Pattern Based Method For P/G Grid Analysis

Jin Shi, Yici Cai, Xianlong Hong
Sheldon X.-D. Tan

EDA Lab. CS Department, Tsinghua University

EE Department, University of California Riverside

Outline

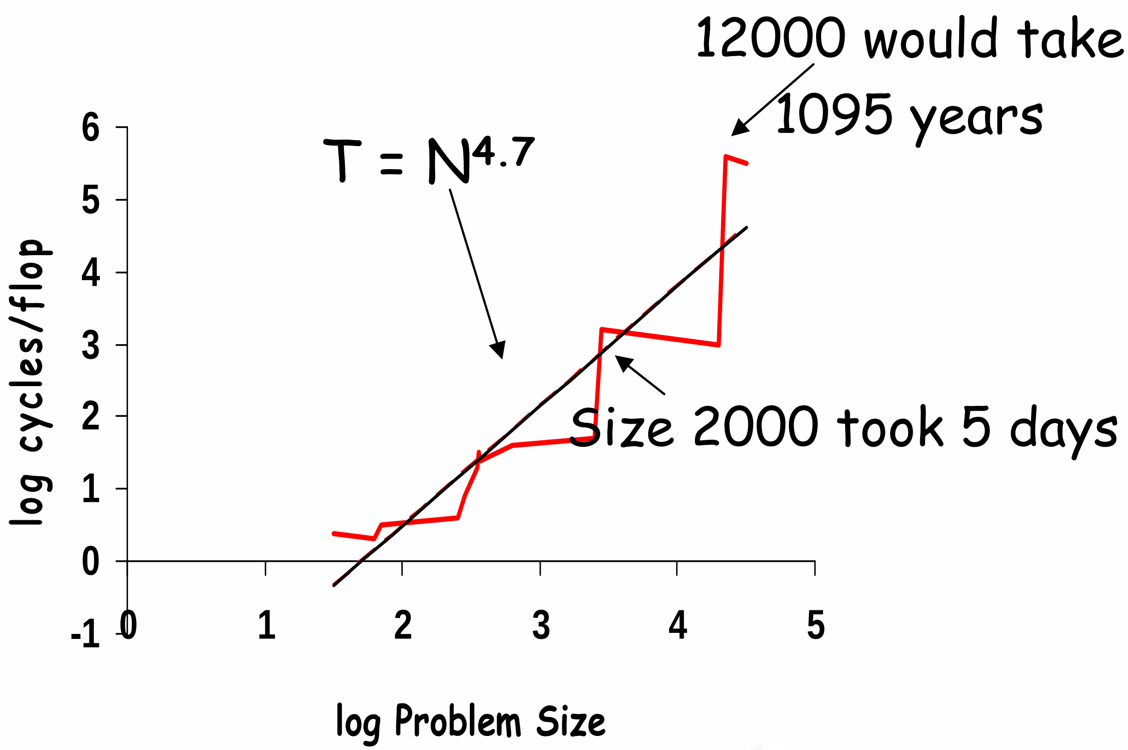
- Practical Computation Problems
- Overview of Existing Methods
- Acceleration Tech
- Some Observations
- Pattern Based Method
- Conclusion

Outline

- *Practical Computation Problems*
- Overview of Existing Methods
- Acceleration Tech
- Some Observations
- Pattern Based Method
- Conclusion

Practical Problems #1

- Linear computation complexity does not mean linear increasing time cost due to Cache Miss



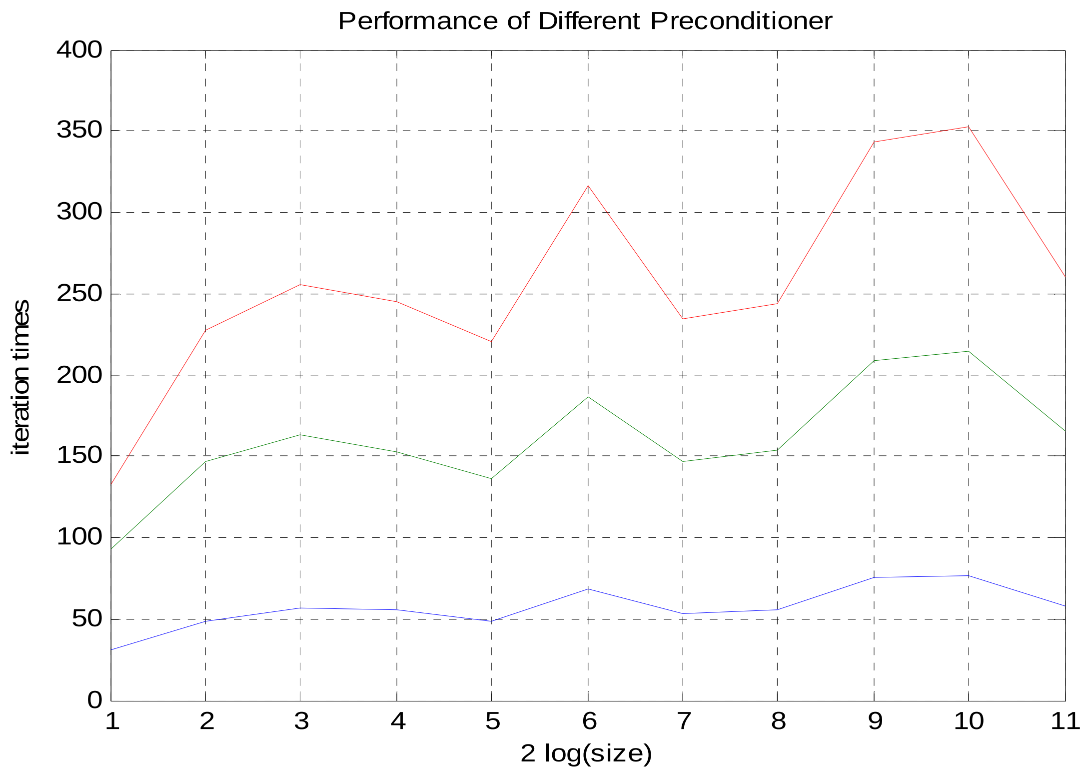
matrix multiply operation $O(n^3)$ algorithm looks like $O(n^5)$

Practical Problems #1

- Summary of Cache Miss
  - Linear computation complexity is not enough
  - Optimize algorithms together with cache performance

Practical Problems #2

- Iterative efficiency
  - Preconditioner's performance decreases as the matrix size increases



Practical Problems #3

- Memory efficiency limitation
- When the Design is too large …
  - Hard to load it from DB
  - Impossible to build matrix
  - Vector malloc run out of memory
  - Too many years to get results

Our Contribution

- Alleviate Cache Miss
- Reduce the overall memory usage
- Present more efficient preconditioner under a partition framework
- Constant preconditioner performance

Outline

- Practical Computation Problems
- *Overview of Existing Methods*
- Acceleration Tech
- Some Observations
- Pattern Based Method
- Conclusion

Summary of Existing Method

- Computation Complexity < $O(n^2)$
  - LU $O(n^2)$
  - PCG $O(n^p)$ p slightly larger than 1
  - M-G $O(n)$ with small coefficient
  - R-W $O(n)$ with large coefficient
  - ADI $O(n)$ with small coefficient
- Memory Efficiency
  - LU $O(n^2)$
  - PCG $O(n)$
  - M-G $O(n)$
  - R-W $O(n)$
  - ADI $O(n)$
- Trade off between speed and accuracy

Summary

- Direct Methods vs Iterative Methods
  - Time cost of LU:

$$t(d(A)) + N \times (\, t(L^{-1}v_1) + t(U^{-1}v_2)\,)$$

$$t(L^{-1}v_1) \propto (1+f)\cdot nnz(A) \qquad f: fill\ in\ ratio$$

  - Time cost of PCG:

$$t(d(A)) + N \times (\, n \cdot t(Av)\,)$$

$$t(Av) \propto nnz(A)$$

  - Which is faster depends on the fill in ratio and performance of preconditioner

Outline

- Practical Computation Problems
- Overview of Existing Methods
- *Acceleration Tech*
- Some Observations
- Pattern Based Method
- Conclusion

Acceleration Tech

- Model Order Reduction
  - S domain Based   before 2000
  - Electrical Equivalent Circuit Based   2002
  - Topological Partition Based   2004
- Among these methods, topological partition method is the most powerful one to simulate large P/G grid

Partition Benefits

- For direct method
  - Decrease decomposition time

    $$n \cdot x^2 < (nx)^2$$

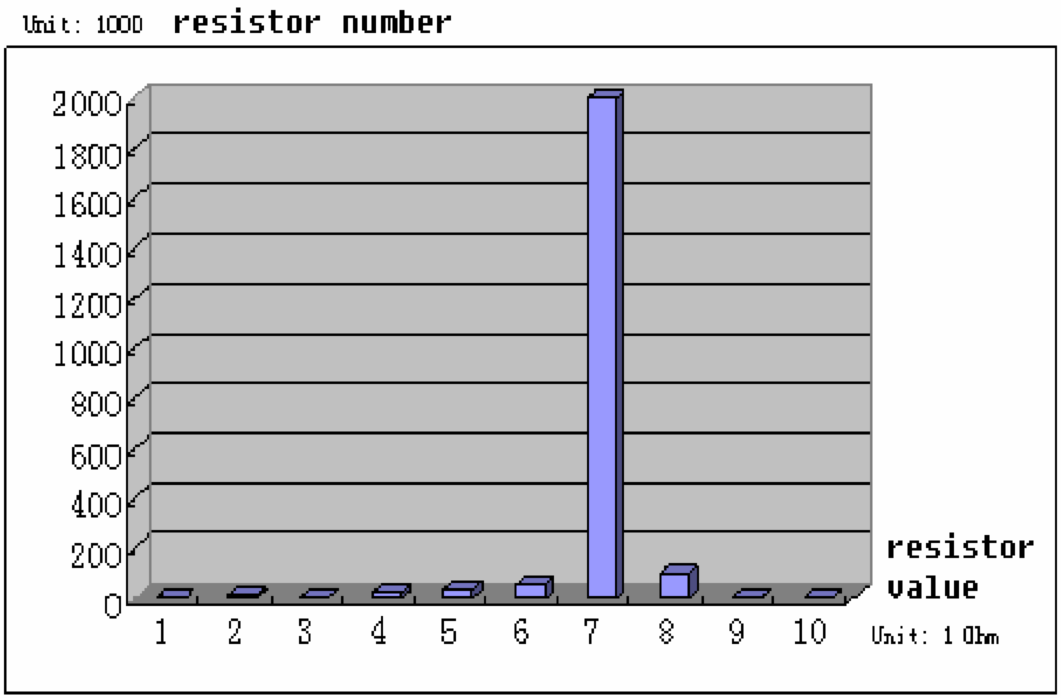
  - Decrease fill in ratio
  - Decrease Cache Miss
- For iterative method
  - Decrease preconditioner construction time
  - Increase preconditioner performance
  - Decrease iteration time
  - Decrease Cache Miss

Outline

- Practical Computation Problems
- Overview of Existing Methods
- Acceleration Tech
- *Some Observations*
- Pattern Based Method
- Conclusion

Observation #1

- Too many elements share the same value
  - Extract R L C with BEM Solver
  - All most all elements are extracted from M1 and M2
  - More than 80% elements in M1 and M2 have the same value



Observation #2

- P/G grid topology is self-similar
  - One layer contains many routed metal in the same direction
  - Metal rails share the same width and pitch
- Possible to transform topology similarity to matrix similarity ?
  - Yes
- How about irregular P/G grid ?
  - Do Local regularization to make elements in local area share the same value
- Continuous in topology domain
  - Local regularization will not introduce obvious computation errors

Observation #3

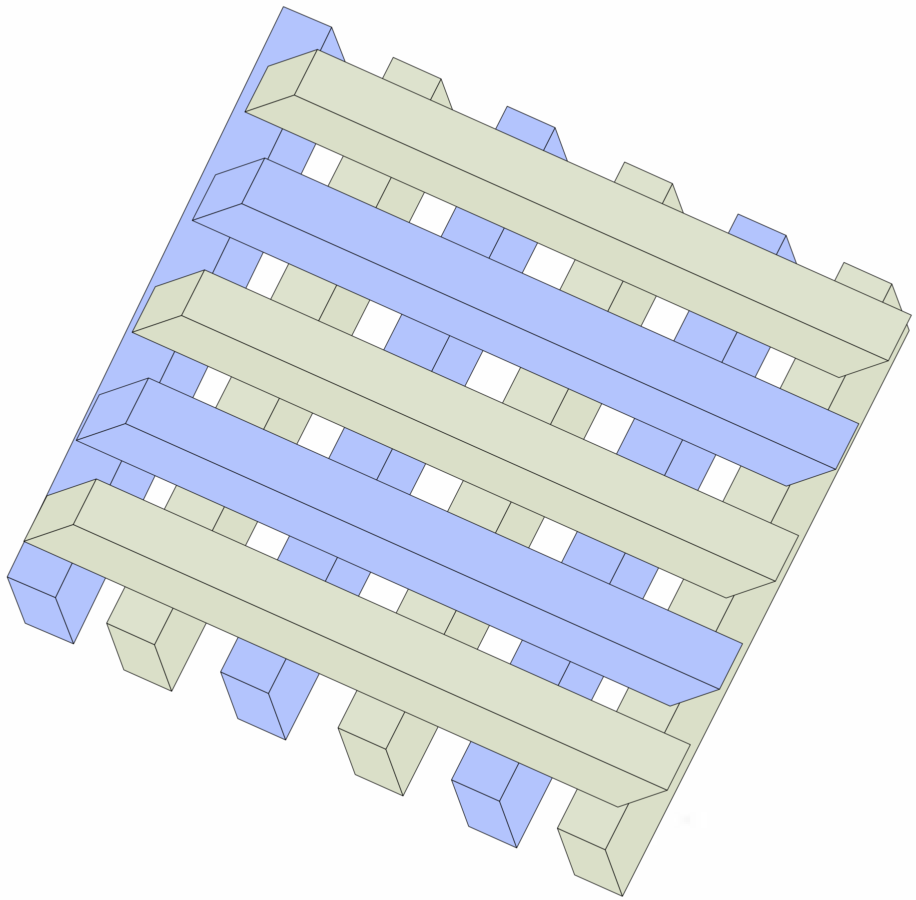
- When matrix size is medium, PCG method is faster than any other method
  - PCG can converge within 10 times iteration
  - LU usually has fill in factor larger than 30
  - R-W is inefficient for small case
  - M-G needs auxiliary computation structure and time to construct them
- Traditional preconditioner can be improved
  - Use topology similarity property
  - Use element value similarity property

Outline

- Practical Computation Problems
- Overview of Existing Methods
- Acceleration Tech
- Some Observations
- *Pattern Based Method*
- Conclusion

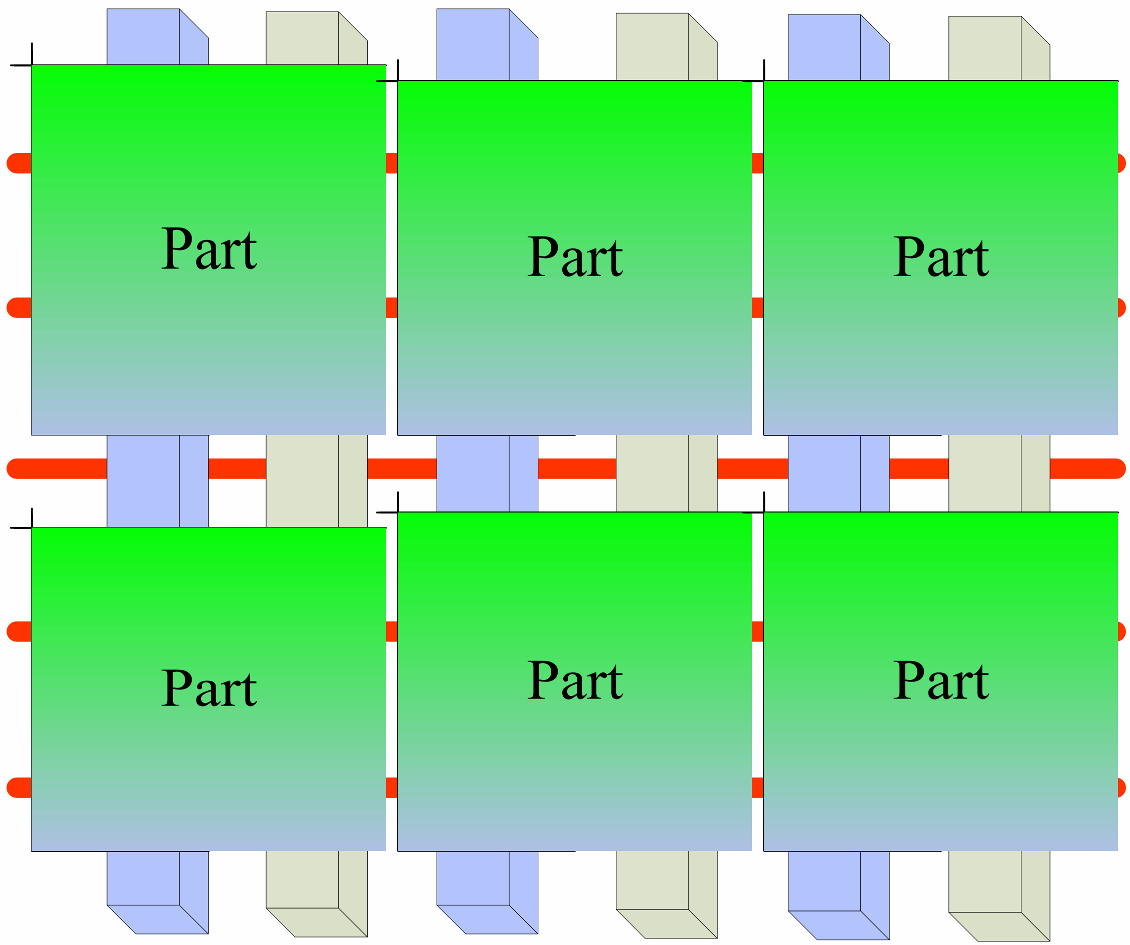
Pattern Based Method

- Regular P/G Grid in 3D



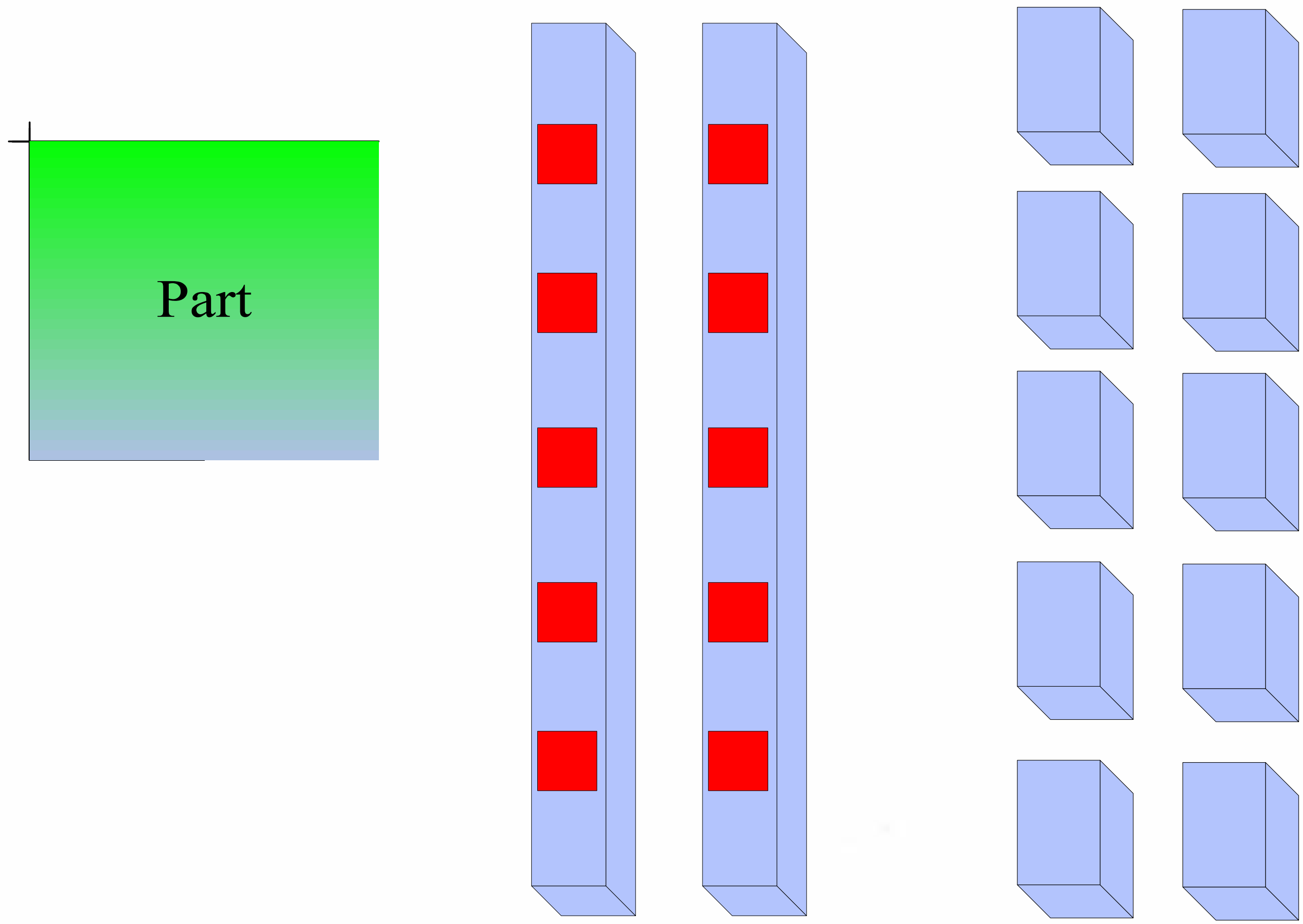
Pattern Base Method

- Self Similarity: Global Similarity (Global Pattern)



Pattern Base Method

- Self Similarity: Local Similarity (Local Pattern)

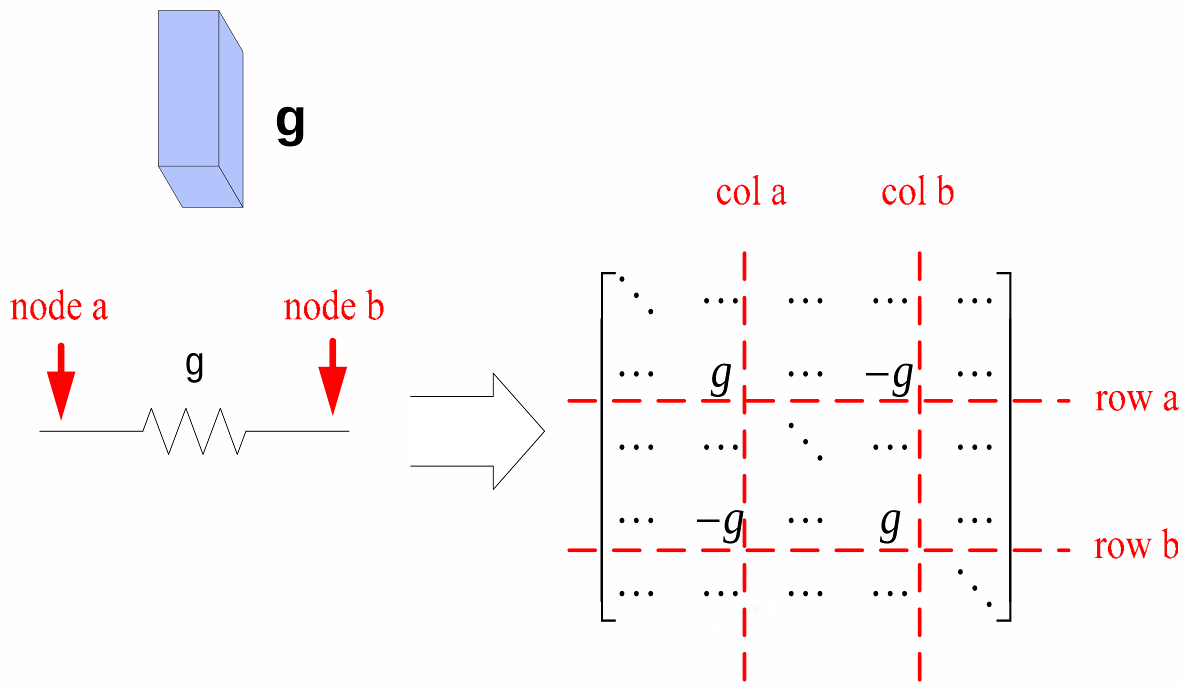


Pattern Base Method

- Similarity Summary
  - Patterns are elements similar to each other
  - Patterns exist not only in global area but also in local area
- Strategy
  - Partition global area to blocks, perform relaxation iteration between global blocks
  - Reuse local pattern to perform local simulation

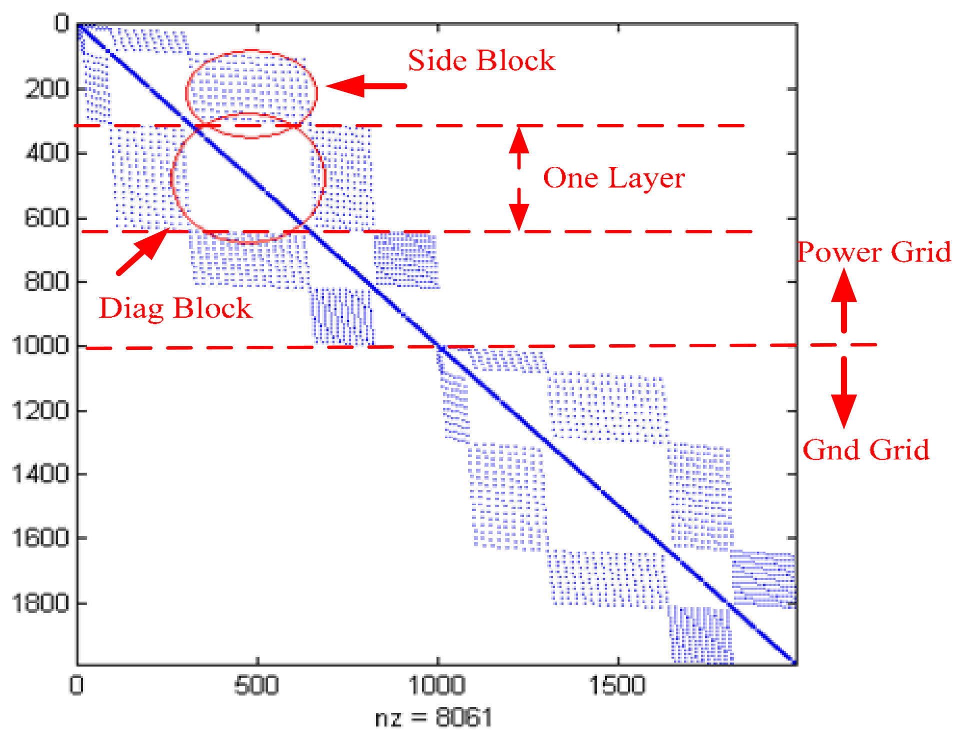
Pattern Based Method

- Local Matrix Generation
  - Element fill in under NA method
  - Node order is important



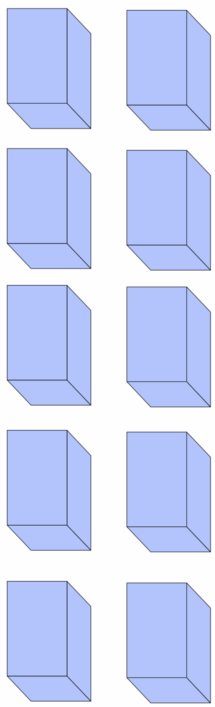
Pattern Based Method

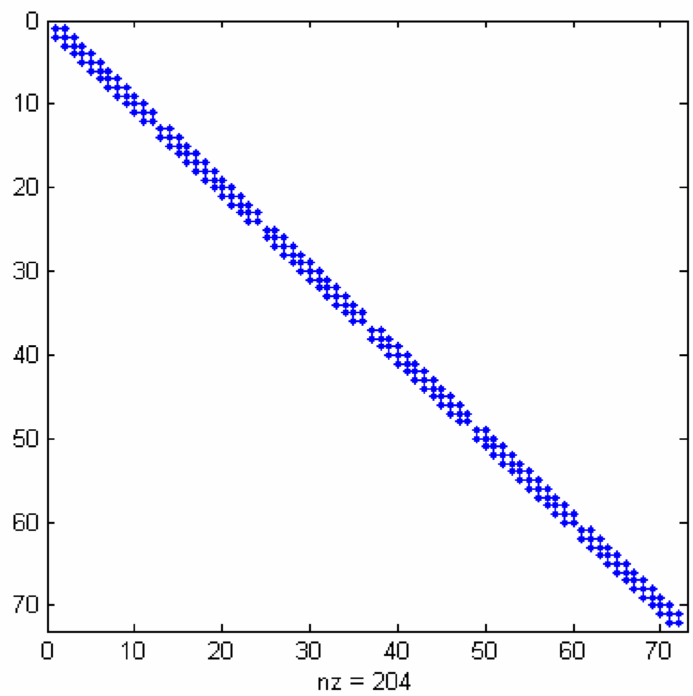
- Local Matrix Generation
  - Ordering node number according to the routing direction of each layer



Pattern Based Method

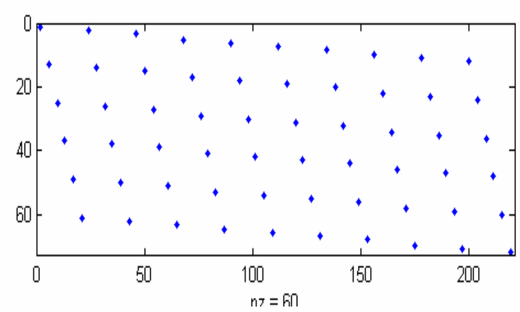
- Diagonal Block
  - transform topology similarity to matrix similarity
  - one tridiagonal matrix need to be stored

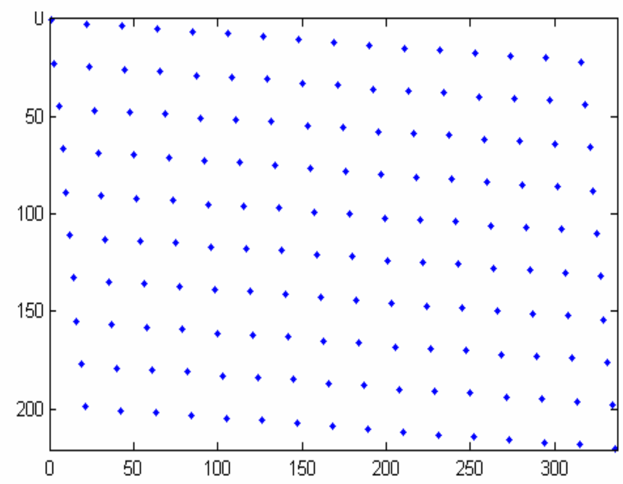




Pattern Based Method

- Via Block
  - All the vias share the same value between two adjacent layers
  - Fill in of via element is regular : size , pitch
  - No need to store sub matrixes caused by vias





Pattern Based Method

- Drawback of traditional preconditioner
  - Performance of ILU or incomplete choleskey decomposition are restricted to memory usage
  - More fill in cause faster convergence speed

No odering residual=1e-6		
Drop threshold	Fill in ratio	Avg Iter times
1e-1	1.5	221
1e-2	2.79	93
1e-3	10.96	44
1e-4	48.41	17

Pattern Based Method

- Preconditioner Generation: Two Metal Layer Case

$$\begin{bmatrix} A & C \\ C^T & B \end{bmatrix} \quad A = \begin{bmatrix} a & & \\ & a & \\ & & a \end{bmatrix} \quad B = \begin{bmatrix} b & & \\ & b & \\ & & b \end{bmatrix} \quad C = c_{ij} \begin{cases} j = \alpha \bmod(i, \beta) & c_{ij} = g_{via} \\ other & c_{ij} = 0 \end{cases}$$

- Schur-alike Decomposition to get approximate inverse

$$\begin{bmatrix} A & C \\ C^T & B \end{bmatrix} = \begin{bmatrix} I & \\ C^T A^{-1} & I \end{bmatrix} \begin{bmatrix} A & \\ & B' \end{bmatrix} \begin{bmatrix} I & A^{-1}C \\ & I \end{bmatrix}$$

$$B' = B - C^T A^{-1} C$$

$$\begin{bmatrix} A & C \\ C^T & B \end{bmatrix}^{-1} = \begin{bmatrix} I & -A^{-1}C \\ & I \end{bmatrix} \begin{bmatrix} A^{-1} & \\ & B'^{-1} \end{bmatrix} \begin{bmatrix} I & \\ -C^T A^{-1} & I \end{bmatrix}$$

Pattern Based Method

- Preconditioner Generation
  - Perform Precondition in CG Algorithm

$$P \cdot r = \begin{bmatrix} A & C \\ C^T & B \end{bmatrix}^{-1} r = \begin{bmatrix} I & -A^{-1}C \\ & I \end{bmatrix} \begin{bmatrix} A^{-1} & \\ & B'^{-1} \end{bmatrix} \begin{bmatrix} I & \\ -C^T A^{-1} & I \end{bmatrix} r$$

$$step\,1:\ \ r_1 = \begin{bmatrix} r_1^1 \\ r_1^2 \end{bmatrix} = \begin{bmatrix} I & \\ -C^T A^{-1} & I \end{bmatrix} r \ = \begin{bmatrix} r^1 \\ -C^T A^{-1} r^1 + r^2 \end{bmatrix} = \begin{bmatrix} r^1 \\ -C^T p + r^2 \end{bmatrix} \quad p = A^{-1} r^1$$

$$step\,2:\ r_2 = \begin{bmatrix} r_2^1 \\ r_2^2 \end{bmatrix} = \begin{bmatrix} A^{-1} & \\ & B'^{-1} \end{bmatrix} r_1 = \begin{bmatrix} p \\ B'^{-1} r_1^2 \end{bmatrix}$$

$$step\,3:\ r_3 = \begin{bmatrix} I & -A^{-1}C \\ & I \end{bmatrix} r_2 = \begin{bmatrix} r_2^1 - A^{-1} C r_2^2 \\ r_2^2 \end{bmatrix}$$

Pattern Based Method

- Preconditioner Generation
  - Inverse of diagonal block

$$A^{-1}r = \begin{bmatrix} a^{-1} & & \\ & a^{-1} & \\ & & a^{-1} \end{bmatrix} r$$

  - Other matrix vector dot operation

$$\begin{bmatrix} A & C \\ C^T & B \end{bmatrix} r = \begin{bmatrix} Ar^1 + Cr^2 \\ Br^2 + C^T r^1 \end{bmatrix}$$

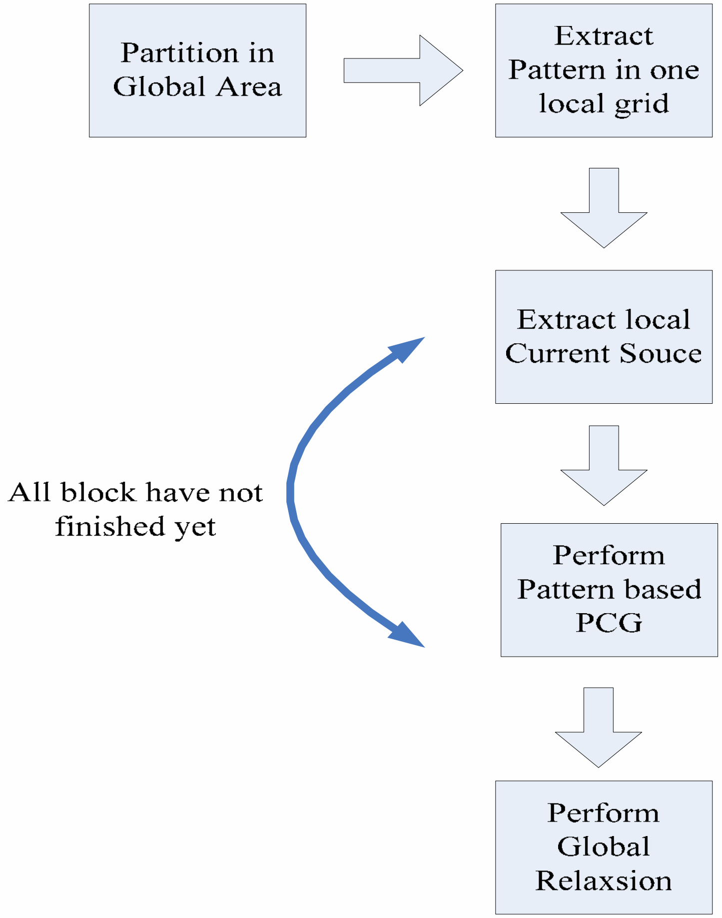
$$Ar = \begin{bmatrix} a & & \\ & a & \\ & & a \end{bmatrix} r$$

Pattern Based Method

- Benefit of Using Pattern Structure
  - Memory cost is reduced from $O(n)$ to less than $O(n^{0.5})$
  - Reuse of sub matrixes and vectors can reduce Cache Miss dramatically
  - Better preconditioner can be constructed to accelerate convergence speed

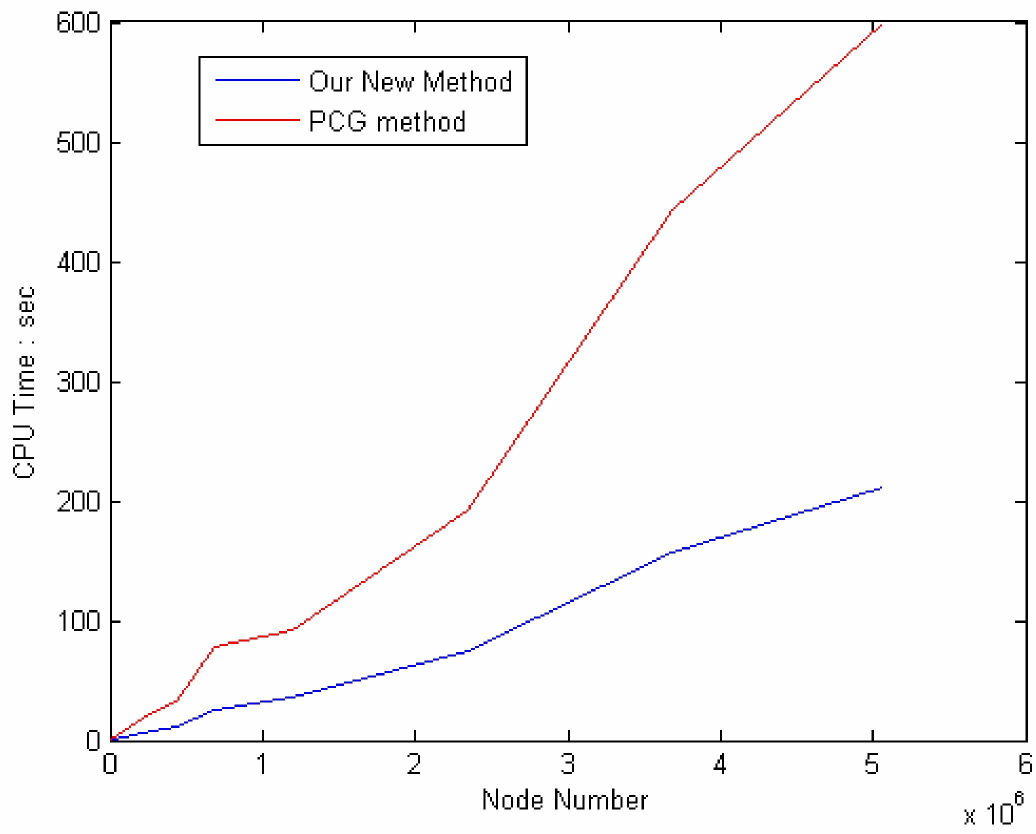
Pattern Based Method

- Algorithm Flow



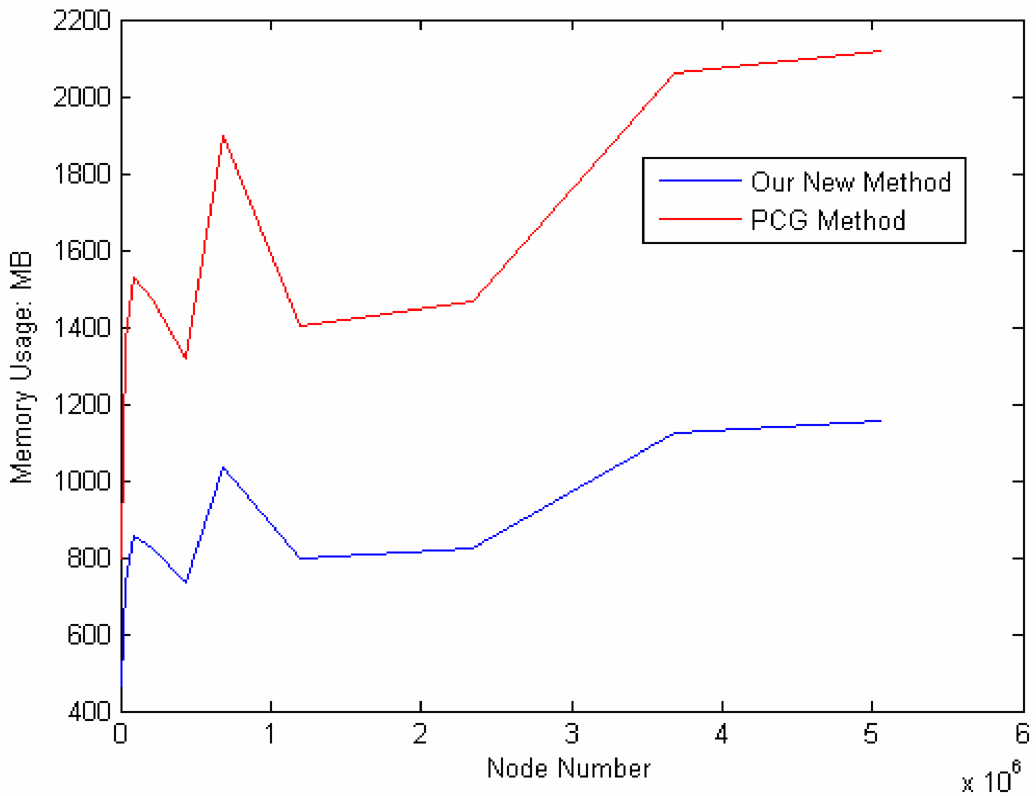
Pattern Based Method

- Experimental Performance: Speed



Pattern Based Method

- Experimental Performance: Memory



Conclusion

- New pattern based method is presented
- Combine advantages of direct method and iterative method
- Decrease the Cache Miss dramatically
- Reduce the memory efficiency dramatically
- New preconditioner is faster than traditional preconditioner
- Fast linear PCG can be achieved even the grid size is huge

Thank you !