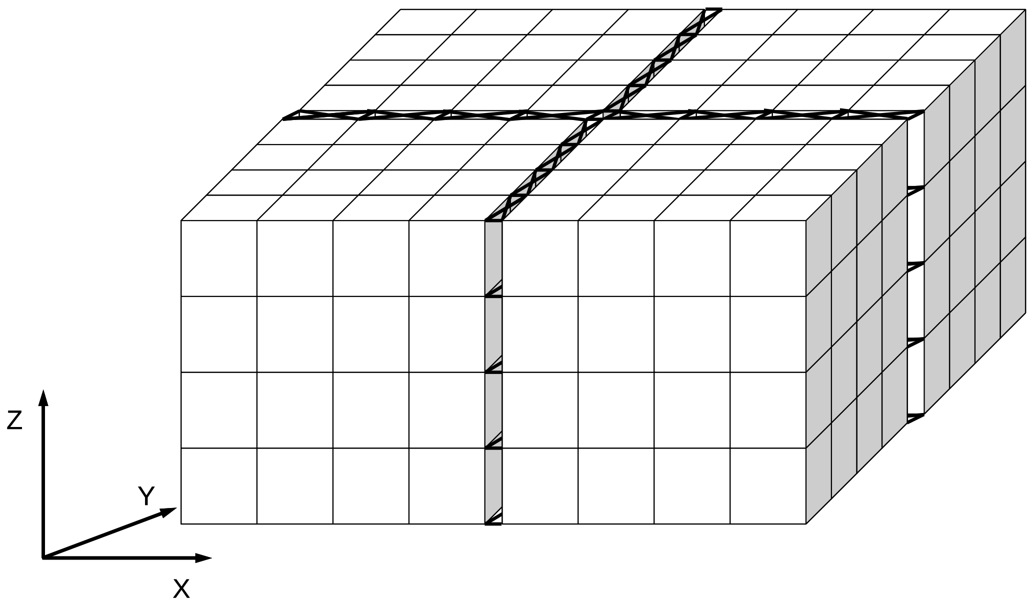
Robust and efficient preconditioning for parallel iterative solvers in more general cases

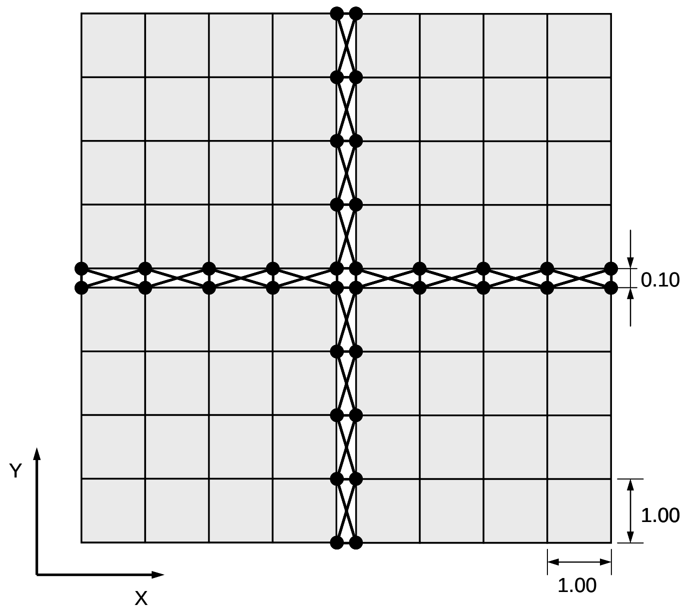
- Selective fill-ins for serial & parallel computing
- Selective overlapping for parallel computing

- Features of individual element are utilized.

Example for "Inconsistent" Cases

This model simulates contact problem in assembly structure

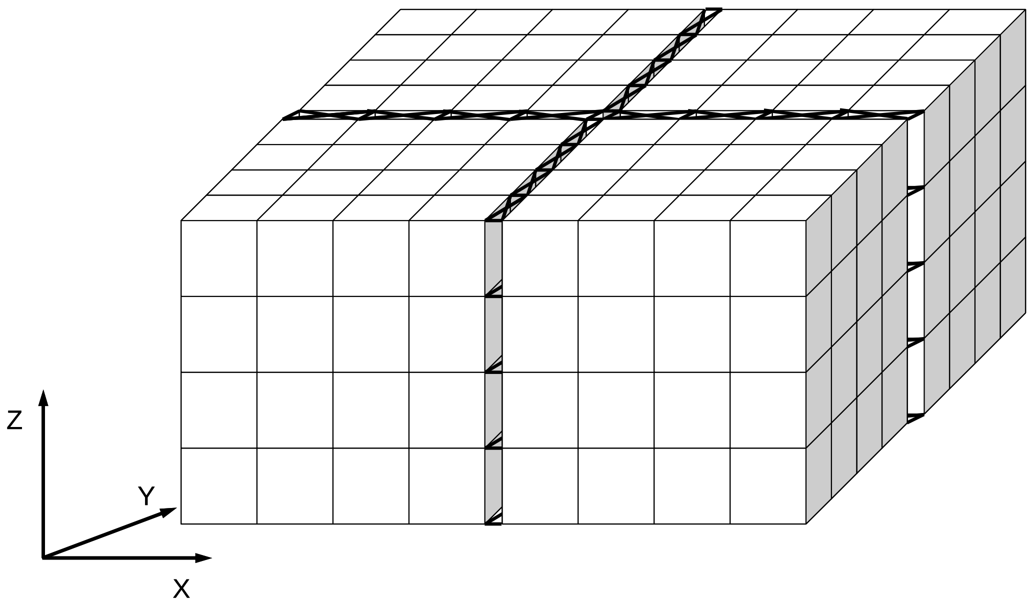


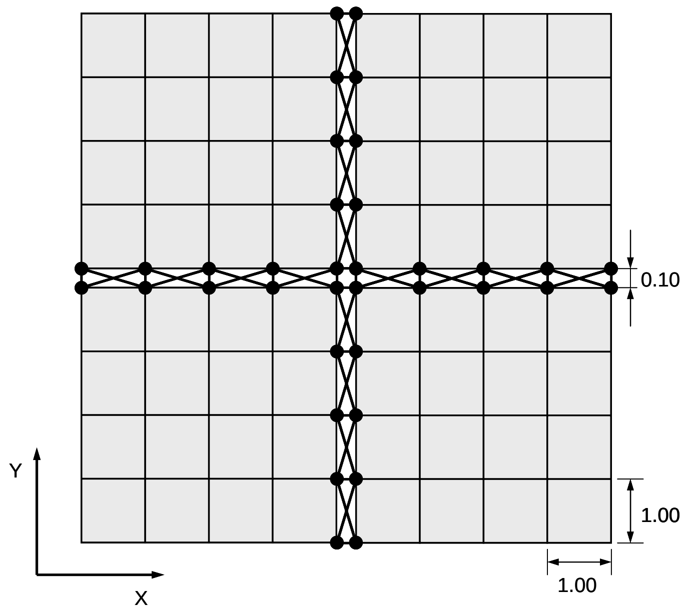


- Each block is discretized into cubic tri-linear elements
  - elastic material: E= 1.00, Poisson ration= 0.25
- Each block is connected through elastic truss elements generated on each node on contact surfaces.
  - Truss elements are crossing.

Example for "Inconsistent" Cases

This model simulates contact problem in assembly structure





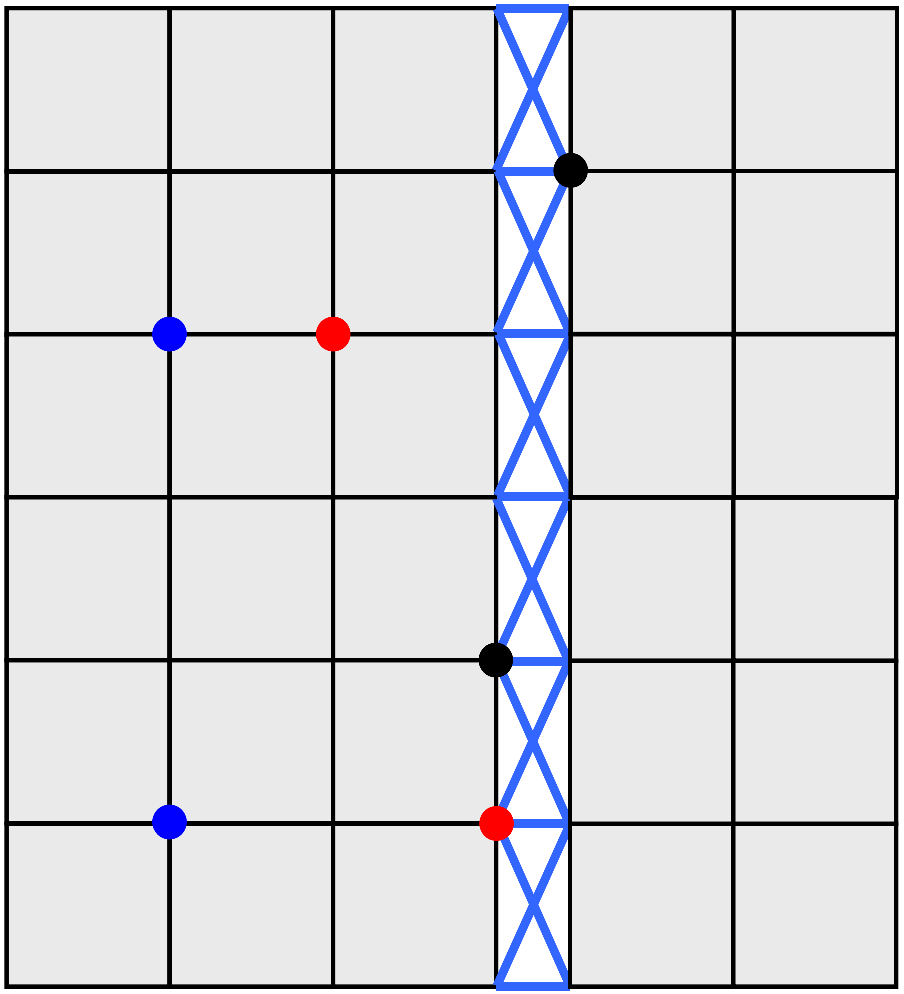
- Elastic coefficient of truss elements is set to 10^3 times as large as that of solid elements.
  - This condition simulates constraint boundary conditions for contact.
- Distributed uniform force at $z=z_{max}$ surface
  - u=0@x=0, v=0@y=0, w=0@z=0

Selective Fill-ins [KN 2007]

- Apply higher order of fill-ins between nodes which connect to truss-type elements.
  - Similar concept as “selective blocking”

- In this work: **BILU(1+)**
  - BILU(2) for these special nodes (2nd order fill-ins)
  - BILU(1) for general nodes (1st order fill-ins)
- Cost is similar to that of BILU(1), but effect of preconditioning is expected to be competitive with that of BILU(2).

Idea of “Selective Fill-ins”: ILU(1+)



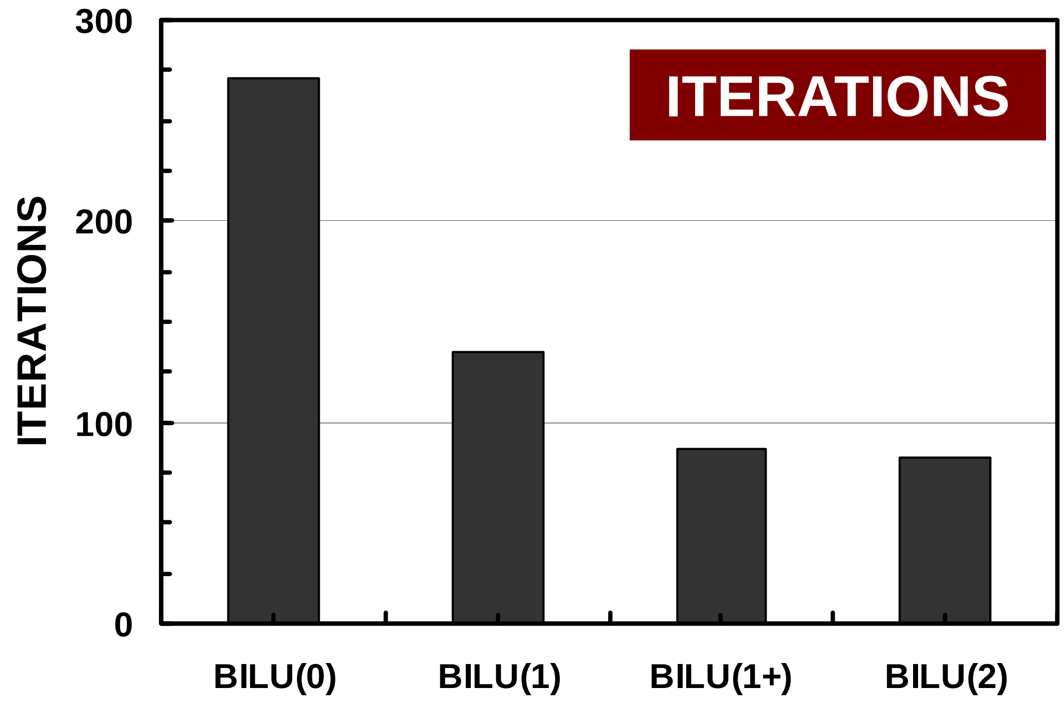
- 2nd order fill-in’s are considered for these nodes
- 2nd order fill-in’s are NOT considered for these nodes
- 2nd order fill-in’s are NOT considered for these nodes

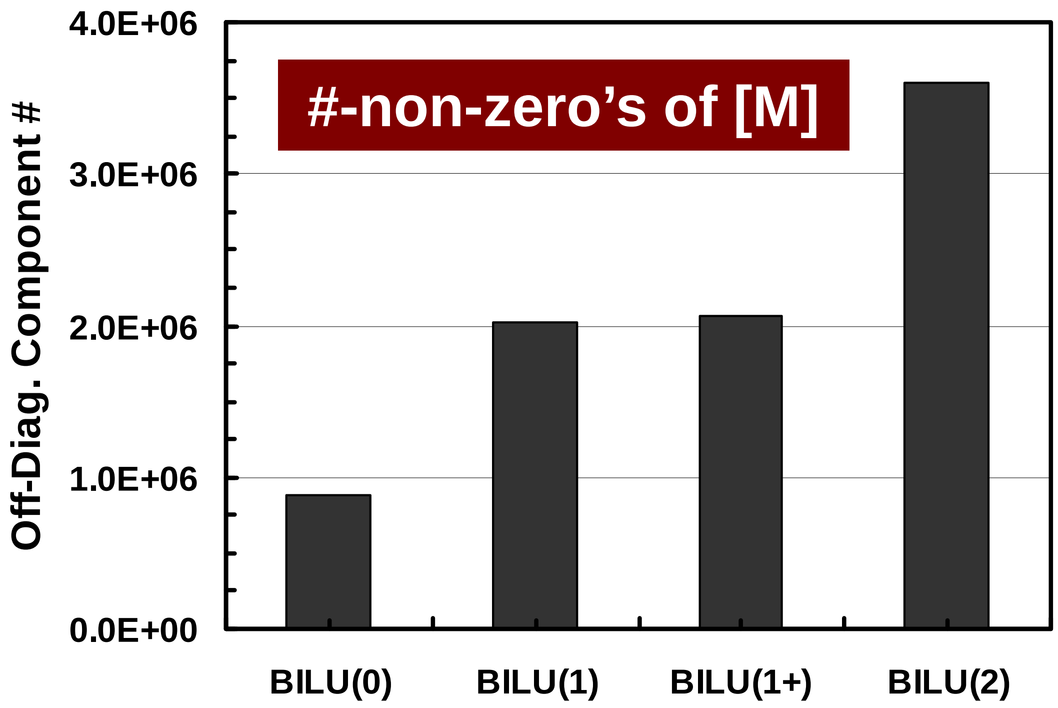
Summary of Problem Setting Single Core

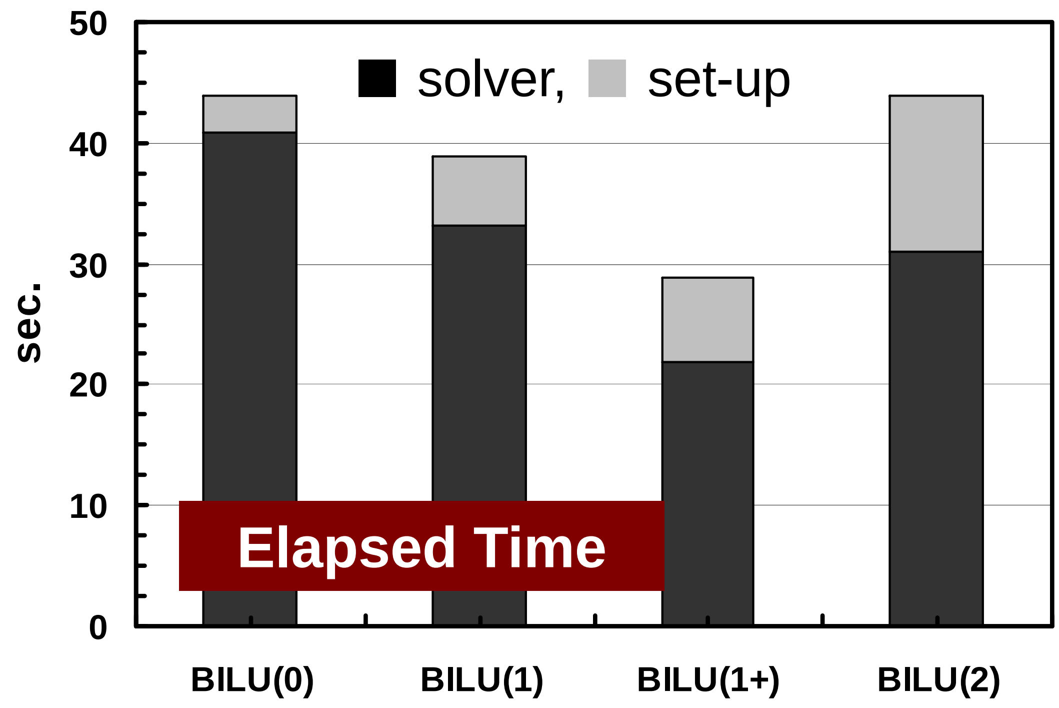
- Problem Size
  - 32,768 elements (except truss's) 117,708 DOF
- Preconditioned GPBiCG [Zhang, 1997]
  - for general matrices, although the matrices are SPD
    - BILU（0,1,2）, **Selective Fill-in (BILU(1+))**
- Environment
  - dual-core AMD Opteron 275 (2.2GHz)
  - F90 + MPI

Results: Single Core

107,811 DOF, $\lambda=10^3$, $\varepsilon=10^{-8}$







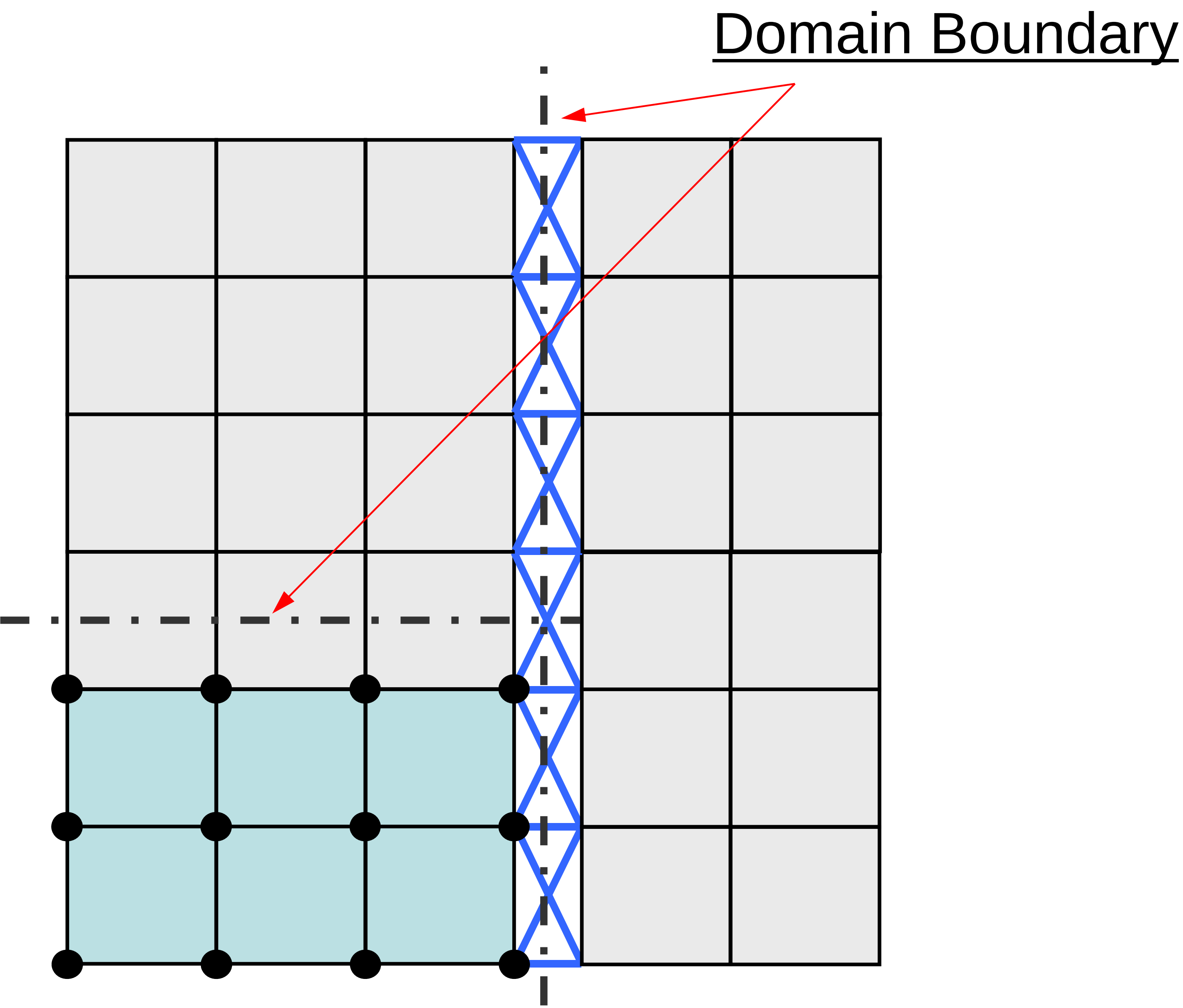
Selective Overlapping [KN 2007]

- Same rules in “selective fill-ins” are applied to extention of overlapping zones.
  - Similar concept as “selective blocking”

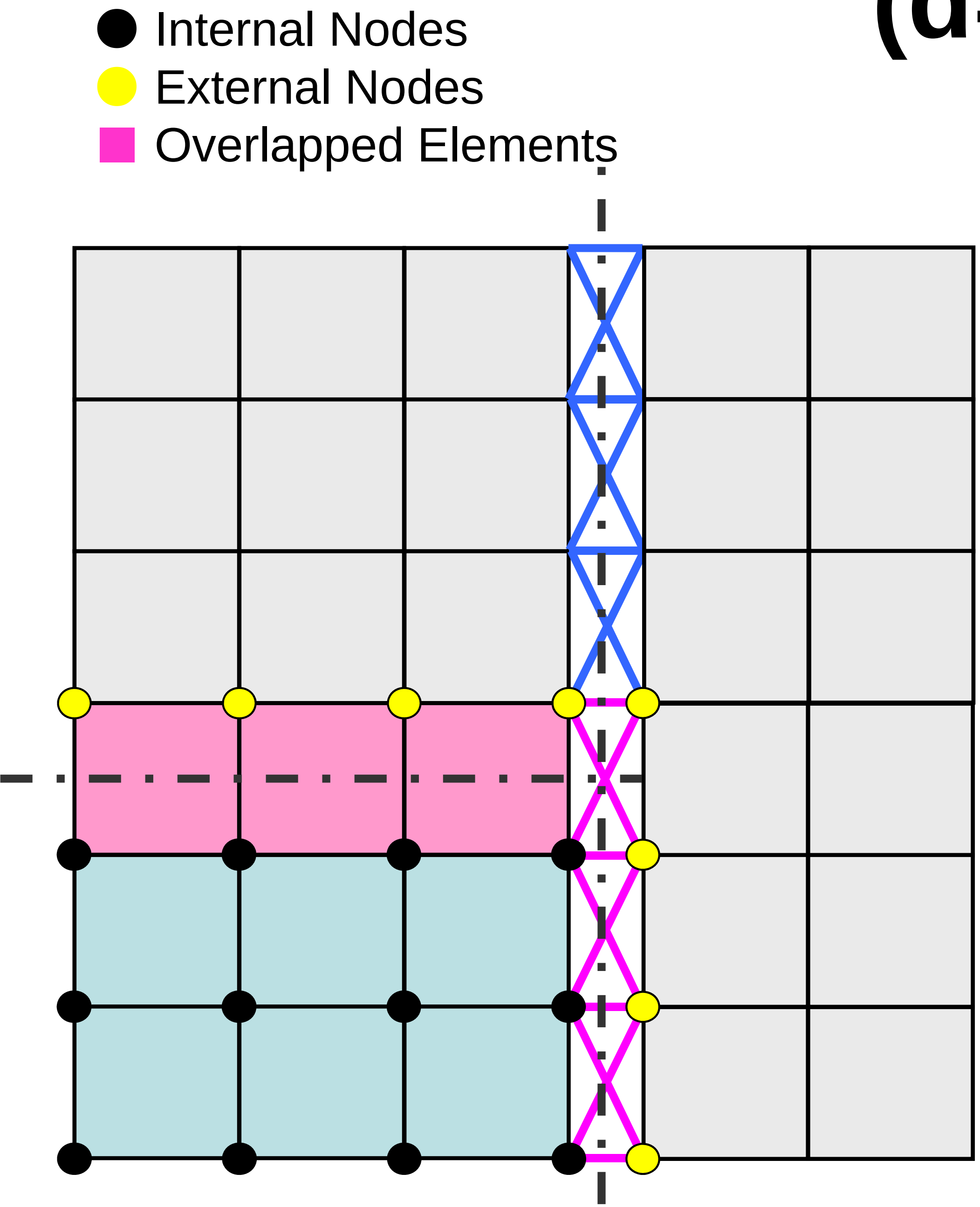
- In selective overlapping, extension of overlapping for nodes that are not connected to special elements for contact conditions is *delayed*.
- The increase in cost for computation and communication by extension of overlapped elements is suppressed.

Internal Nodes for Partitioning

● Internal Nodes

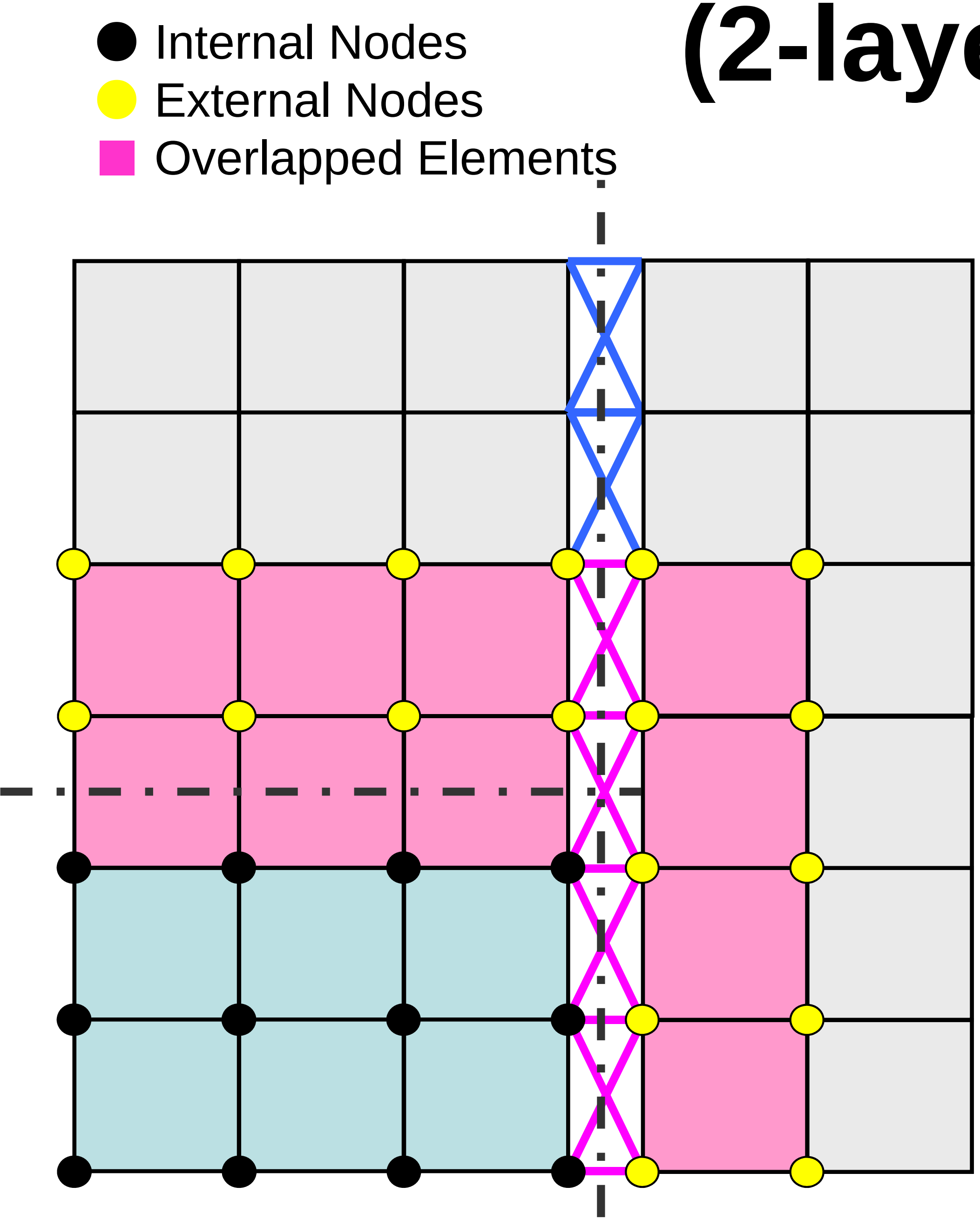


One-Layer Overlapping (d=0/1)

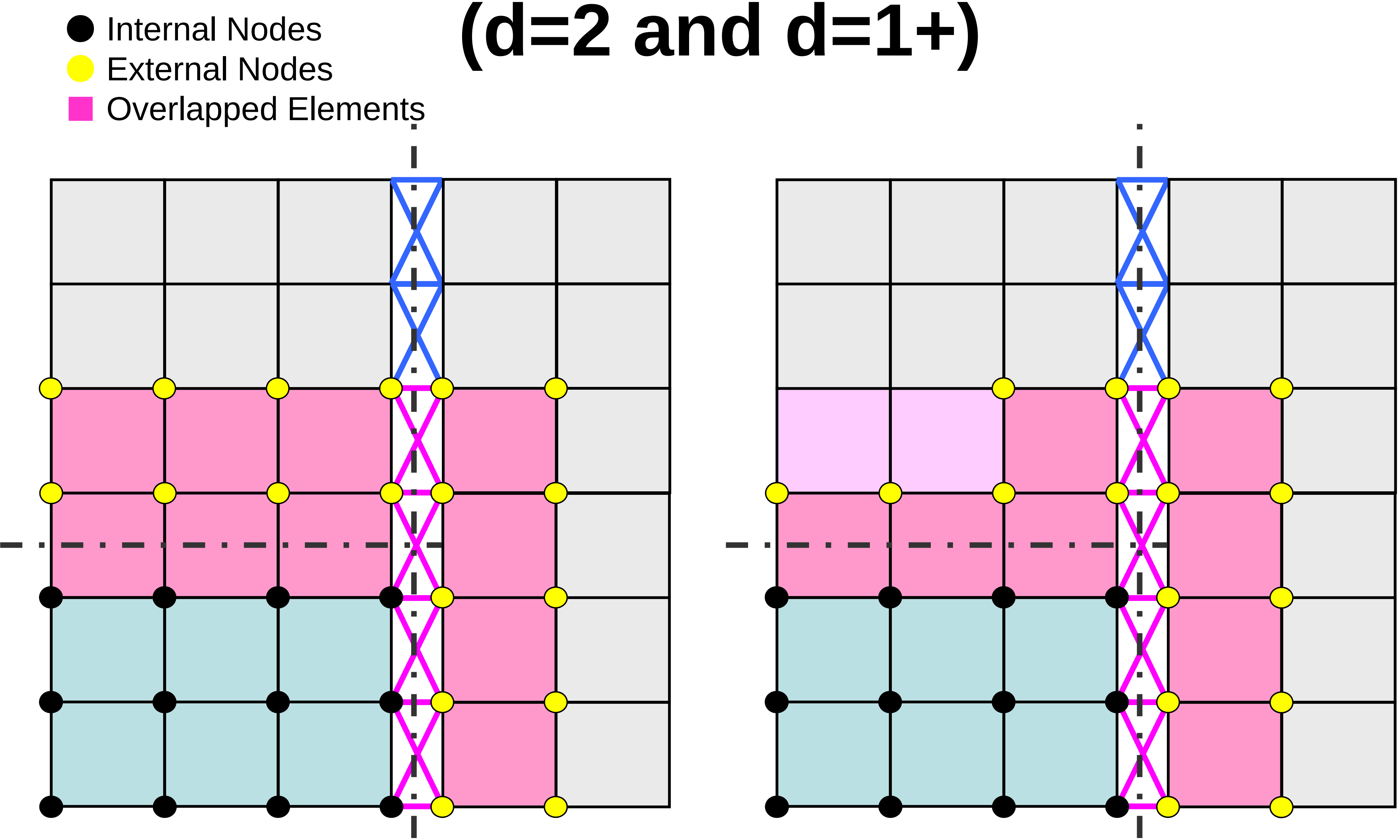


This is the general configuration of local data set for parallel FEM (one-layer of overlapping).

Extension of Overlapped Zones (2-layers: d=2)

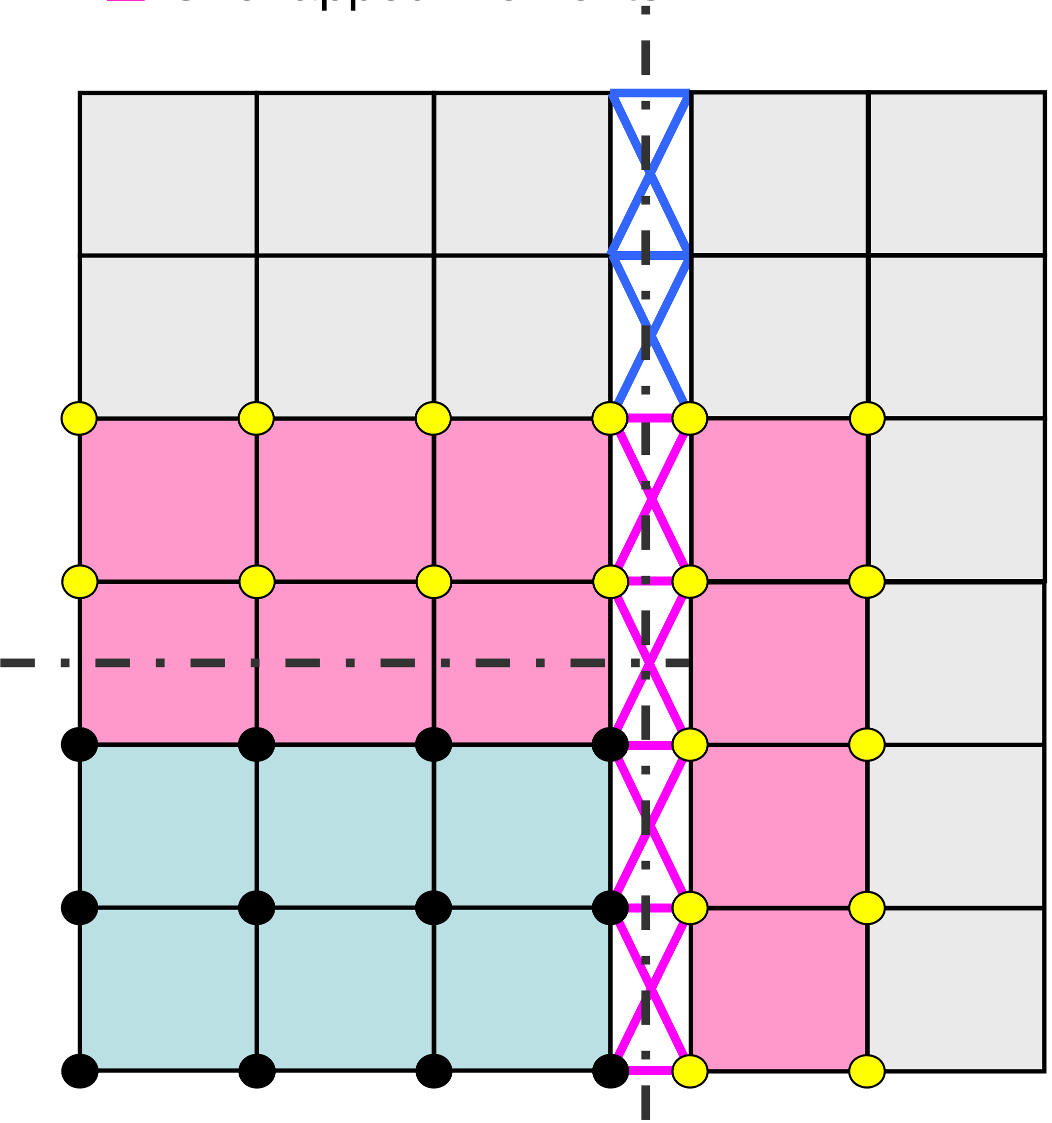


Extension of Overlapped Zones (d=2 and d=1+)



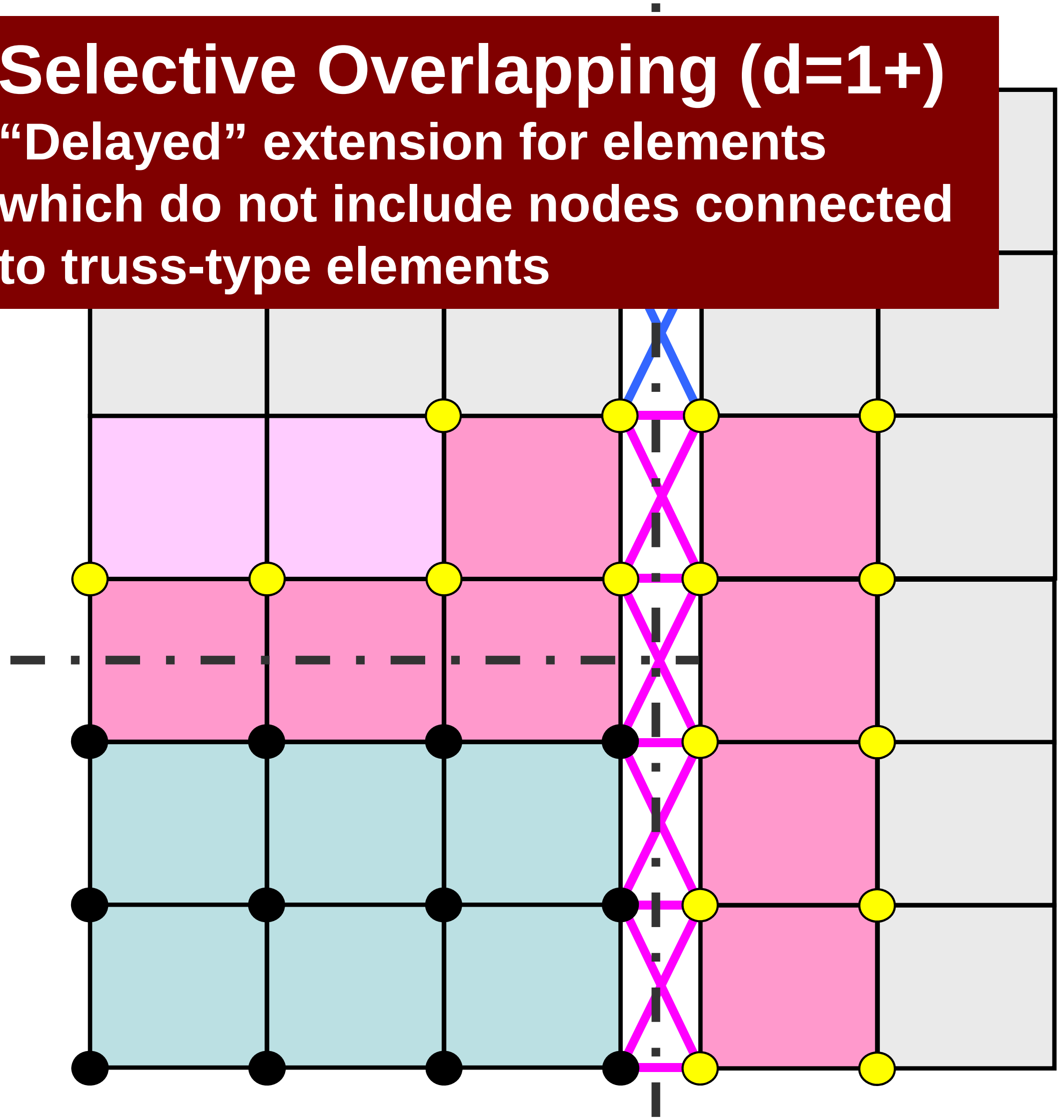
Extension of Overlapped Zones (d=2 and d=1+)

- Internal Nodes
- External Nodes
- Overlapped Elements



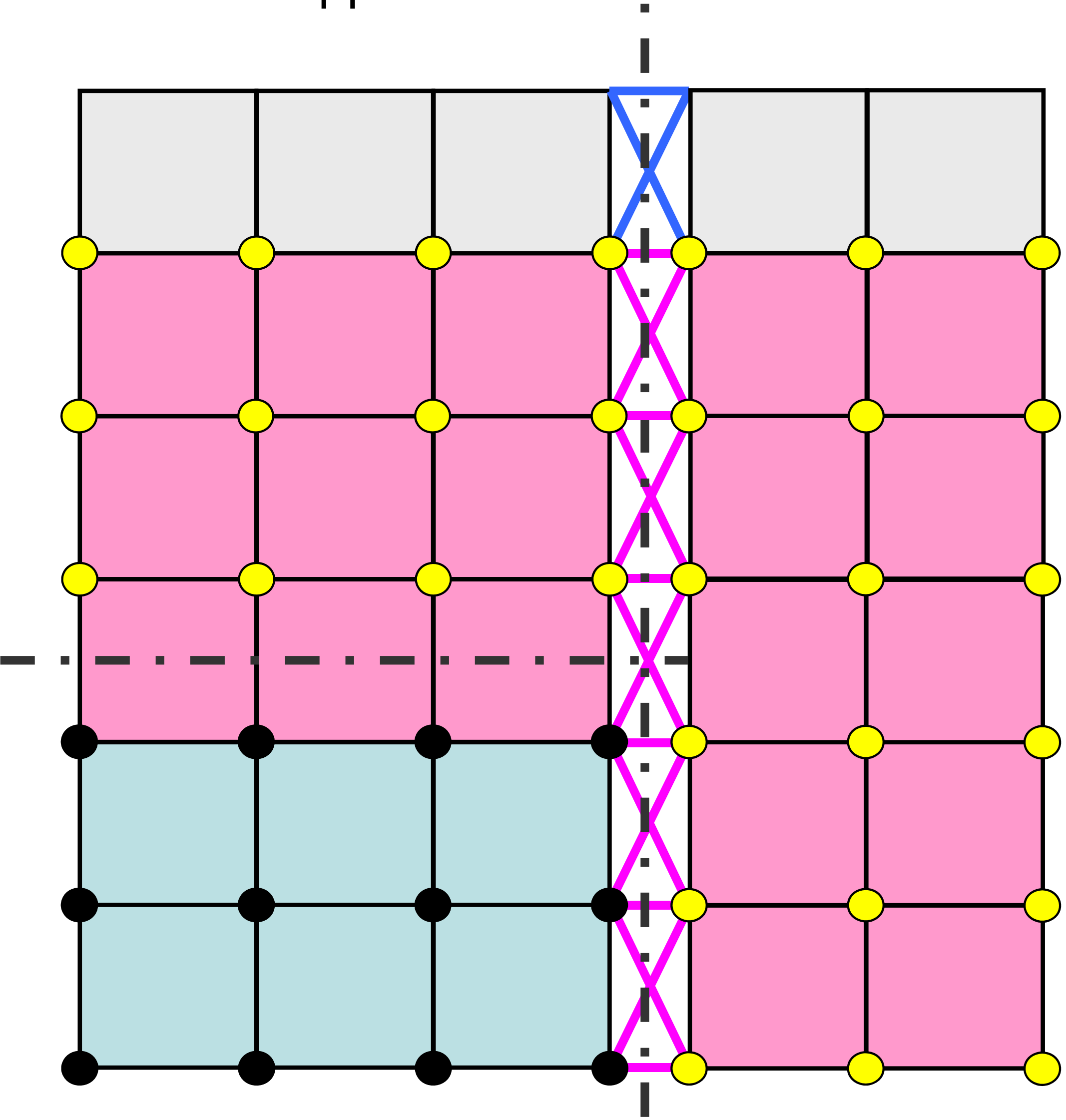
Selective Overlapping (d=1+)

“Delayed” extension for elements which do not include nodes connected to truss-type elements

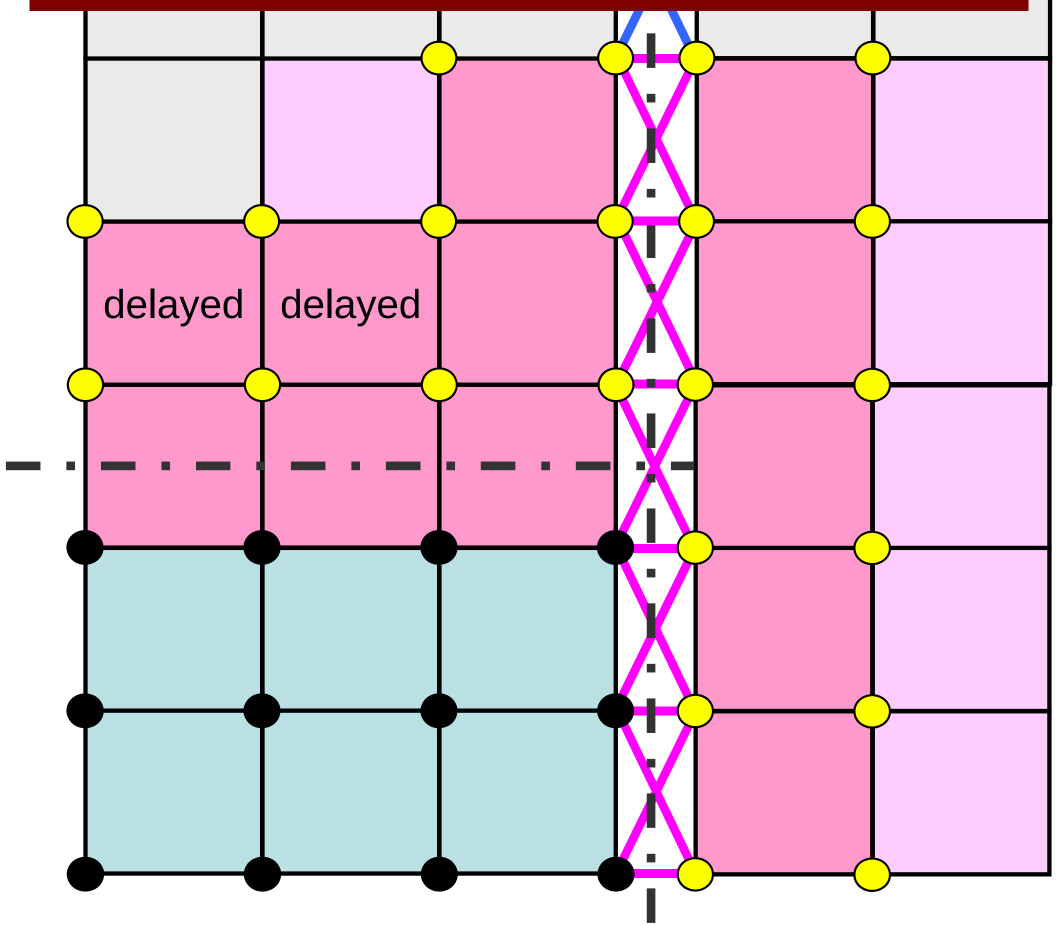


Extension of Overlapped Zones (d=3 and d=2+)

- Internal Nodes
- External Nodes
- Overlapped Elements



Selective Overlapping (d=2+)
Reduced cost for computations and communications



BILU with selective fill-in/overlapping

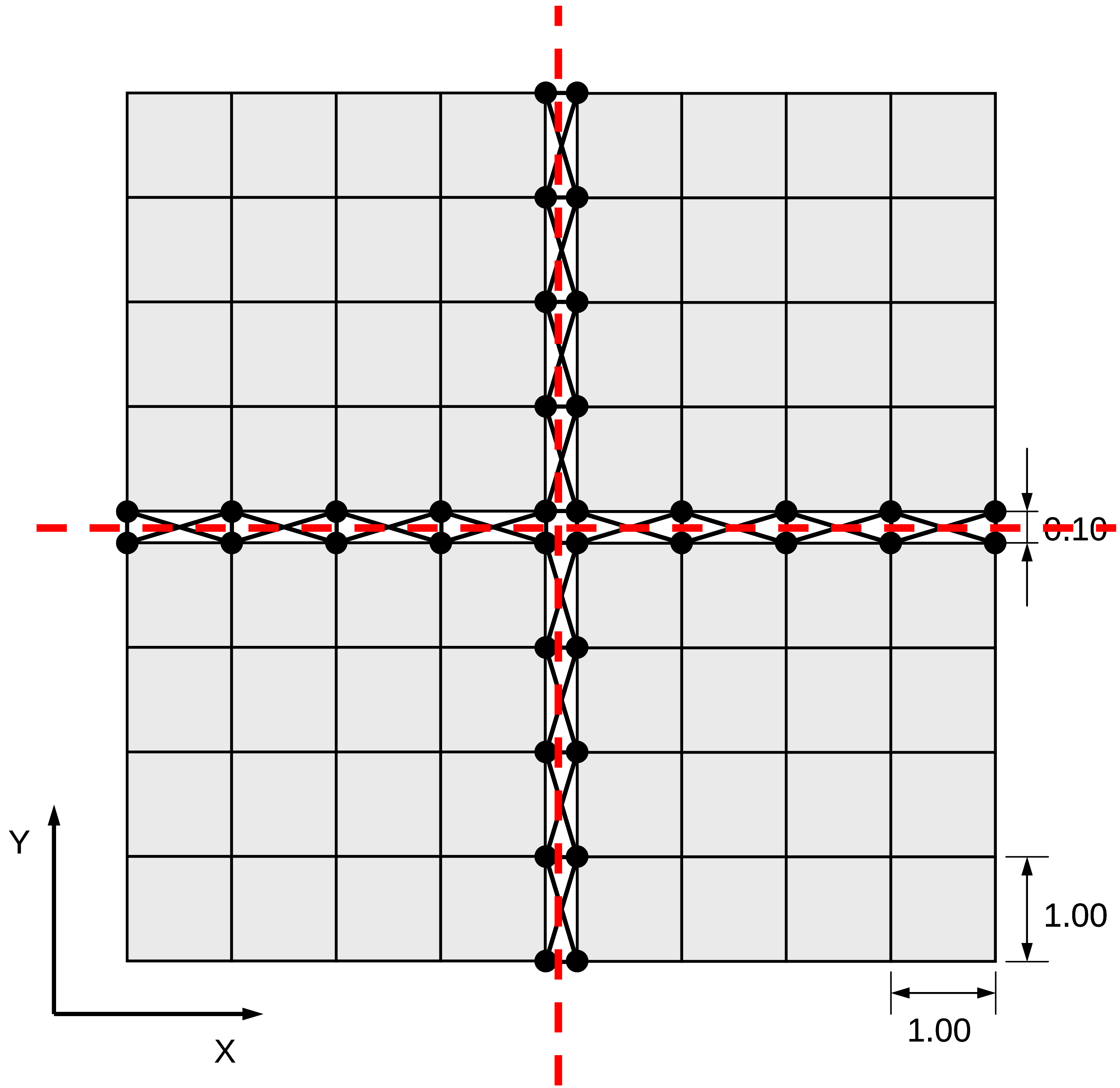
- **BILU (p)-(d)**
  - **p** level of fill-ins (0, 1, 1+, 2, 2+ …)
  - **d** depth of overlapping (0, 1, 1+, 2, 2+ …)

Summary of Problem Setting Multiple Cores

- Problem Size
  - Large: 1,000,000 elements (except truss's), 3,152,412 DOF
- Preconditioned GPBiCG [Zhang, 1997]
  - for general matrices, although the matrices are SPD
  - Localized preconditioning (block Jacobi type)
    - BILU（0,1,2）, **Selective Fill-in (BILU(1+))**
- Partitioning
  - GeoFEM-based local data structure: http://geofem.tokyo.rist.or.jp/
  - Recursive Coordinate Bisection（RCB）: 8~64
    - **Selective Overlapping**
- Environment
  - 64-core AMD Opteron 275 (2.2GHz), Infiniband
  - F90 + MPI

Domain boundaries are on "truss's"

worst cases for convergence

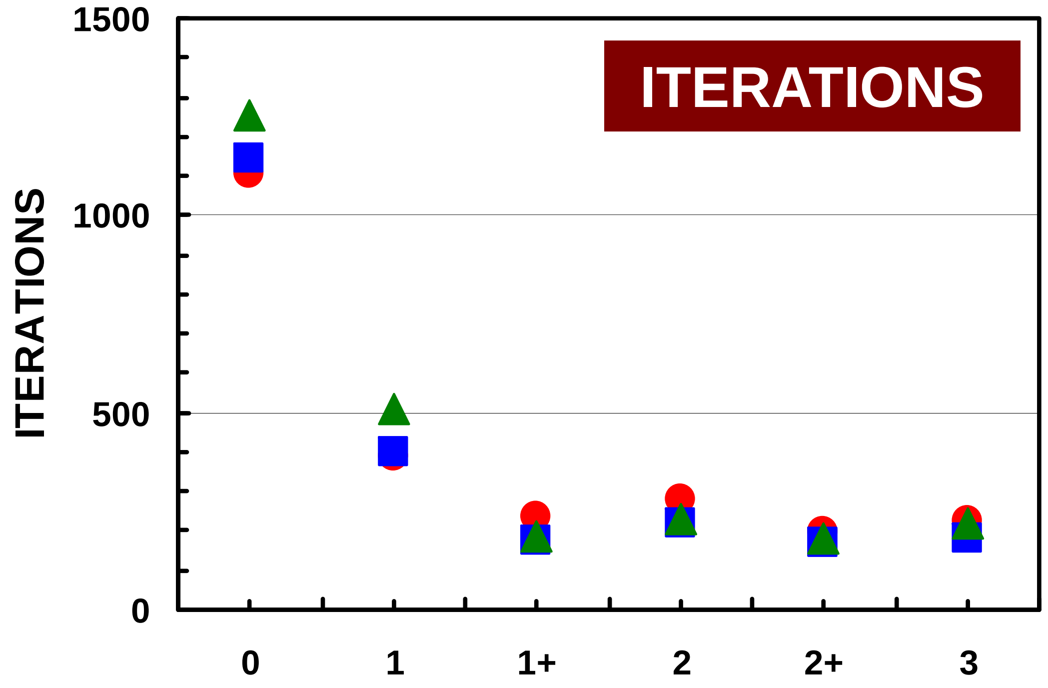


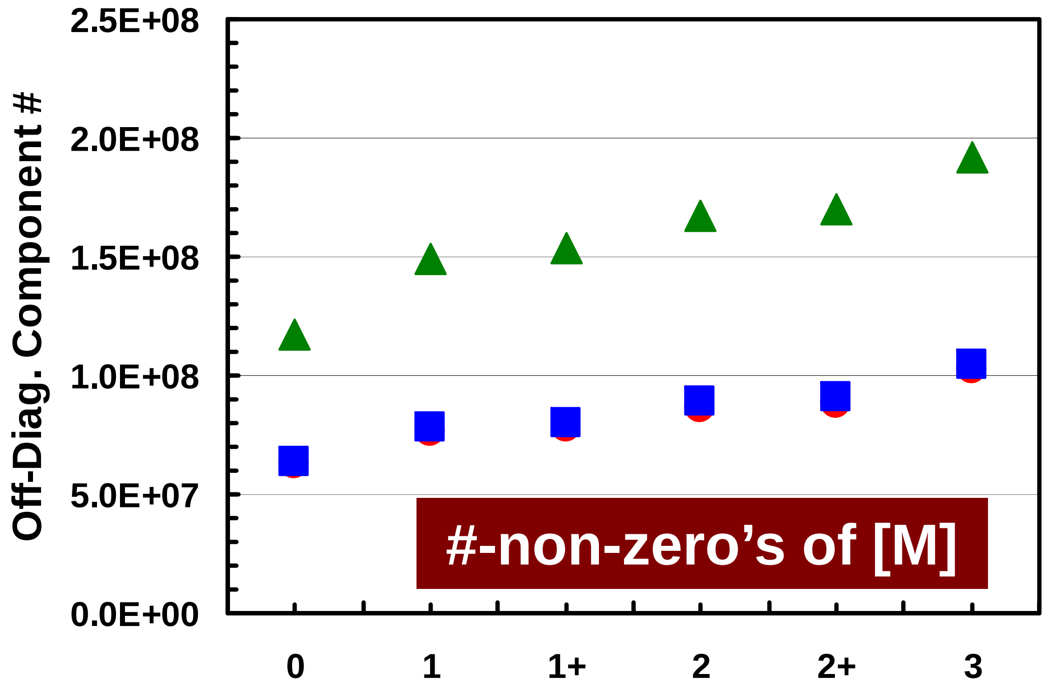
Results: 64 cores

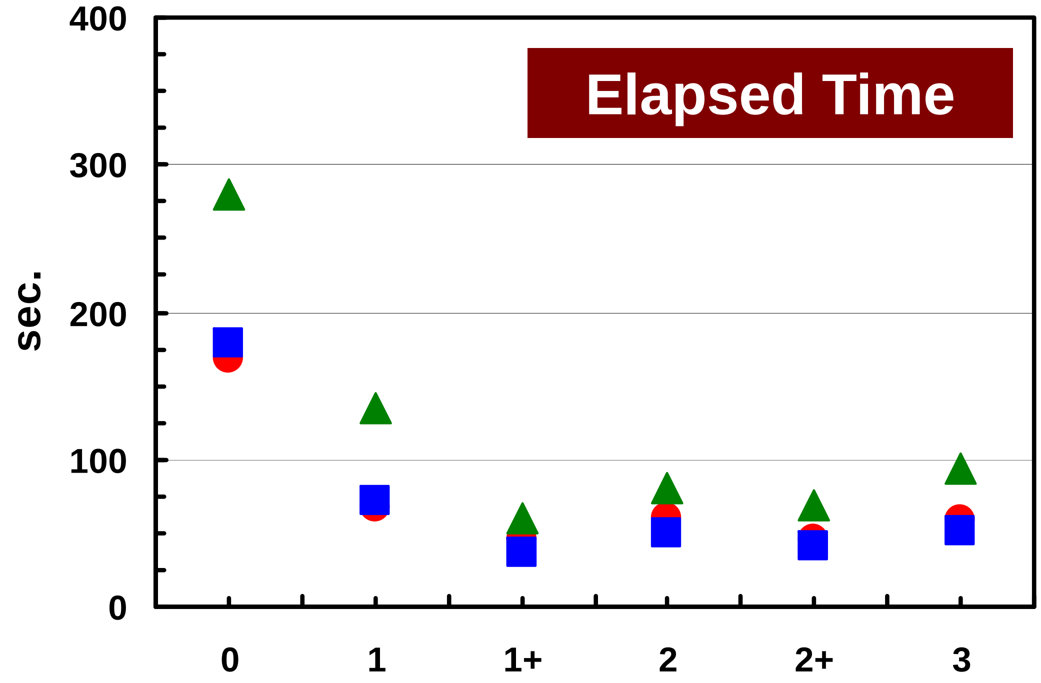
3,090,903 DOF, $\lambda=10^3$, $\varepsilon=10^{-8}$

Effect of Overlapping

- BILU(1)-(d)
- BILU(1+)-(d)
- BILU(2)-(d)





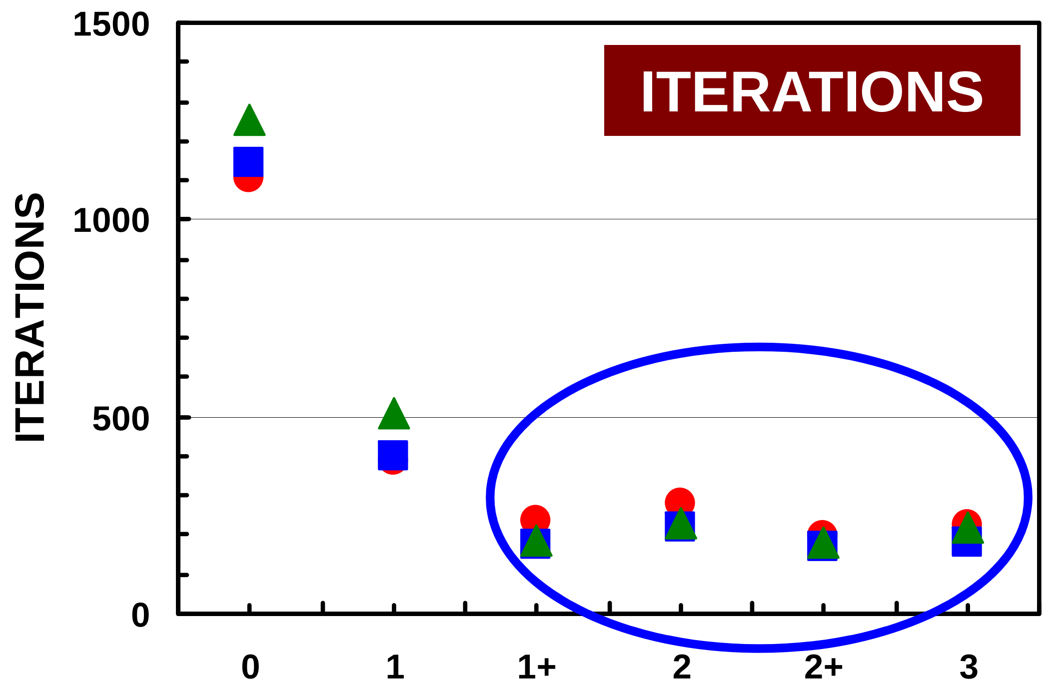


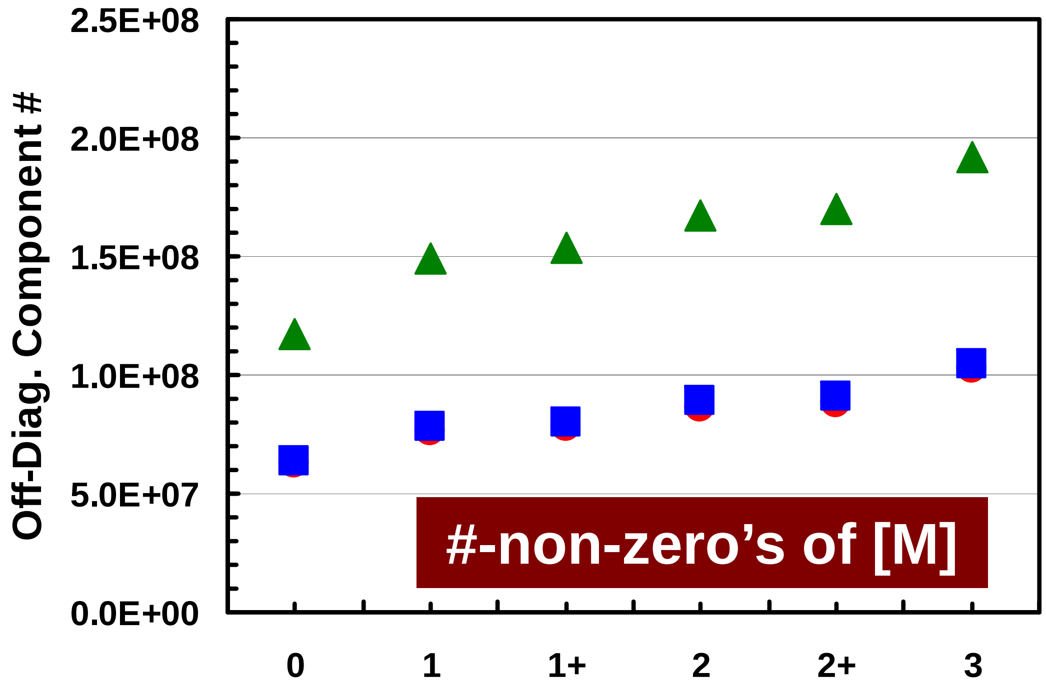
Results: 64 cores

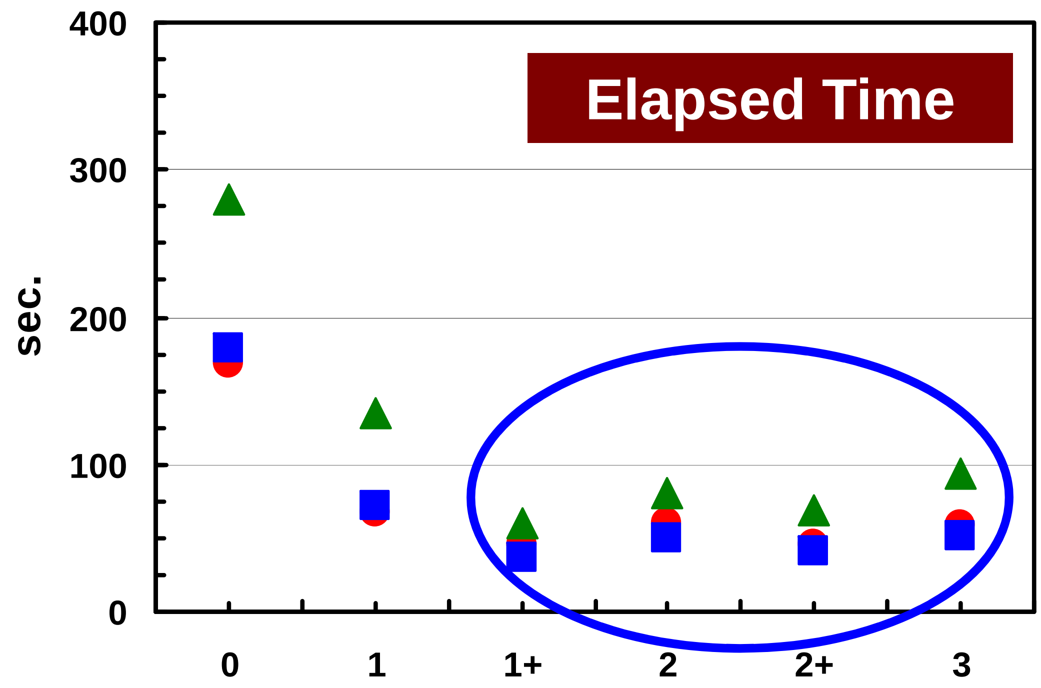
3,090,903 DOF, $\lambda=10^3$, $\varepsilon=10^{-8}$

Effect of Overlapping

● BILU(1)-(d)
■ BILU(1+)-(d)
▲ BILU(2)-(d)



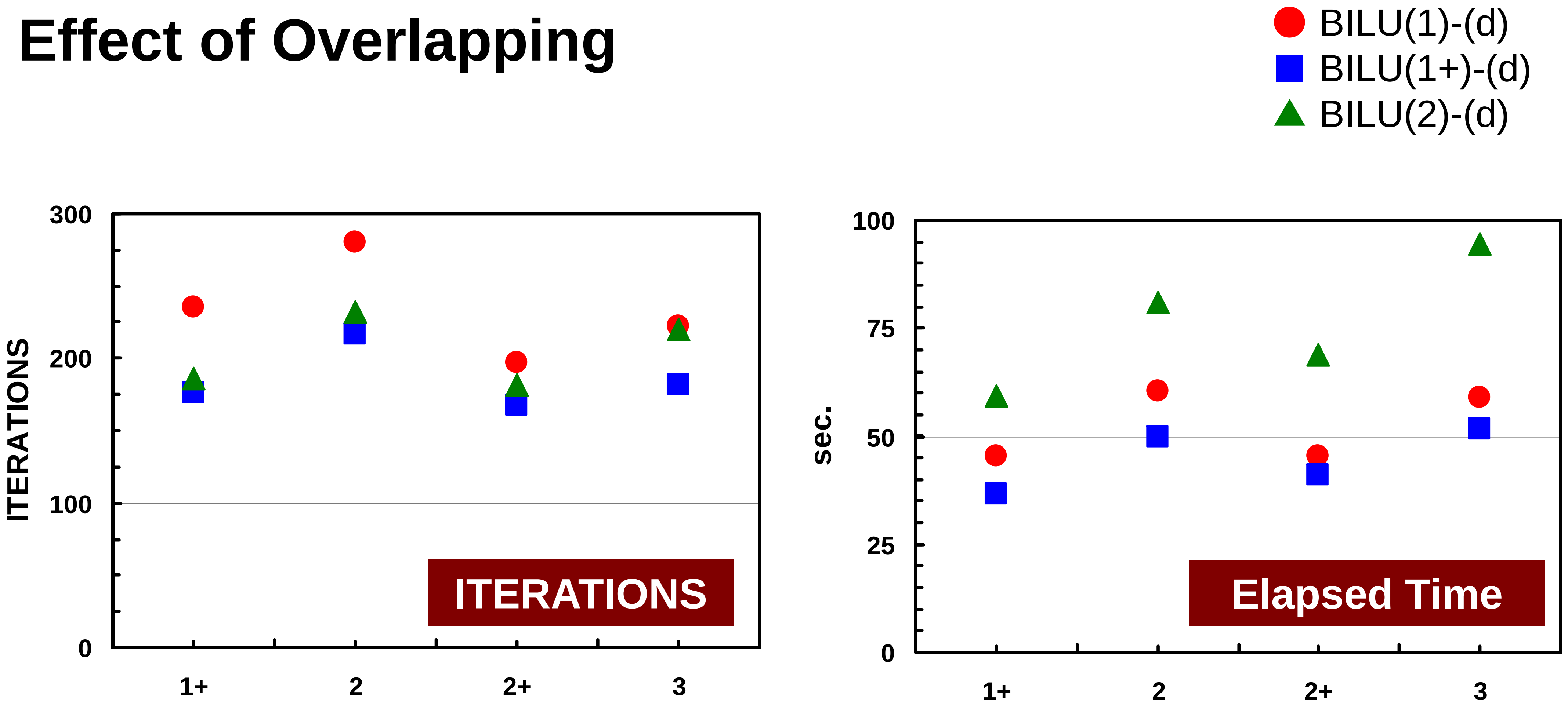




Results: 64 cores

3,090,903 DOF, $\lambda=10^3$, $\varepsilon=10^{-8}$

Effect of Overlapping



Summary

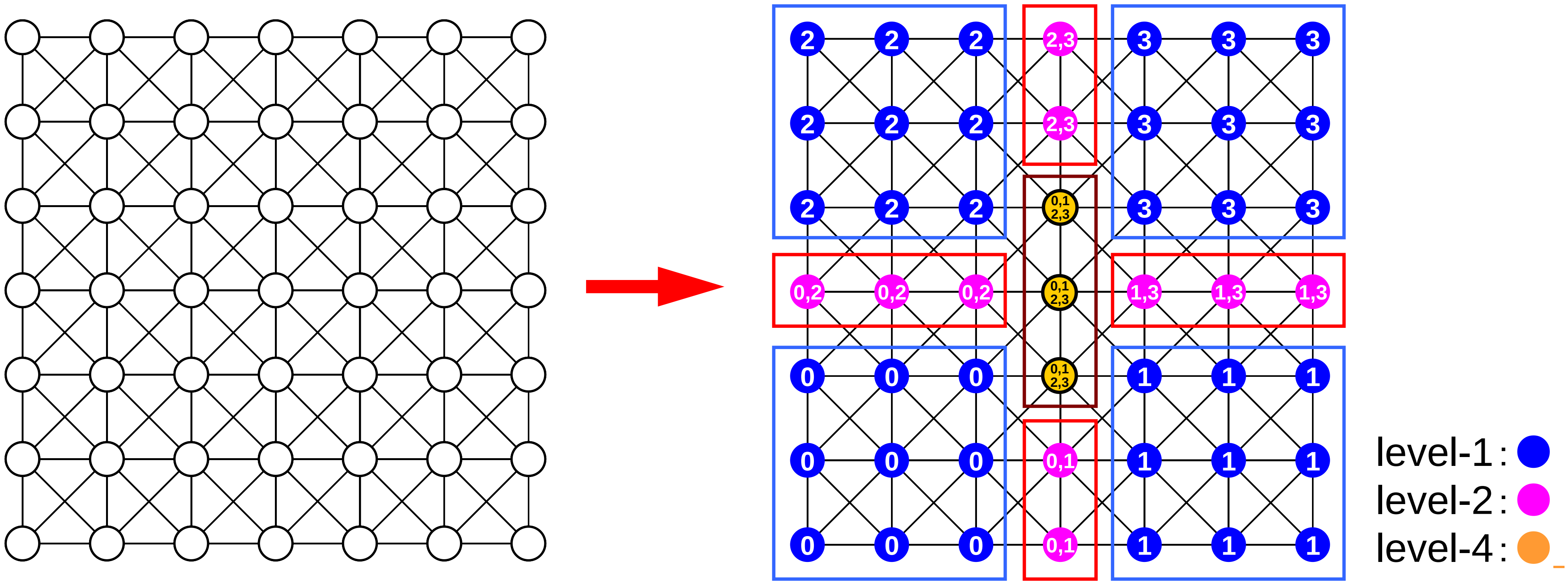
- Selective Fill-ins
- Selective Overlapping
  - Features of FEM applications (element-by-element) are utilized
  - Factorization processes are executed according to information of each element
    - much cheaper than ILUT-based methods, where dropping rules are applied after forming entire matrix

- Generally, BILU(1+)-(1+) is robust and efficient
- Significant improvement of convergence if **d** (depth of overlapping) is 0⇒1⇒1+.

- Background
  - Selective Blocking
  - More General Problems
    - Extension of Overlapped Zones
- Preconditioning/Partitioning Methods
  - Target Application
  - Selective Fill-ins, Selective Overlapping
- **HID**
  - **Hierarchical Interface Decomposition**
- Extended HID
- Fields with Heterogeneity

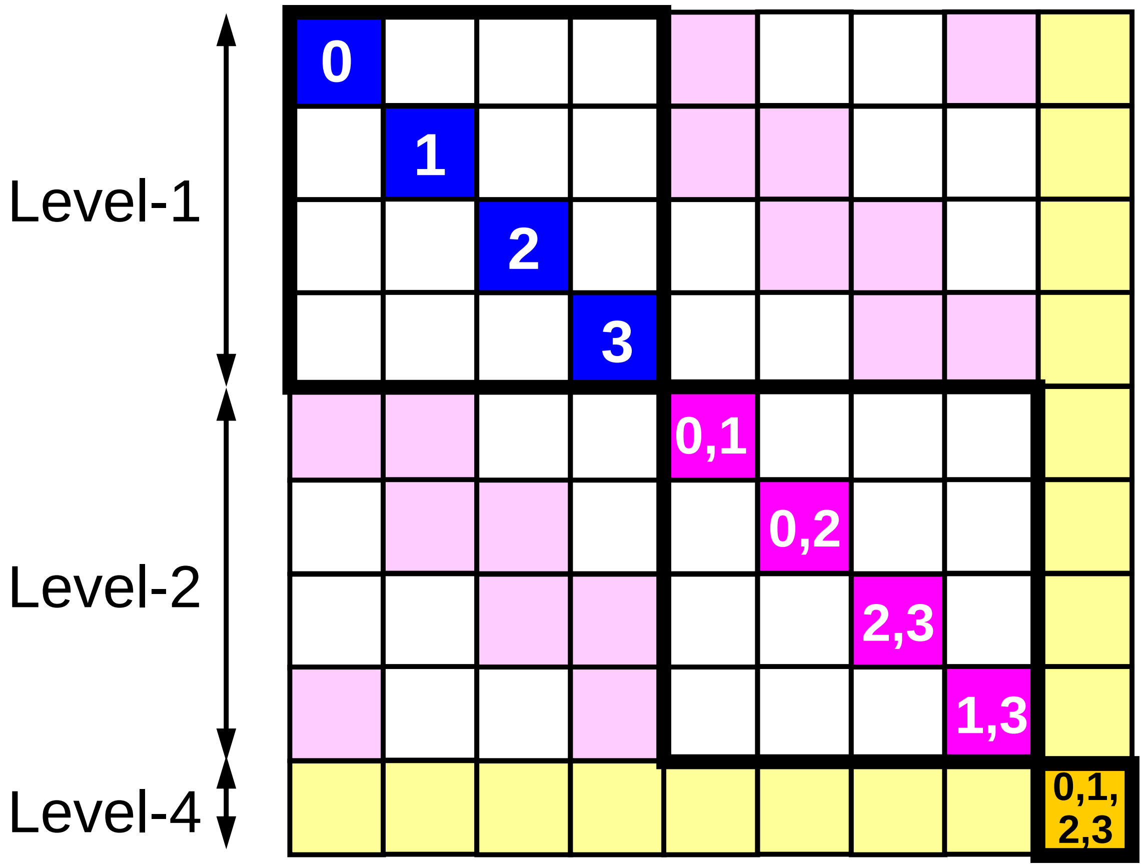
HID: Hierarchical Interface Decomposition [Henon & Saad 2007]

- Multilevel Domain Decomposition
  - Extension of Nested Dissection
- Non-overlapping at each level: Connectors, Separators
- Suitable for Parallel Preconditioning Method



Parallel ILU for each Connector at each LEVEL

- The unknowns are reordered according to their **level** numbers, from the lowest to highest.
- The block structure of the reordered matrix leads to natural parallelism if ILU/IC decompositions or forward/backward substitution processes are applied.



Communications at Each Level Forward Substitutions

```
  do lev= 1, LEVELtot
    do i= LEVindex(lev-1)+1, LEVindex(lev)
      SW1= WW(3*i-2,R); SW2= WW(3*i-1,R); SW3= WW(3*i  ,R)
      isL= INL(i-1)+1; ieL= INL(i)
      do j= isL, ieL
        k= IAL(j)
        X1= WW(3*k-2,R); X2= WW(3*k-1,R); X3= WW(3*k  ,R)
        SW1= SW1 - AL(9*j-8)*X1 - AL(9*j-7)*X2 - AL(9*j-6)*X3
        SW2= SW2 - AL(9*j-5)*X1 - AL(9*j-4)*X2 - AL(9*j-3)*X3
        SW3= SW3 - AL(9*j-2)*X1 - AL(9*j-1)*X2 - AL(9*j  )*X3
      enddo
      X1= SW1; X2= SW2; X3= SW3
      X2= X2 - ALU(9*i-5)*X1
      X3= X3 - ALU(9*i-2)*X1 - ALU(9*i-1)*X2
      X3= ALU(9*i  )*  X3
      X2= ALU(9*i-4)*( X2 - ALU(9*i-3)*X3 )
      X1= ALU(9*i-8)*( X1 - ALU(9*i-6)*X3 - ALU(9*i-7)*X2)
      WW(3*i-2,R)= X1; WW(3*i-1,R)= X2; WW(3*i  ,R)= X3
    enddo

    call SOLVER_SEND_RECV_3_LEV(lev,…):

  enddo
```

Additional Comm.

Communications using Hierarchical Comm. Tables.

HID: Hierarchical Interface Decomposition [Henon & Saad 2007]

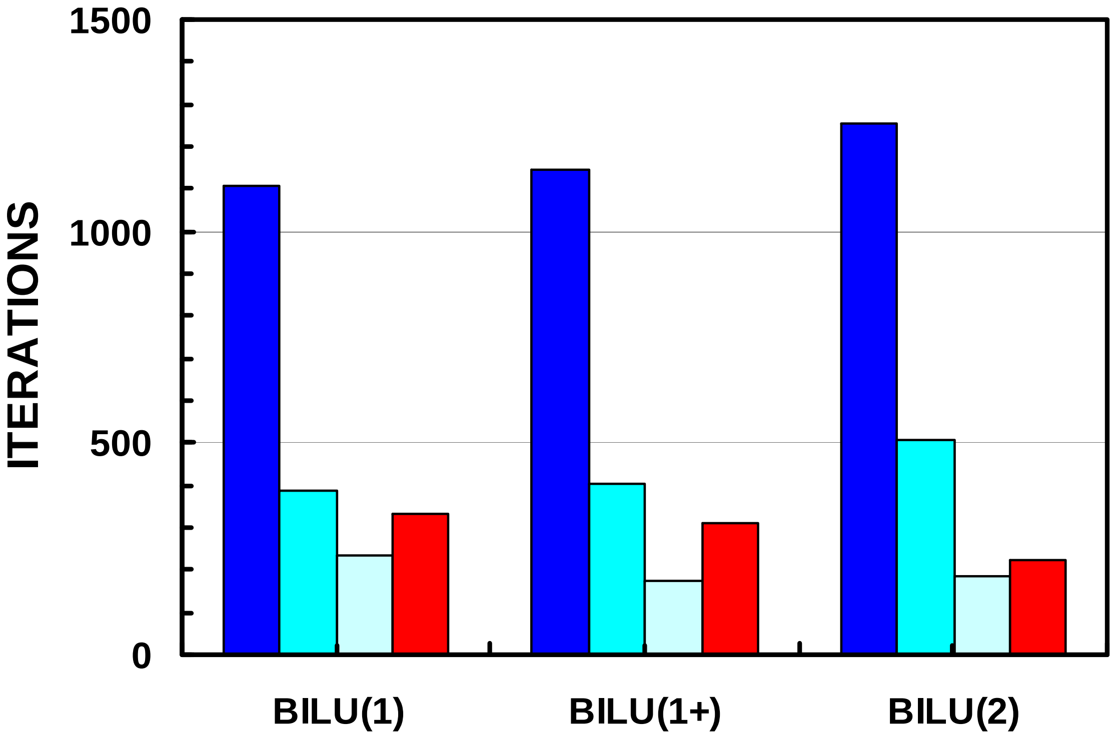
- Multilevel Domain Decomposition
- Non-overlapped Approach
  - see the paper for detailed information
- Suitable for Parallel Preconditioning Method

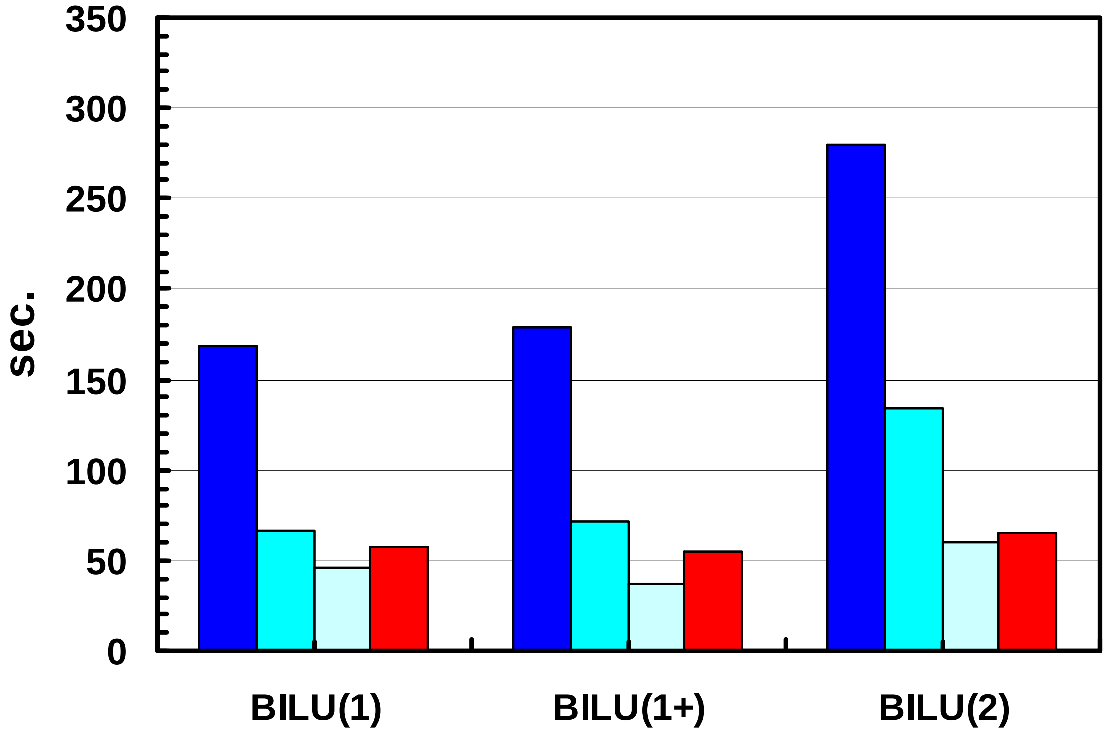
- Comparison with Selective Overlapping
  - Cost of HID corresponds to that of (**d=0**) or (**d=1**), but as robust as (**d=1+**) or (**d=2**)
  - More robust than Block Jacobi.

Results: 64 cores
Contact Problems

3,090,903 DOF

BILU(p)-(0): Block Jacobi
BILU(p)-(1)
BILU(p)-(1+)
HID
GPBiCG

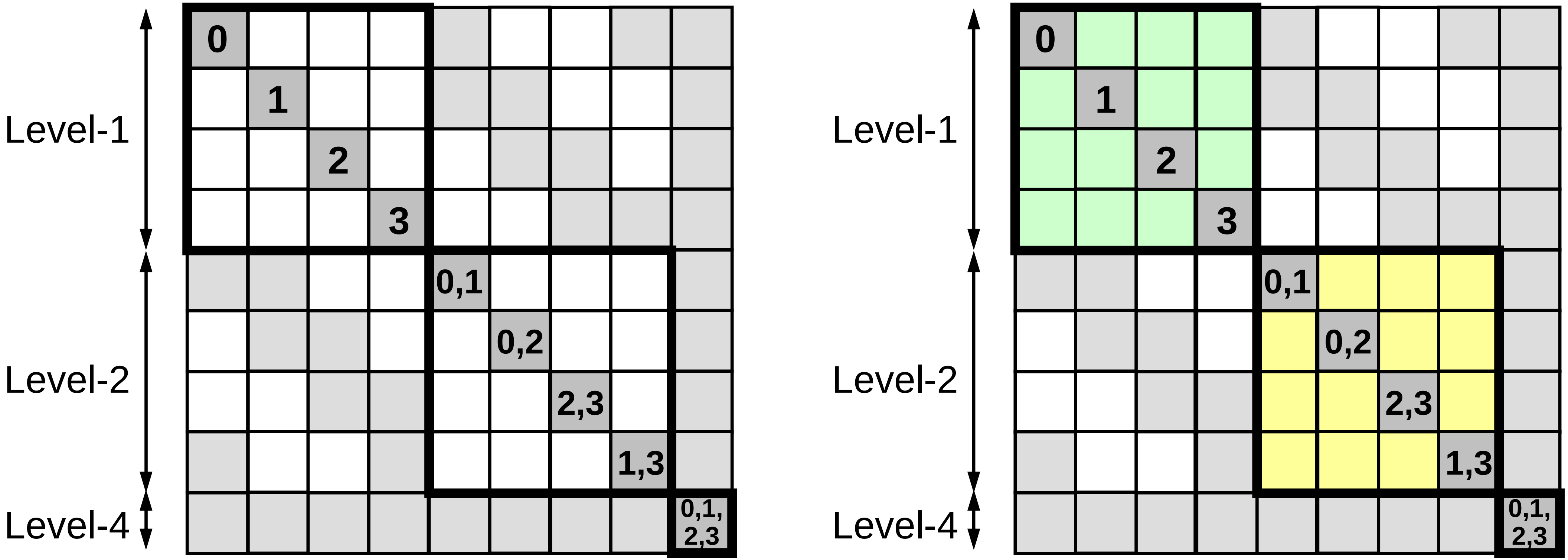




- Background
  - Selective Blocking
  - More General Problems
    - Extension of Overlapped Zones
- Preconditioning/Partitioning Methods
  - Target Application
  - Selective Fill-ins, Selective Overlapping
- HID
  - Hierarchical Interface Decomposition
- **Extended HID**
- Fields with Heterogeneity

Weakness of Original HID

- Original HID cannot consider the effects of fill-ins of higher order at boundary nodes.
  - – although it's perfect for parallel ILU(0).

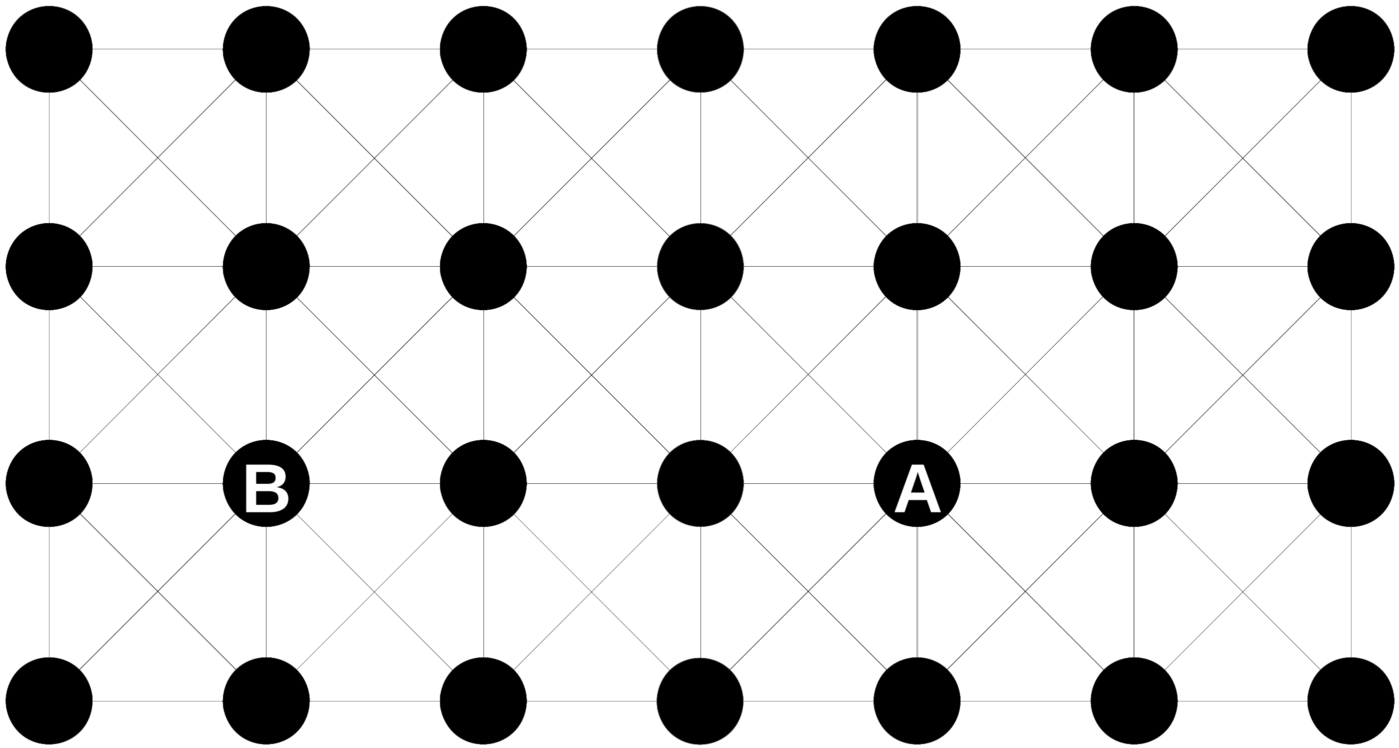


Extended Version of HID

- Extension of Overlapped Elements
- Thicker Layers of Separators

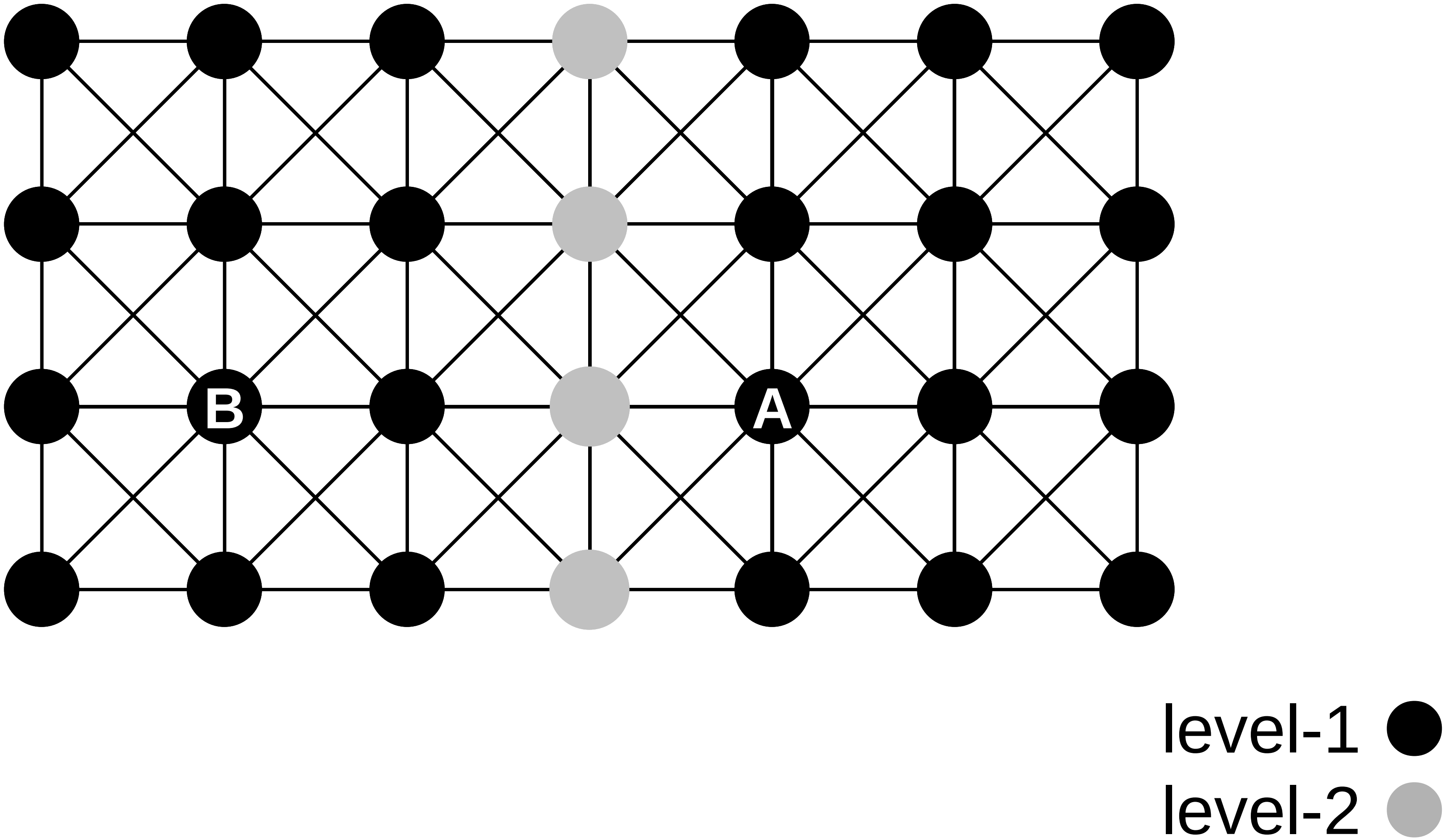
Sample Graph

(A) could be referred from (B)
for ILU(2) (depends on numbering)



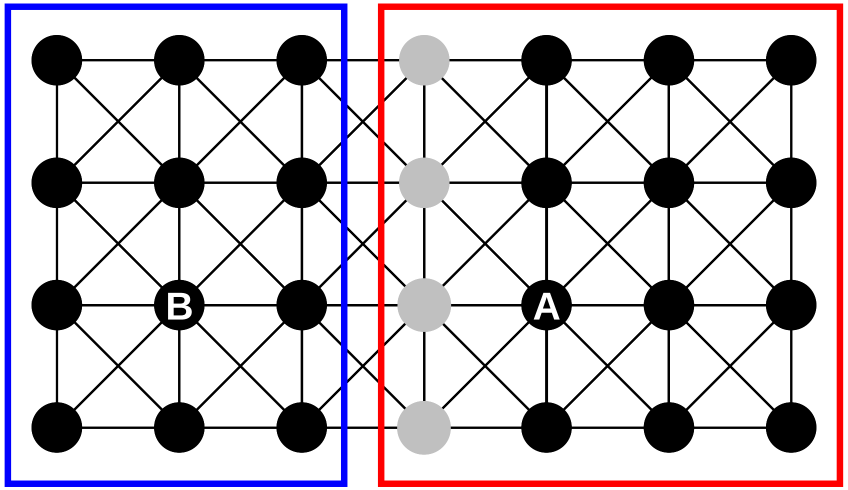
Sample Graph

(A) CANNOT be referred from (B) for ILU(2), because they are at same level and on different domain

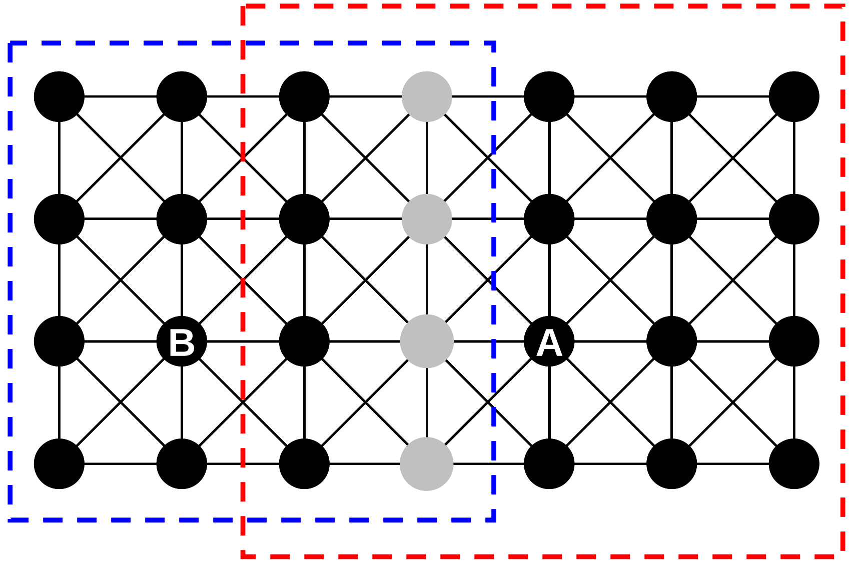


Domain Decomposition & Local Data Set

level-1 ●
level-2 ●



Node-based Domain Decomposition (Internal Nodes)

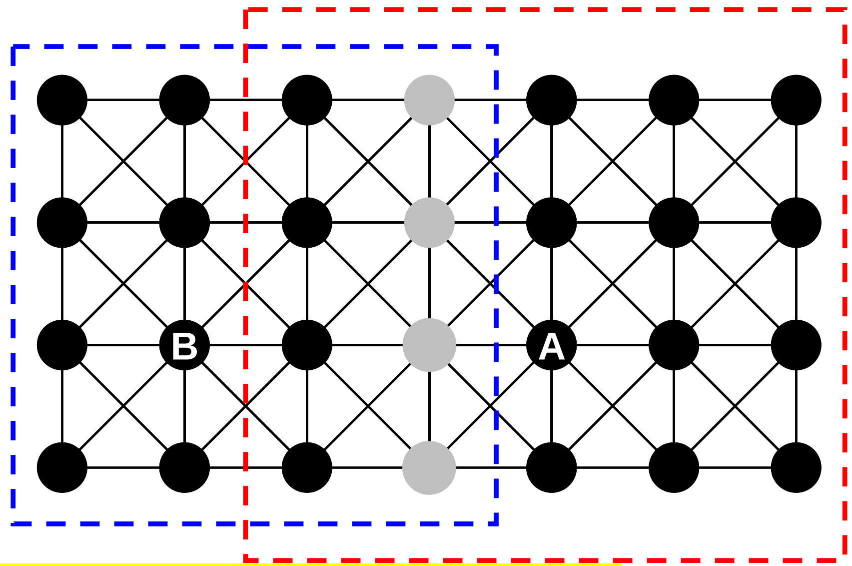


Distributed Local Data (Internal+External Nodes)

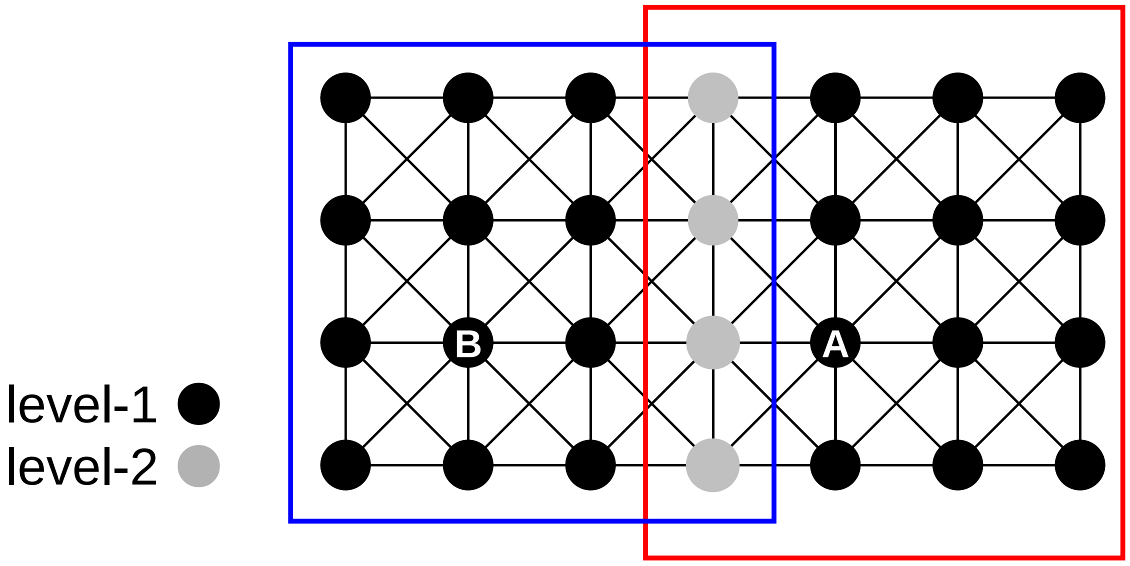
Original Local Data Set

- Original HID
  - NO overlapping/1-layer overlapping
  - cannot consider the effects of fill-ins of higher order for external nodes at *same level*.
    - Effect of “A” is not considered for “B” in BILU(2)

Distributed Local Data



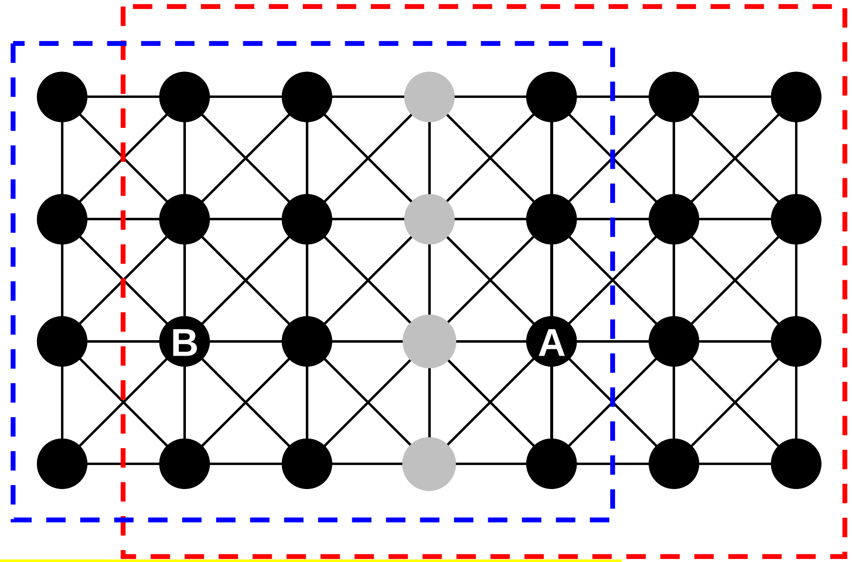
Range for “Global” Operations



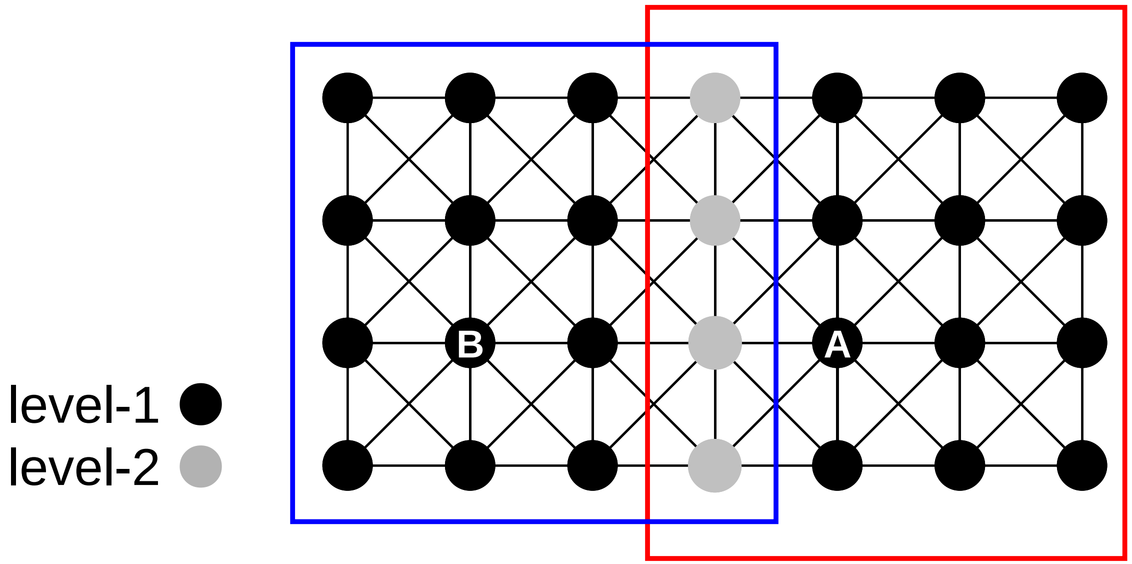
Remedy 1: Extension of Overlapping

- <u>Extension of Overlapping</u>
  - 2-layer overlapping
  - can consider the effects of fill-ins of higher order for external nodes at *same level*.
    - Effect of B can be considered for A in BILU(2)
  - **But still localized, Block Jacobi approach**
    - **because the value at "A" is not the most recent one**

Distributed Local Data



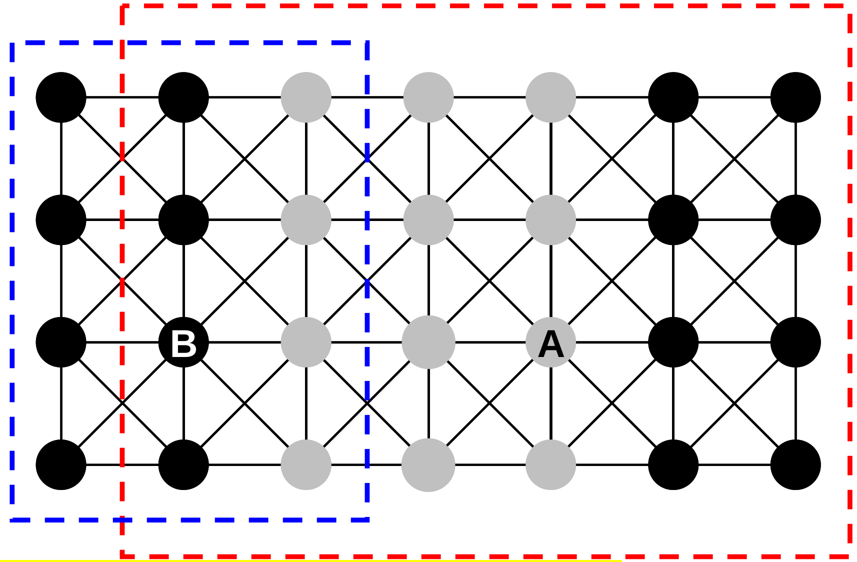
Range for "Global" Operations



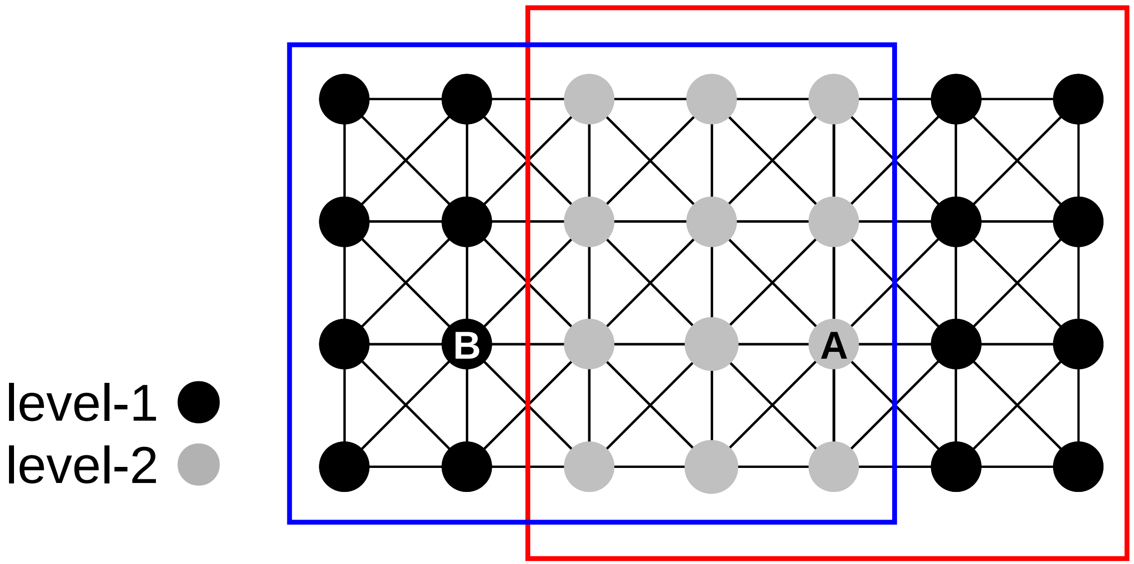
Remedy 2: Thicker Separator Layers

- <u>Thicker Separator</u>
  - <u>HID-new</u>
  - can consider the effects of fill-ins of higher order for external nodes at *same level*.
    - Effect of “A” can be considered for “B” in BILU(2)
  - **In global manner**
  - **seems to provide more robust convergence than Remedy 1.**
  - **difficulty for load-balancing**

Distributed Local Data



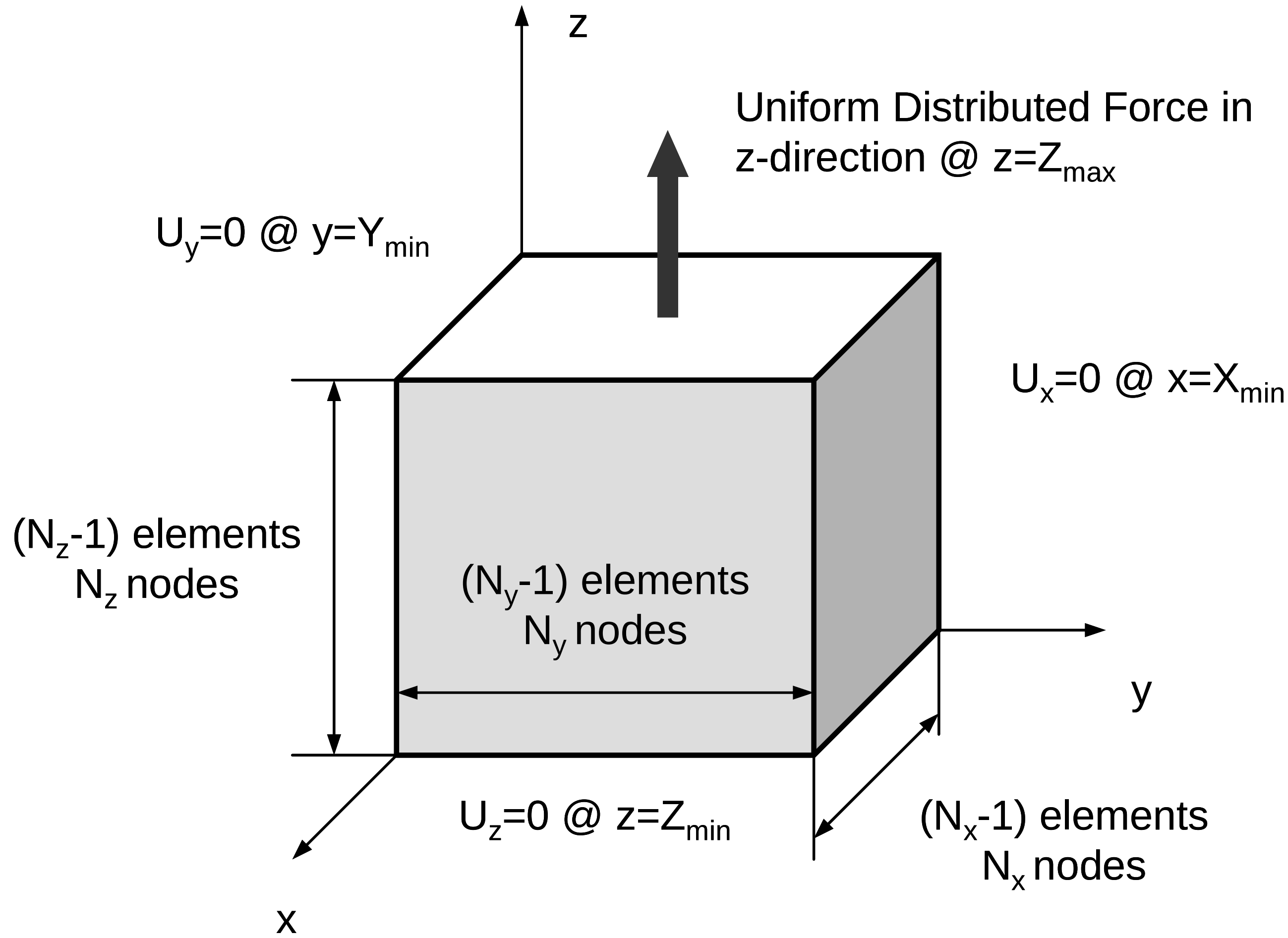
Range for “Global” Operations



- Background
  - Selective Blocking
  - More General Problems
    - Extension of Overlapped Zones
- Preconditioning/Partitioning Methods
  - Target Application
  - Selective Fill-ins, Selective Overlapping
- HID
  - Hierarchical Interface Decomposition
- Extended HID
- **Fields with Heterogeneity**

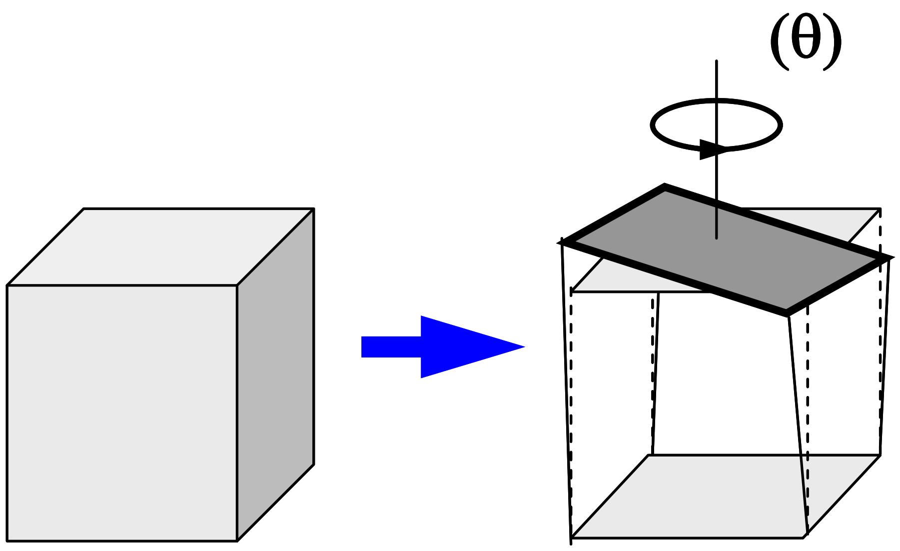
Target Application (1/3)

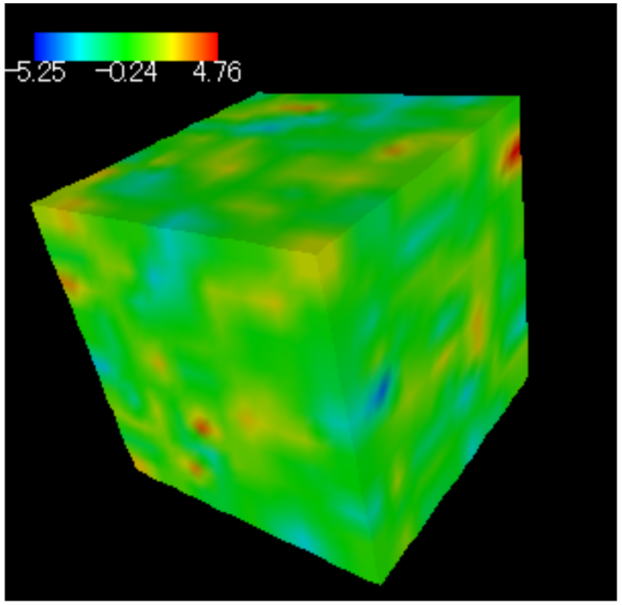
- 3D linear elastic problem with locally distorted elements



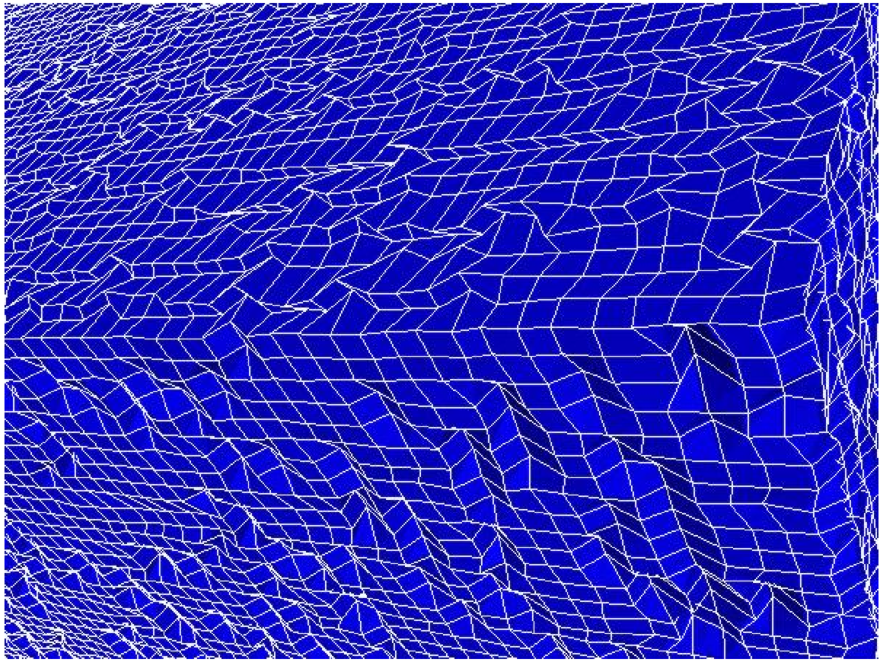
Target Application (2/3)

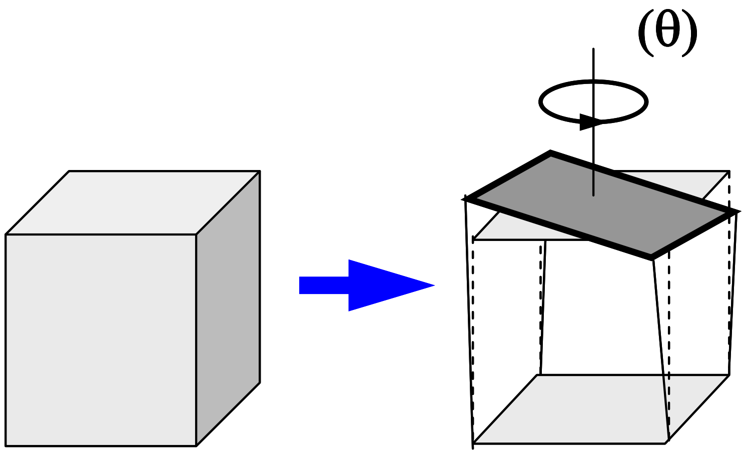
- 3D linear elastic problem with locally distorted elem's
- Initial mesh: cube
  - distortion around Z-axis of each element
- Local Heterogeneity
  - local "intensity" of distortion
  - sequential Gauss algorithm [Deutsch & Journel 1988]





Target Application (3/3)

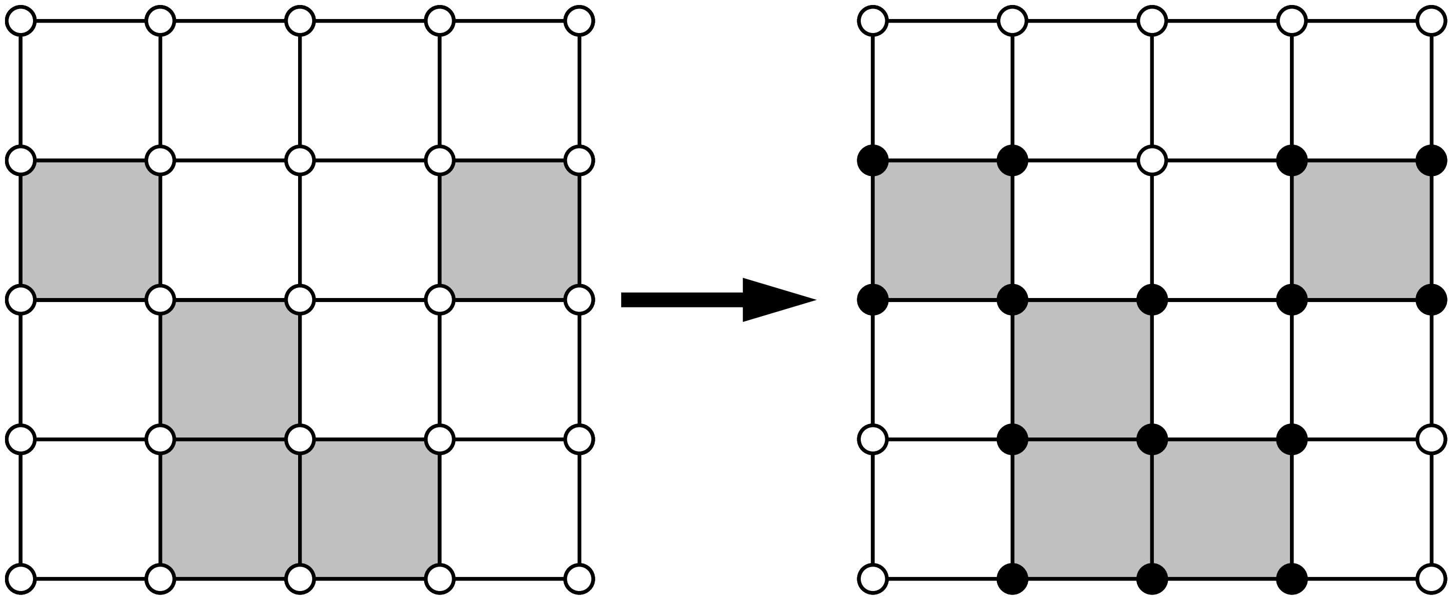




- 3D linear elastic problem with locally distorted elements
- Very ill-conditioned for significant distortion
  - requires BILU(2) or higher
  - semi indefinite
- Maximum distortion= 200 deg.
- Strong Scaling
  - 128^3 Elements
  - 6,440,067 Unknowns

Selective Fill-ins/Overlapping with Threshold

- **BILU (p,ω)-(d,α)**
  - If E > ω selective fill-ins is applied
  - If E > α selective overlappng is applied



●: fill-ins of higher order and extension of overlapping are allowed on these nodes

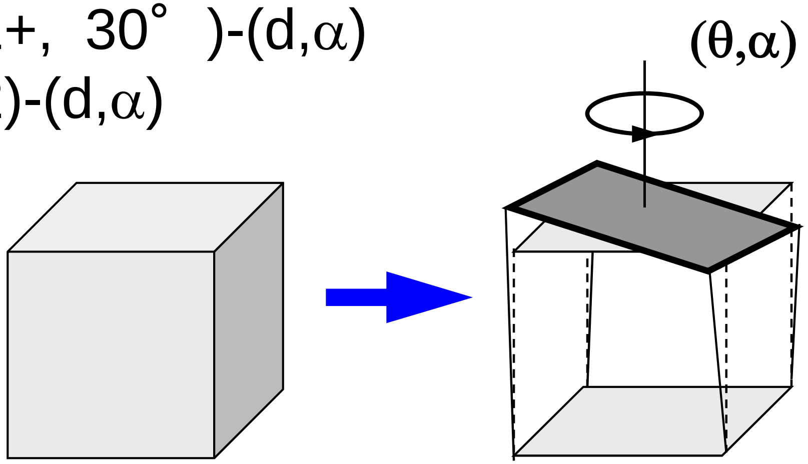
Results: 64 cores Distorted Meshes

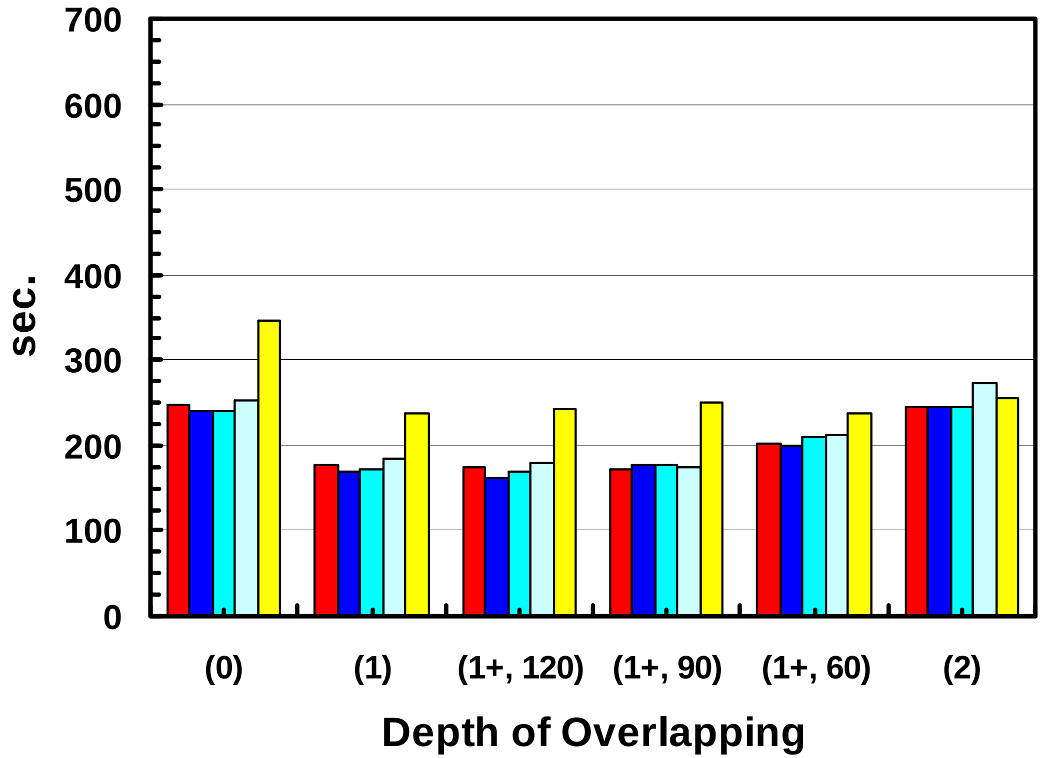
BILU(p,θ)-(d,α)

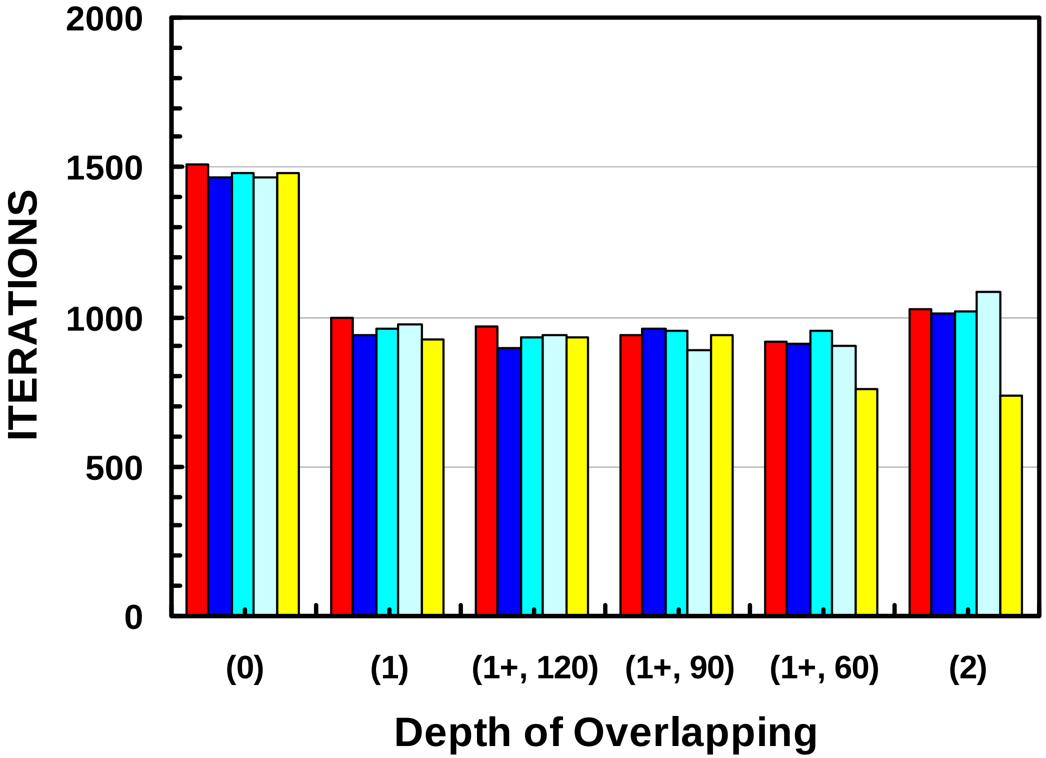
3,090,903 DOF

MAX distortion: **150-deg.**

BILU(1)-(d,α) GPBiCG
BILU(1+,120°)-(d,α)
BILU(1+, 60°)-(d,α)
BILU(1+, 30°)-(d,α)
BILU(2)-(d,α)







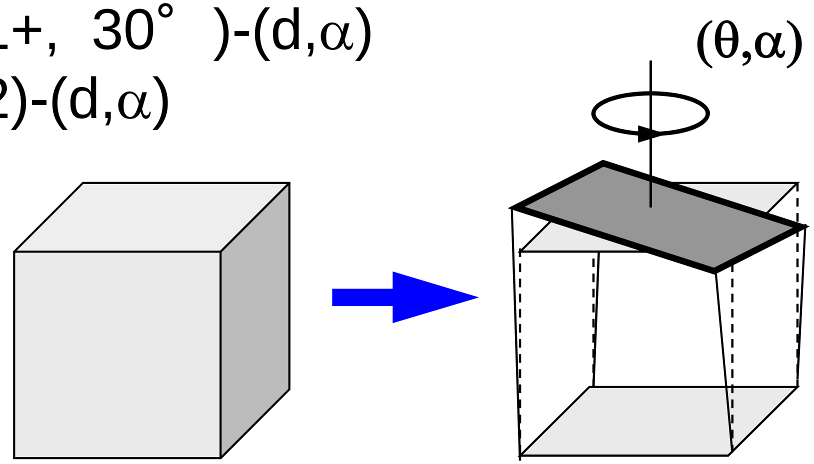
Results: 64 cores Distorted Meshes

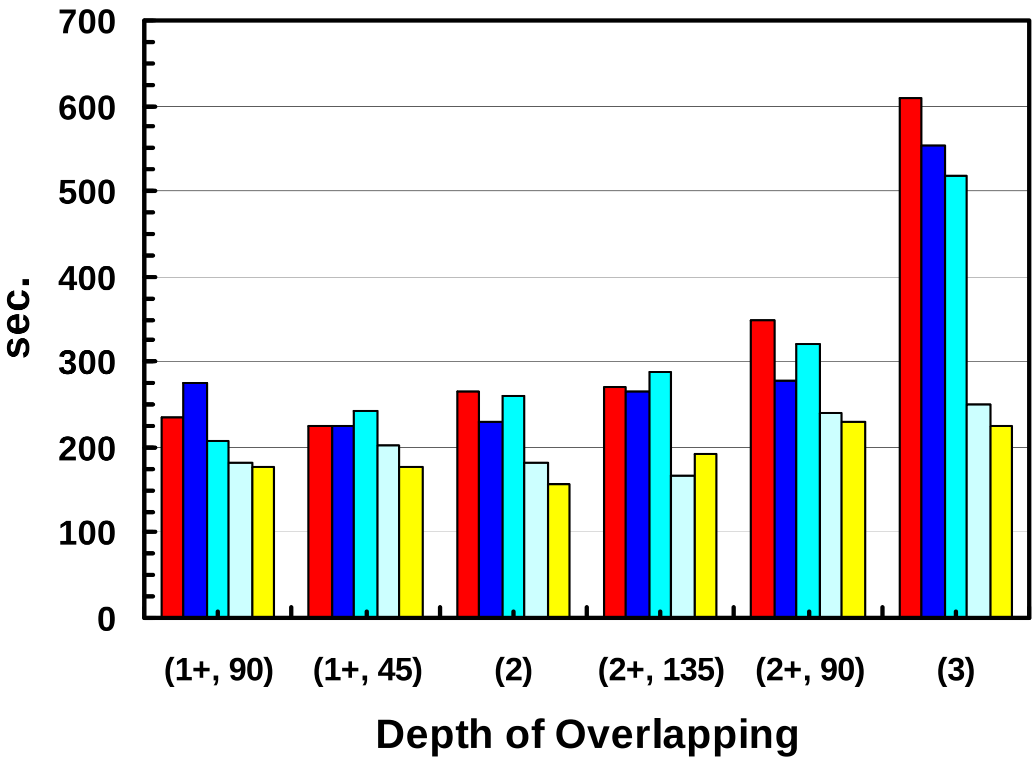
BILU(p,θ)-(d,α)

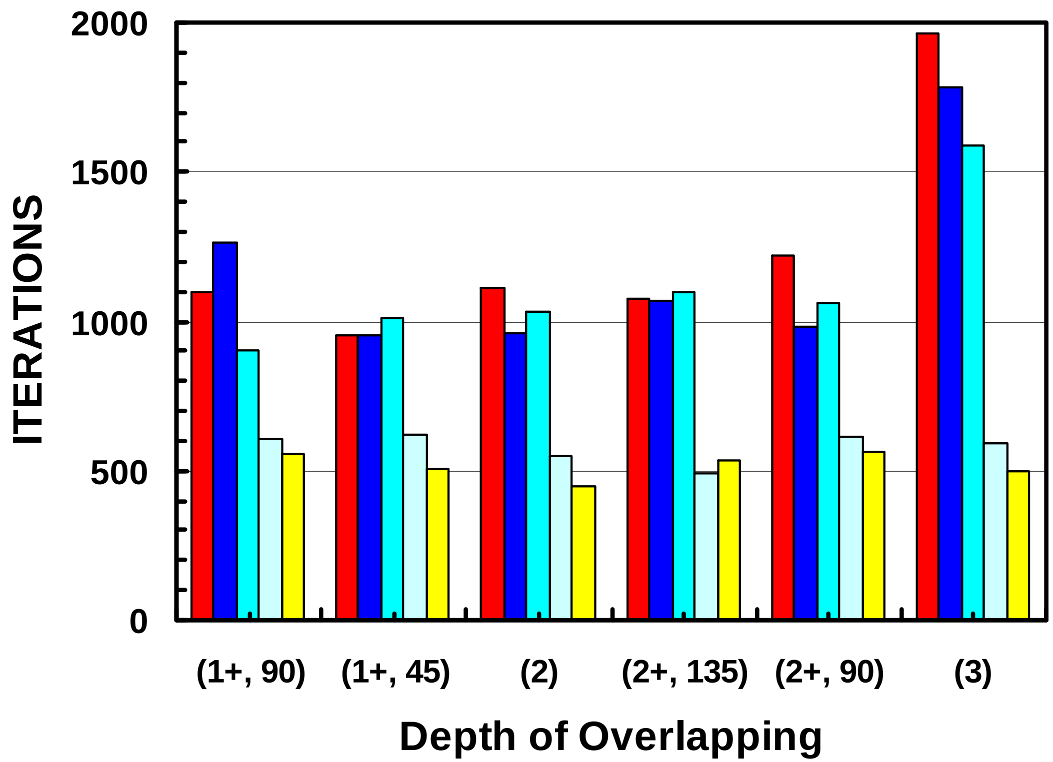
3,090,903 DOF

MAX distortion: **225-deg.**

- BILU(1)-(d,α) GPBiCG
- BILU(1+,120°)-(d,α)
- BILU(1+, 60°)-(d,α)
- BILU(1+, 30°)-(d,α)
- BILU(2)-(d,α)







Selective Blocking/Overlapping does not work well in this case !

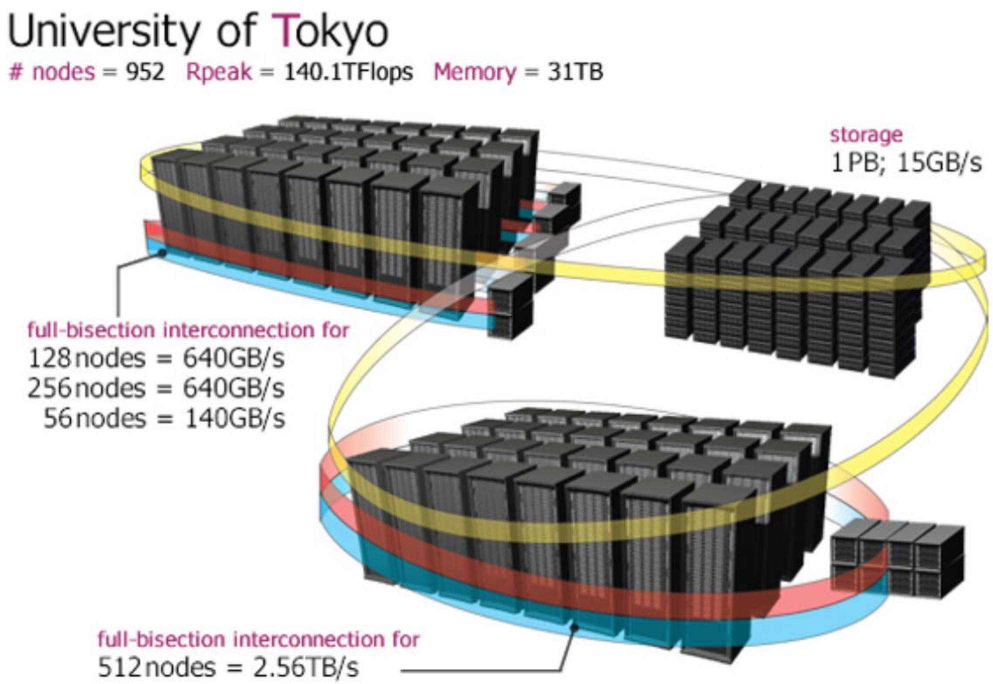
- 150 deg.: BILU(1)-(1)
- 225 deg.: BILU(2)-(2)

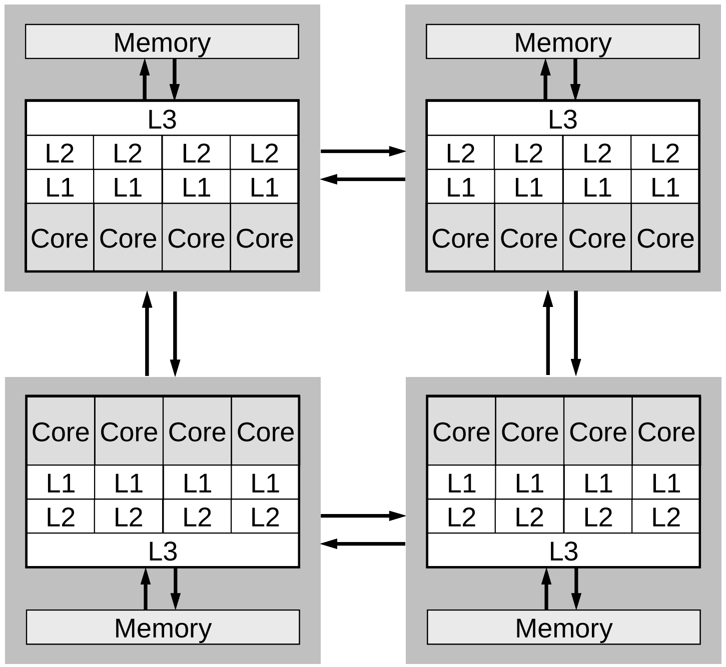
Software, Linear Solvers

- MPI + FORTRAN90 (Hitachi Compiler)
  - Flat MPI
- NUMA control: Optimum case
  - `numactl --cpunodebind=$SOC --membind=$SOC`
- Finite-Element Method
  - Tri-linear hexahedral elements
- Linear Solver
  - GPBiCG [Zhang 1997]
- Preconditioners
  - Block ILU(2,t): 2nd order of fill-ins, Threshold parameter
  - keep $\mathbf{m}_{ij}$ component of preconditioner **[M]** if $\mathbf{m}_{ji} > t$
    - t=0: Original BILU(2)
  - Optimum value of “t” @512 cores= 0.02~0.03

Hardware Environment

- “T2K Open Super Computer (Tokyo)”
  - T2K/Tokyo
  - Total 952 nodes (15,232 cores)
    - each node = 4x AMD Quadcore Opteron Socket (Barcelona)
  - 45th in TOP500 (NOV. 2009)
- up to 32 nodes (512 cores) in this work



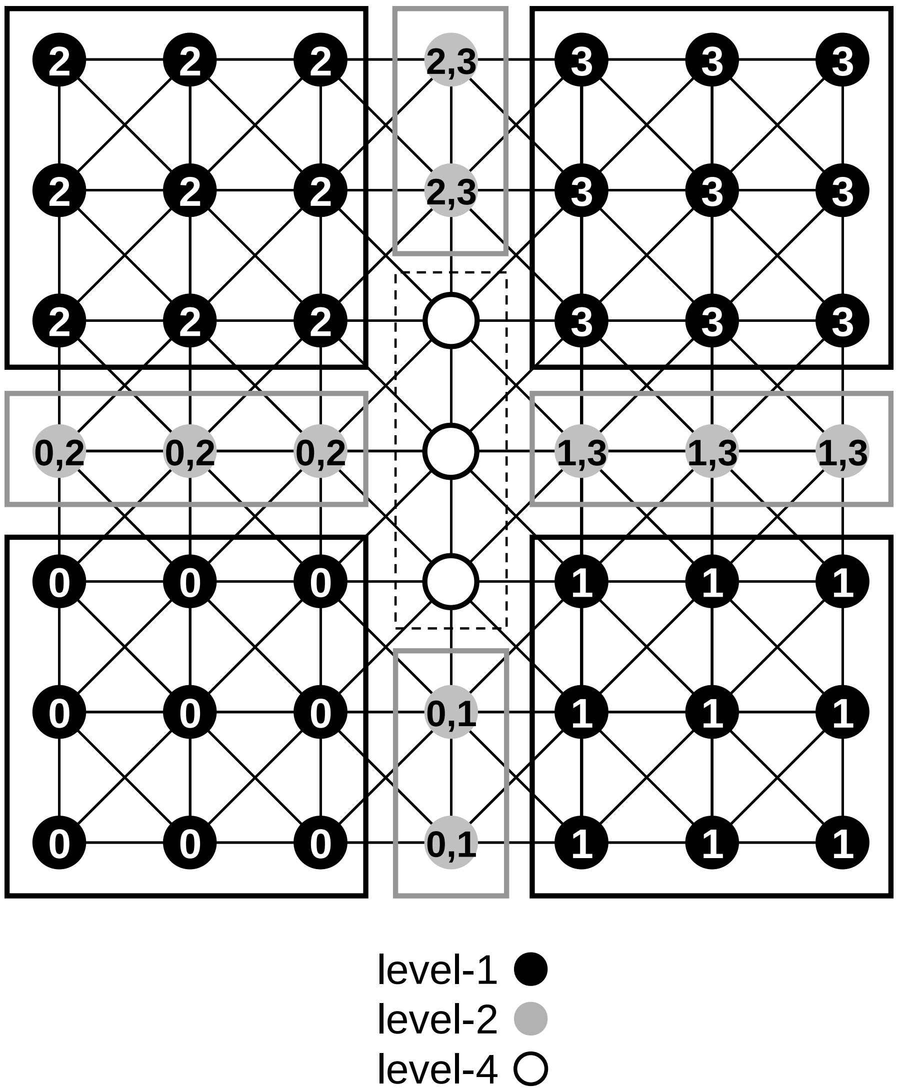


Strategies for Domain Decomposition

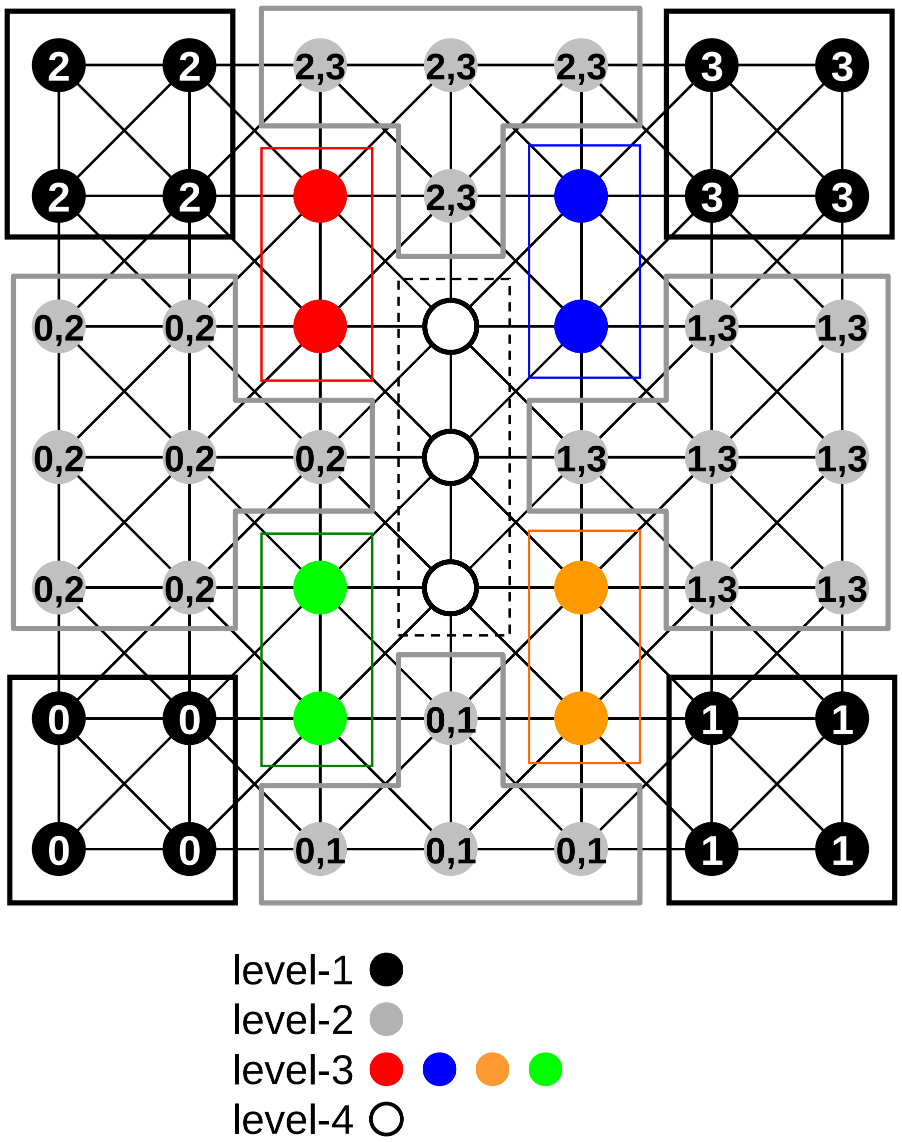
- BILU (2,t, loc-**d**)
  - Localized Block Jacobi with extended overlapping
  - **d**: Depth of overlapping
    - BILU(2,t,loc-1), BILU(2,t,loc-2), BILU(2,t,loc-3)
- BILU (2,t, org-**d**)
  - Original HID (HID-org) with extended overlapping
    - BILU(2,t,org-1), BILU(2,t,org-2)
- BILU (2,t, new-**d**)
  - HID with extended overlapping/thicker separators: HID-new
    - BILU(2,t,new-1), BILU(2,t,new-2)
    - 3 layers for level-2 separators
    - NO special treatment for load-balancing

Strategies for Domain Decomposition

Original HID

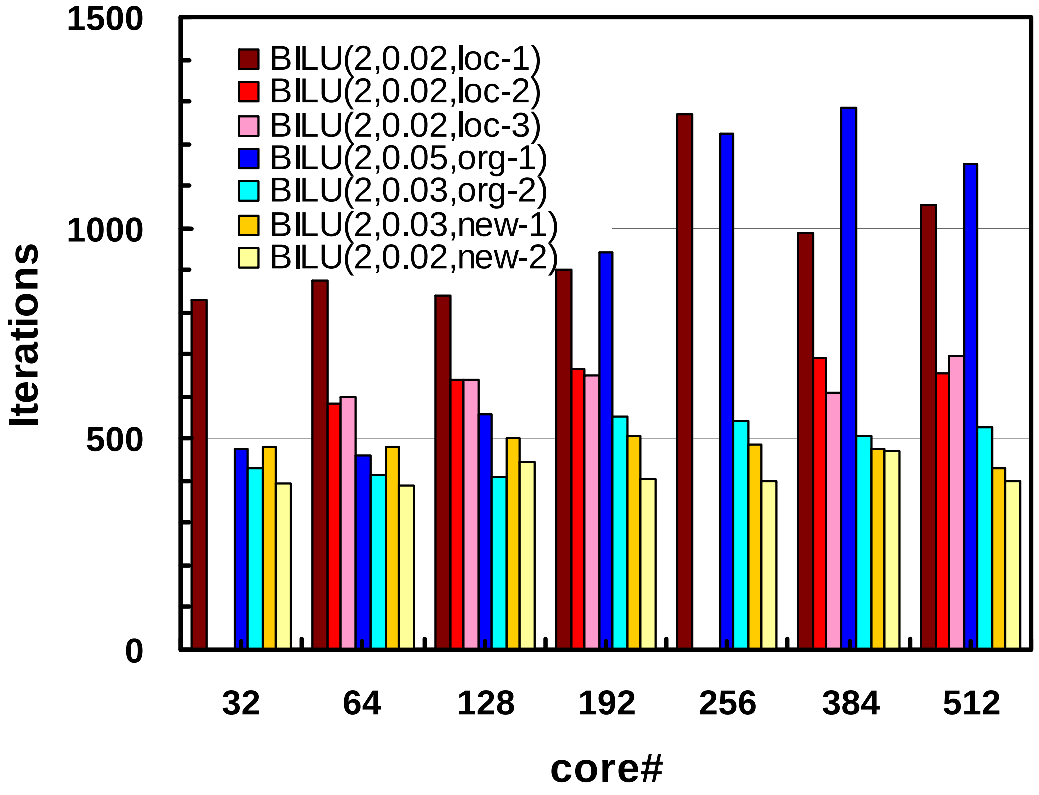


HID-new

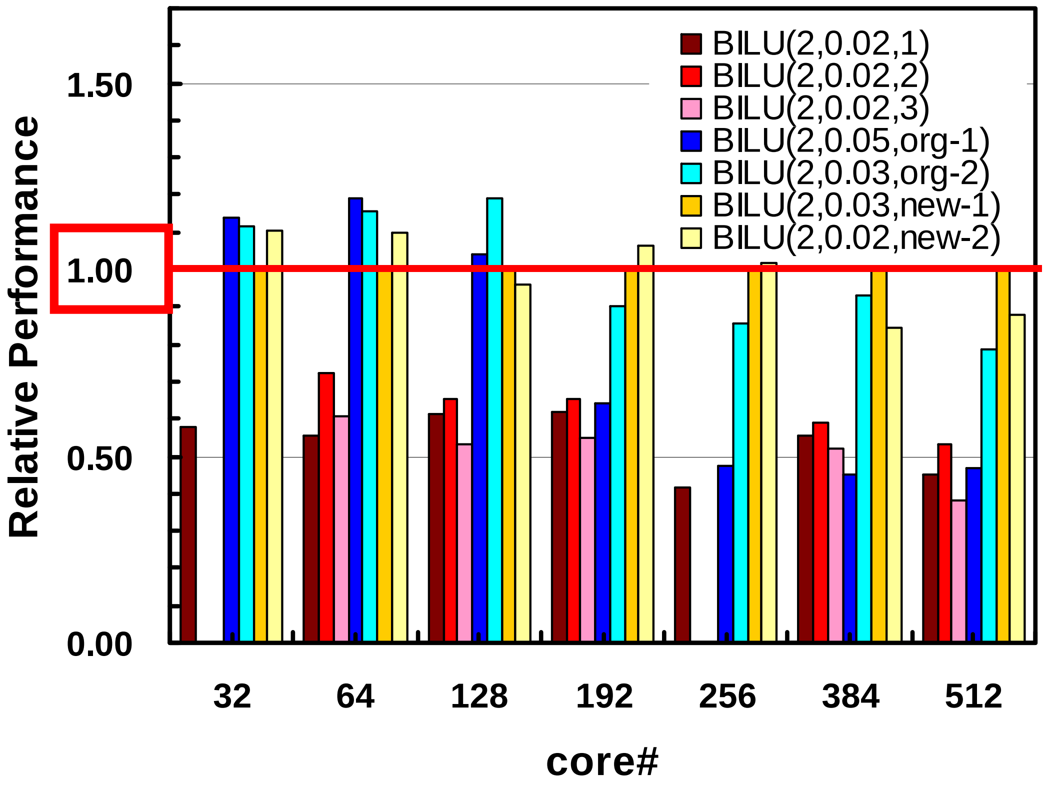


Strong Scaling, 128^3 elements MAX: 200 deg., Scalability

Iterations



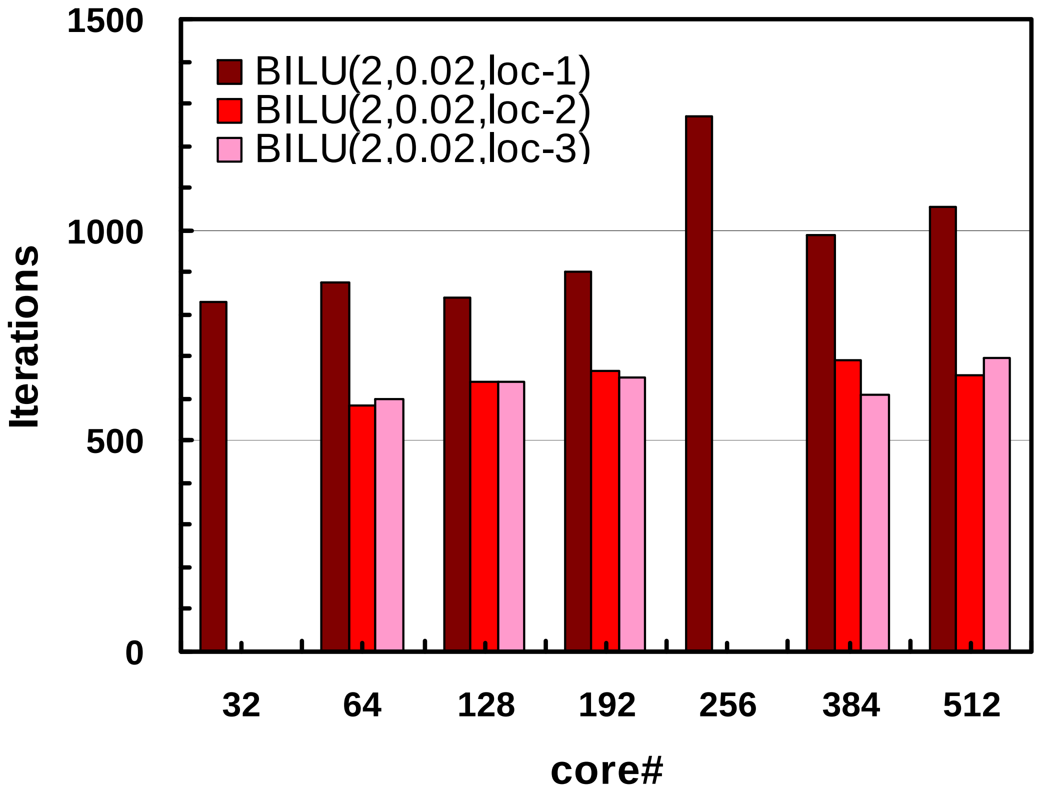
Relative Performance



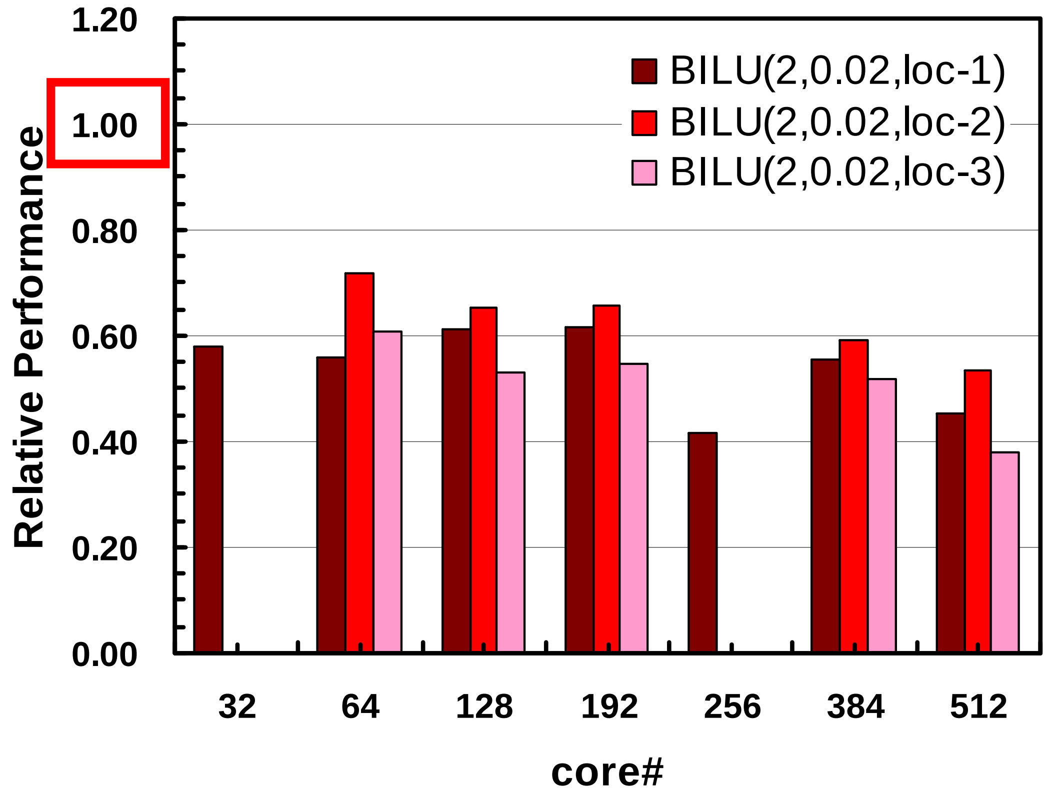
- Normalized by performance of BILU(2,0.03,new-1) at each core

Localized Block Jacobi BILU(2,t,loc-d): not robust

Iterations



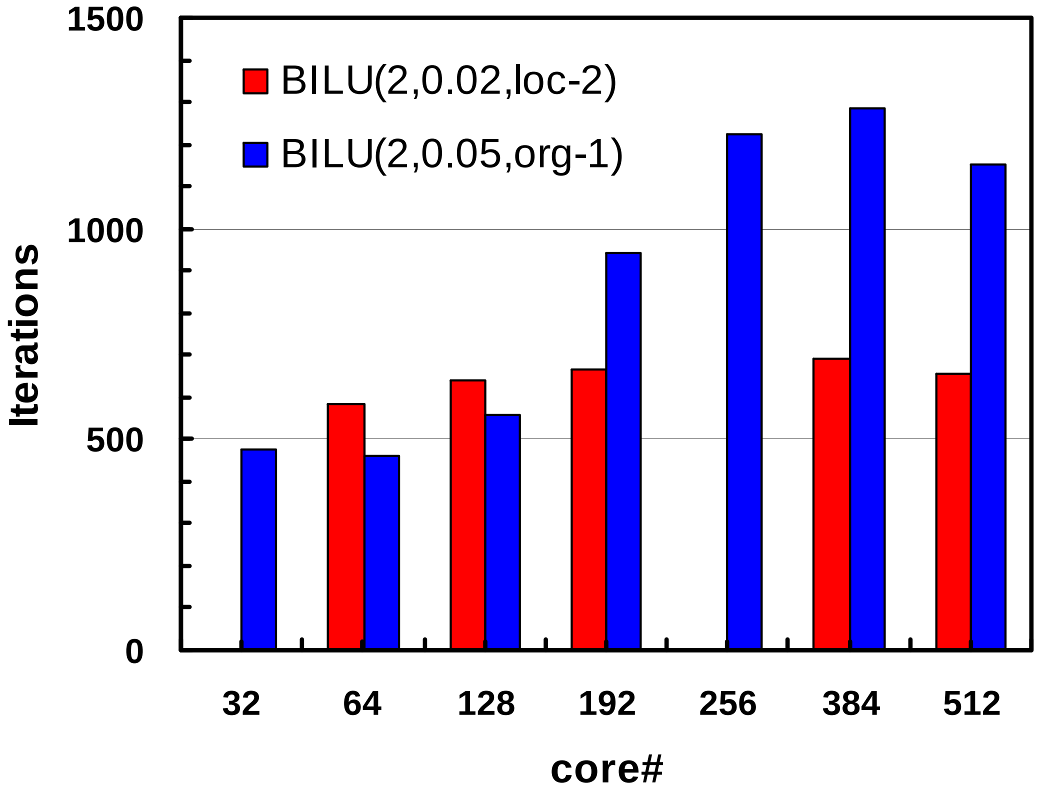
Relative Performance



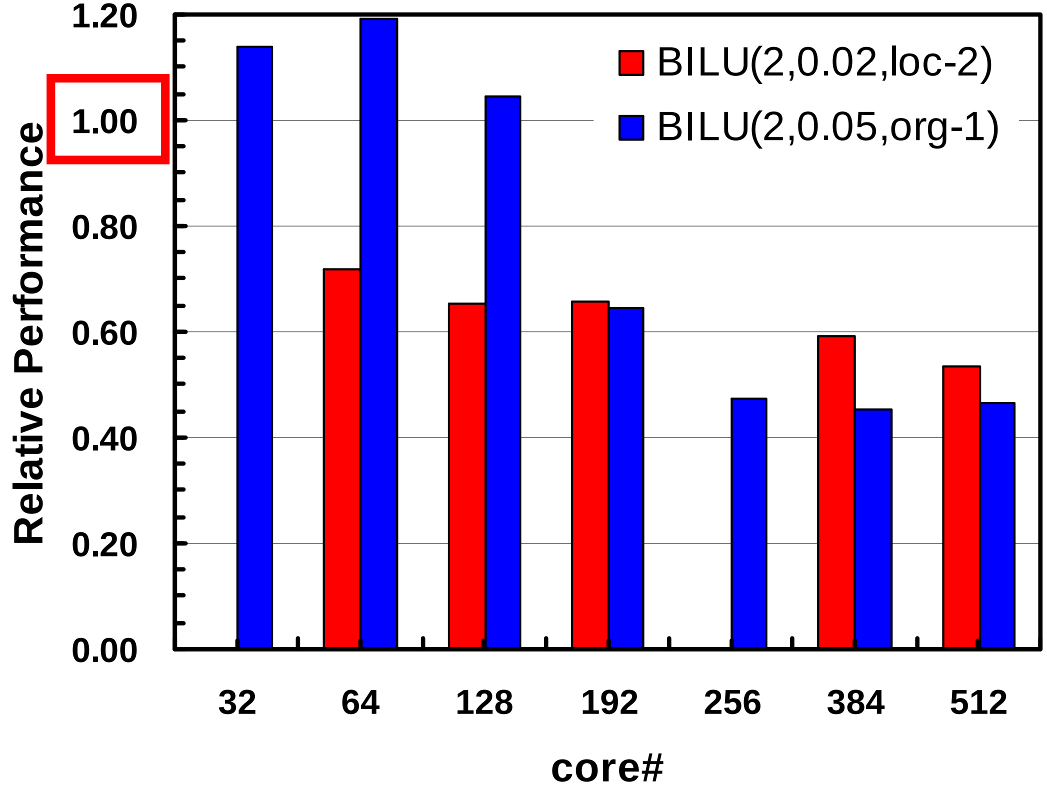
- BILU(2,t,loc-2) is the best
  - although BILU(2,t,loc-d)'s do not converge in some cases.
- Performance is generally worse than BILU(2,t,new-1) with HID

Orig. HID with Extended Overlapping BILU(2,t,org-1)

Iterations



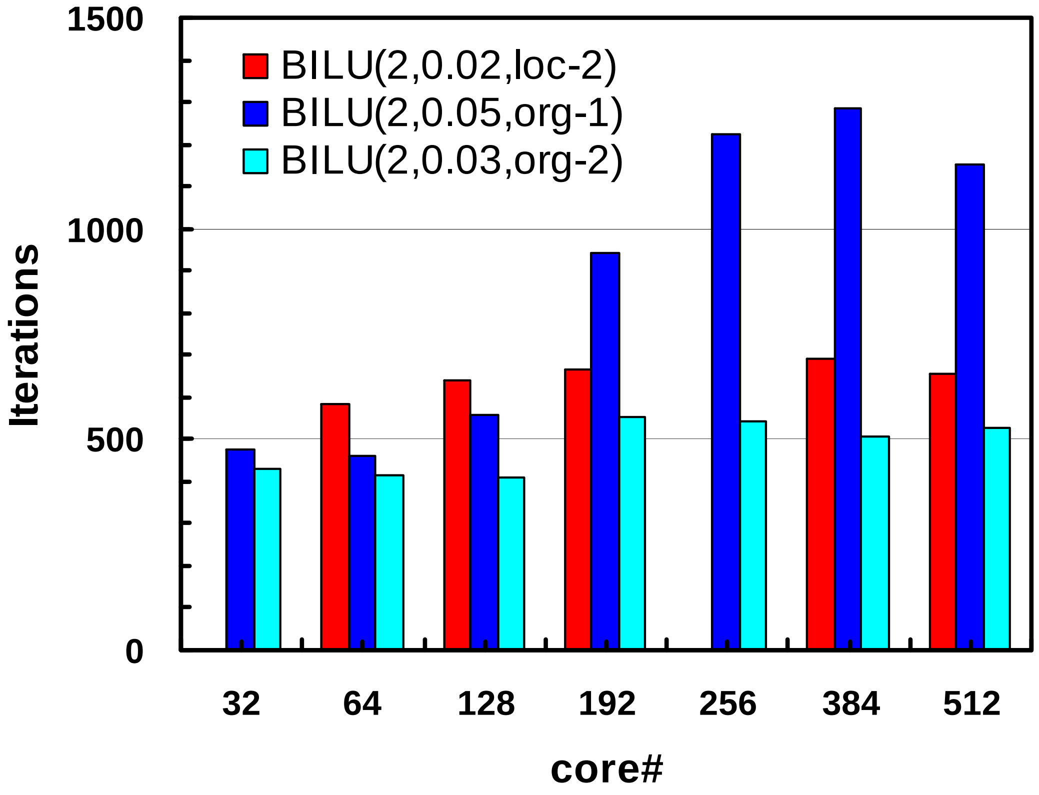
Relative Performance



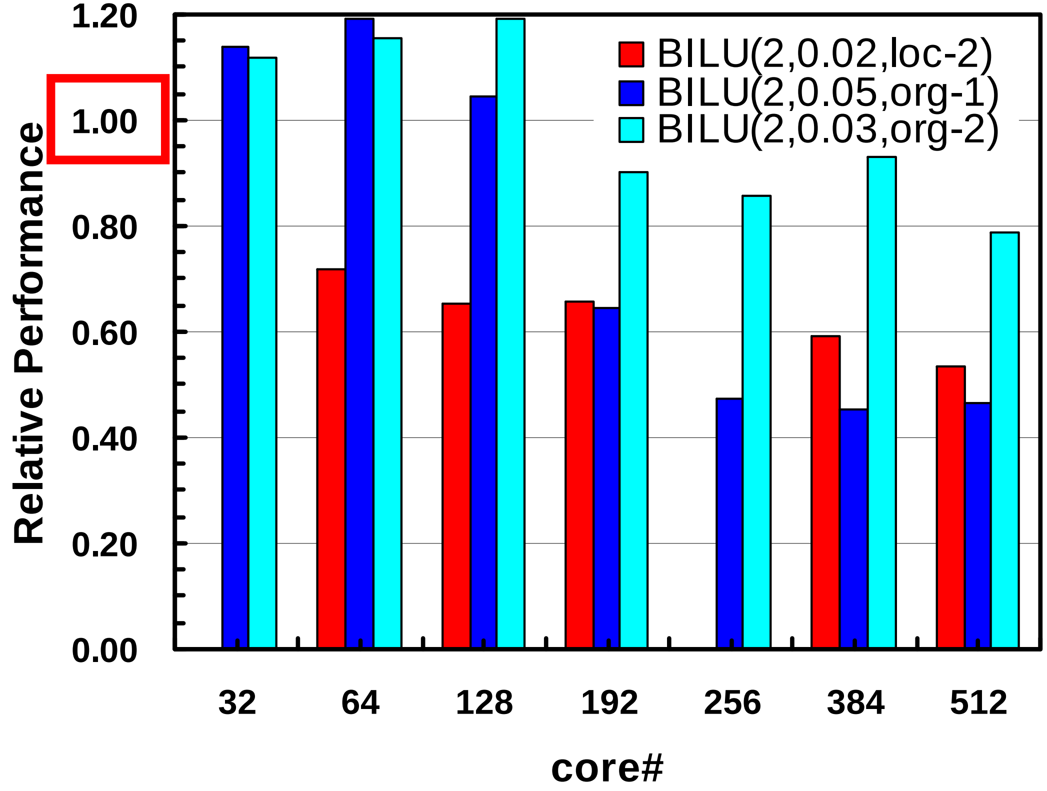
- BILU(2,t,org-1) gets unstable, as core number increase (>128).

Orig. HID with Extended Overlapping BILU(2,t,org-d)

Iterations



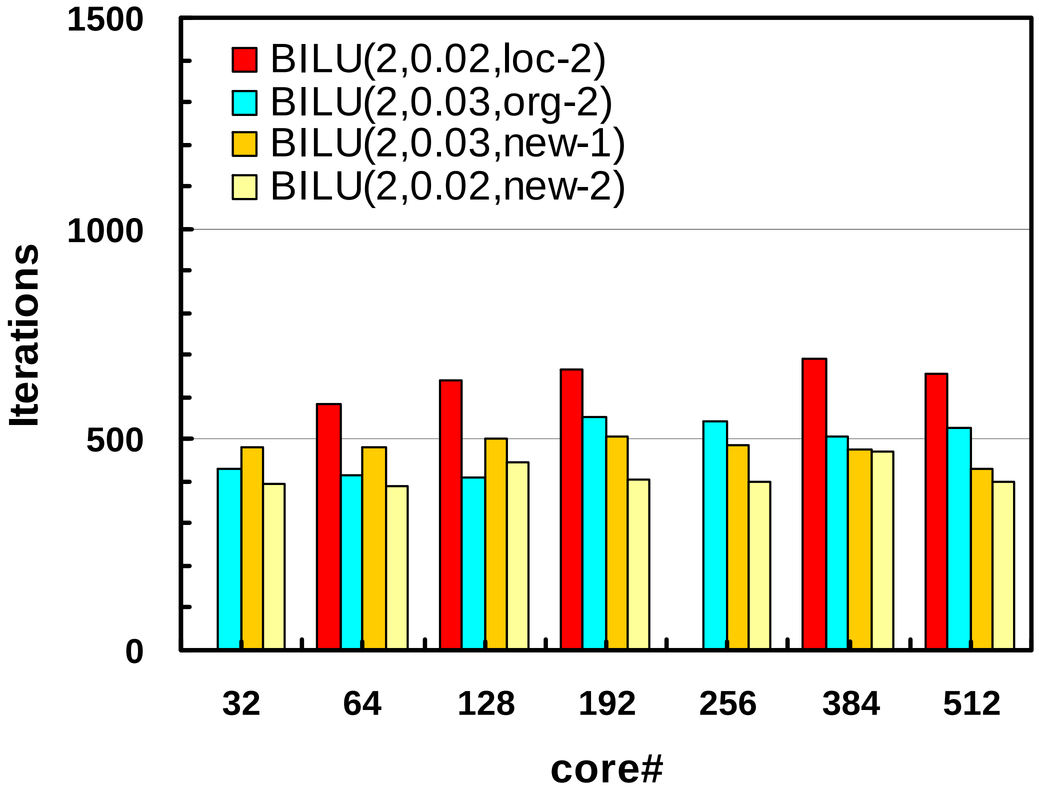
Relative Performance



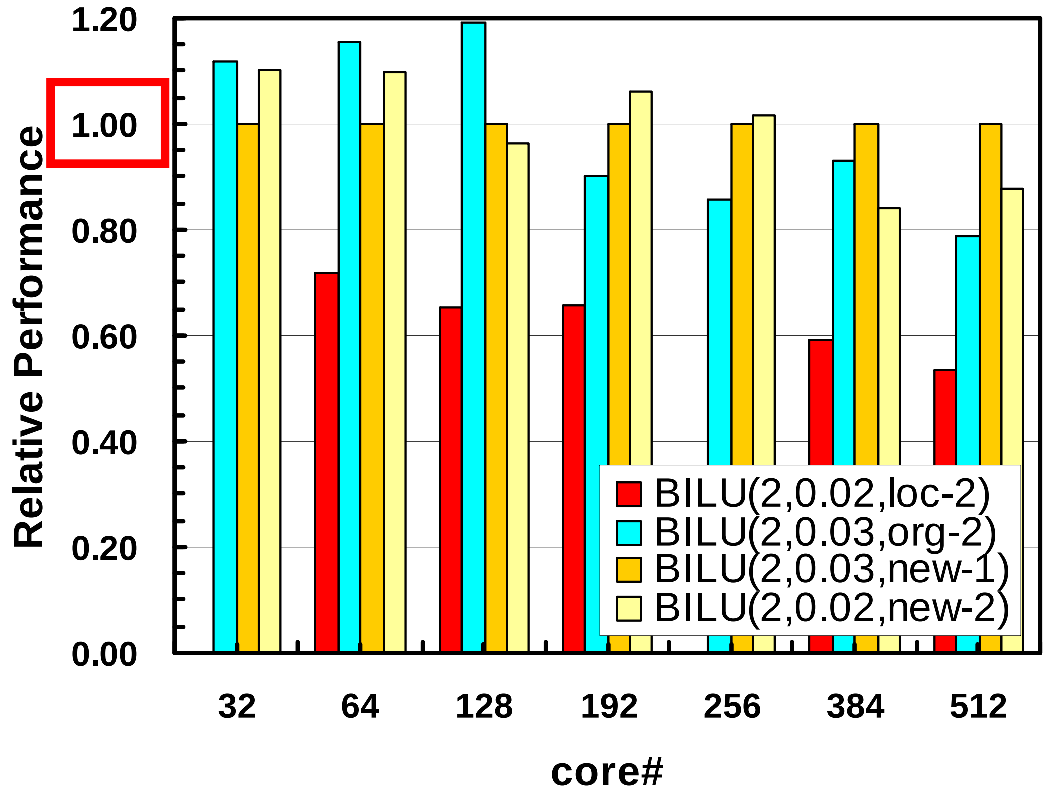
- Extended overlapping provides robustness: BILU(2,t,org-d)

New HID with Extended Overlapping BILU(2,t,new-d)

Iterations



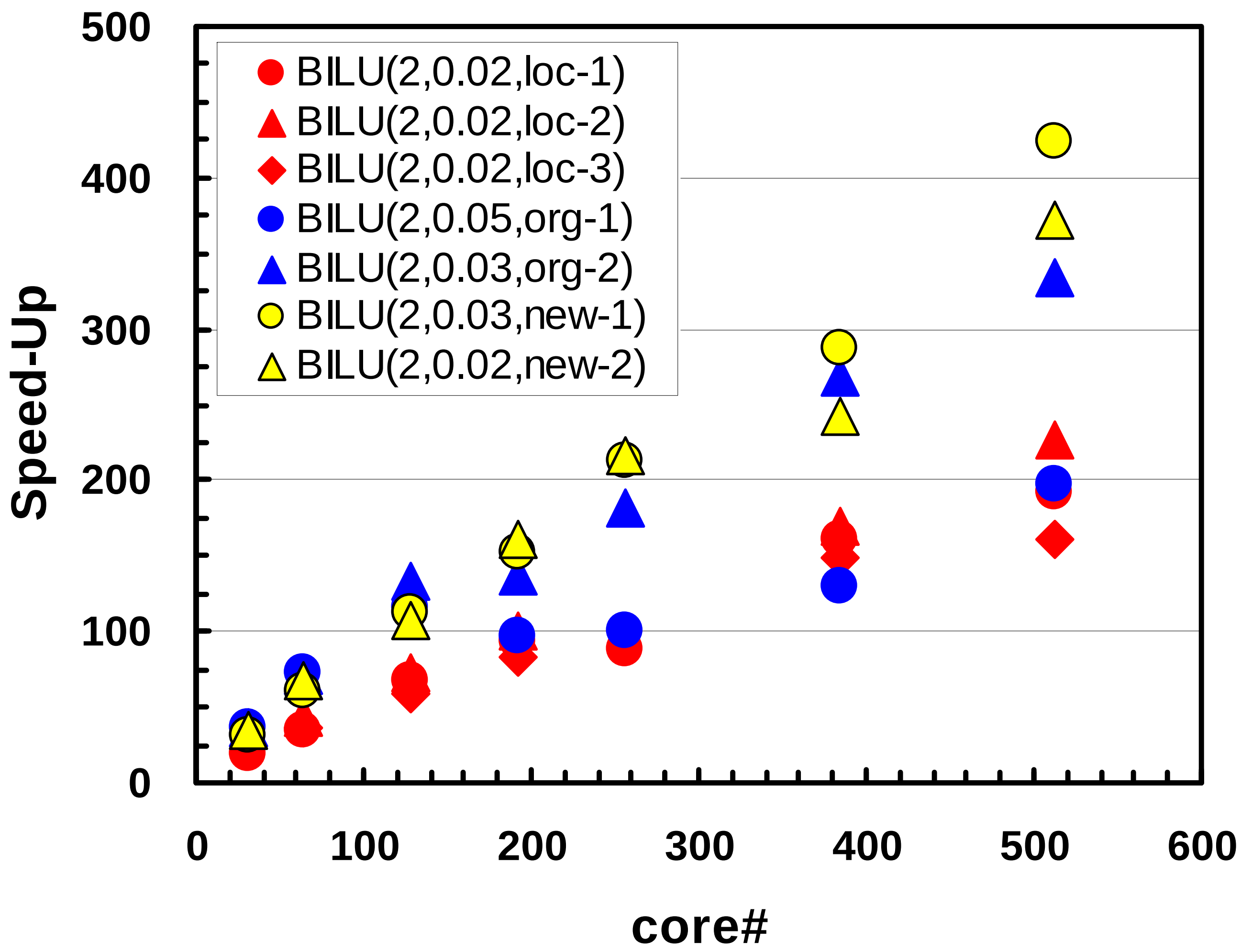
Relative Performance



- BILU(2,t,new-d)’s generally more robust and efficient, if number of cores is larger (BILU(2,t,org-d)’s are better, if core# is smaller).

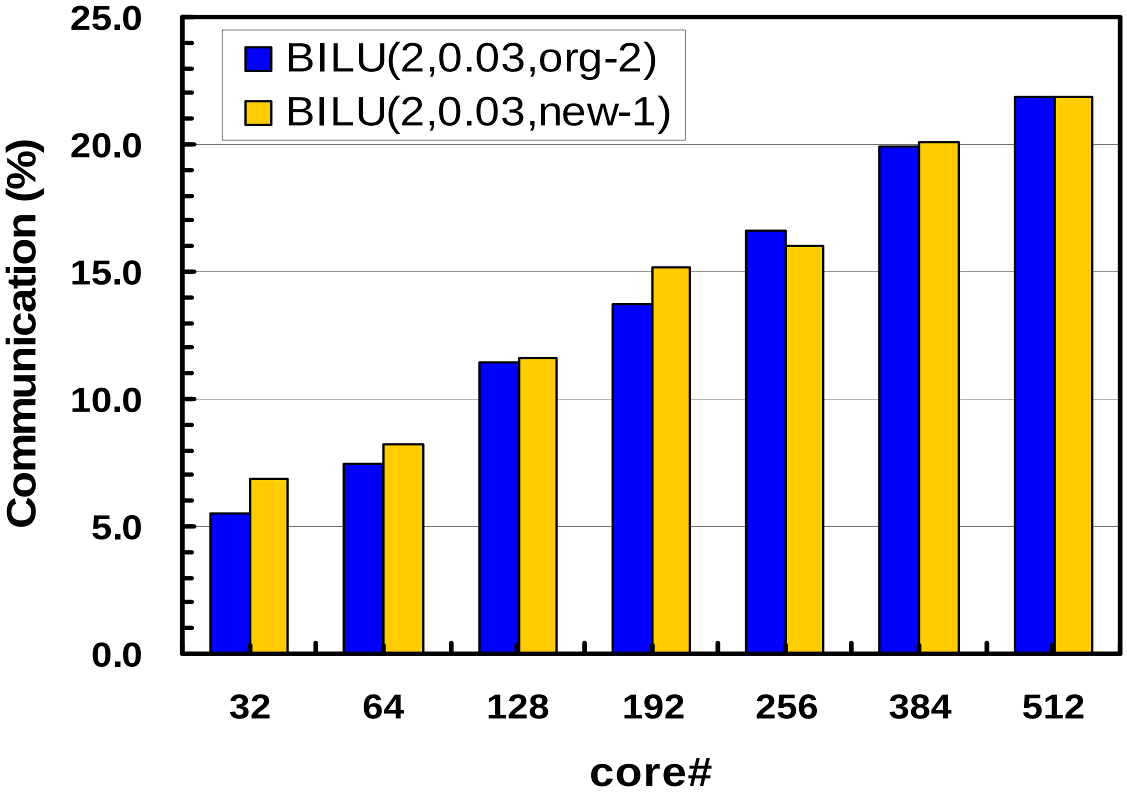
Strong Scalability: 32~512 cores

Performance of BILU(2,0.03,new-1) with 32 cores= 32.0

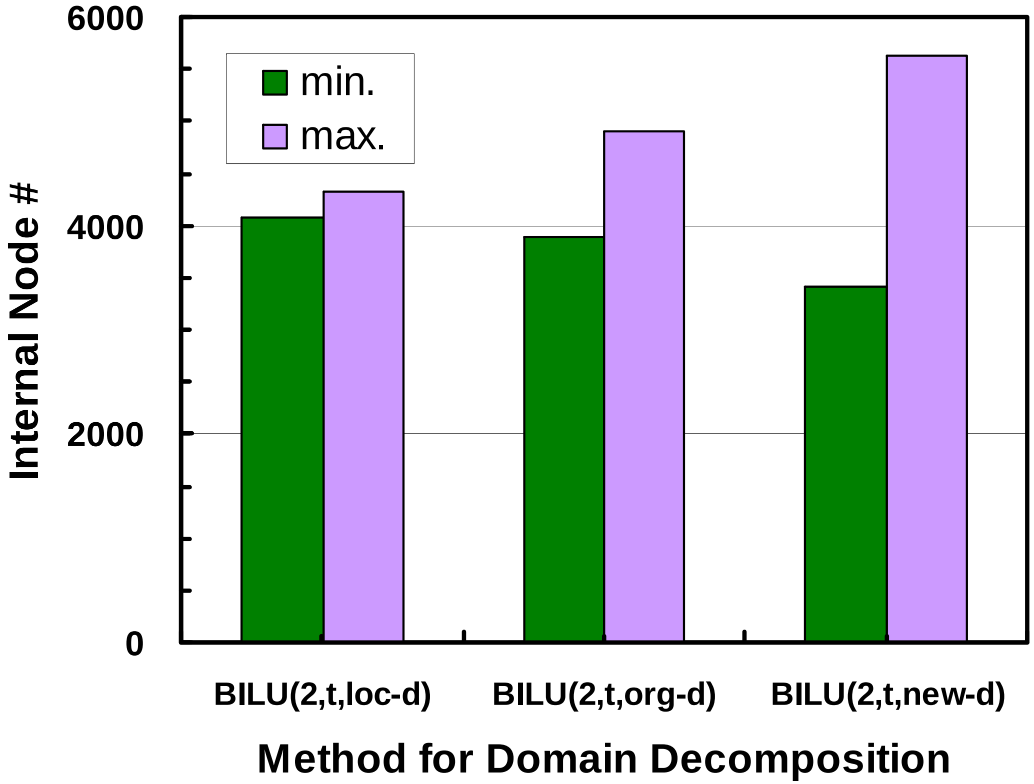


Bottlenecks for Scalability

- Additional Communications in HID-org/HID-new
  - rate for entire solver time



- Load Imbalance (512 cores)
  - Standard Deviation (σ)
    - BILU(2,t,loc-**d**) 85
    - BILU(2,t,org-**d**) 155
    - BILU(2,t,new-**d**) 289



Summary

- Extended version of HID
  - Extension of overlapped elements between domains
  - *Thicker* separators
- *Extended HID* provides more robust and scalable performance than original HID and localized block Jacobi BILU
  - Effect of *thicker separator* is very significant if the number of core is larger.
    - more effective than deeper overlapping
  - *Extended HID* with thicker separator can introduce effect of external nodes efficiently in factorization and forward/backward substitution processes with higher order of fill-ins.

Future Works

- Evaluation of feasibility for various types of applications of:
  - Localized Block Jacobi with Extended Overlapping
    - also selective fill-ins, selective overlapping
  - Original HID, New HID
- Development of sophisticated domain partitioner for complicated geometries
  - key technology for practical application of extended HID to real applications.
  - *Thickening* of separator layers should be considered at every level for robust convergence.
    - Only at level-2 layers in the present work
  - Load-balancing for *extend HID*
    - another big technical issue to be solved in the future.