Design Automation Group

# Parallel Multigrid Preconditioning on Graphics Processing Units (GPUs) for Robust Power Grid Analysis


**Zhuo Feng**
**Michigan Technological University**

**Zhiyu Zeng**
**Texas A&M University**



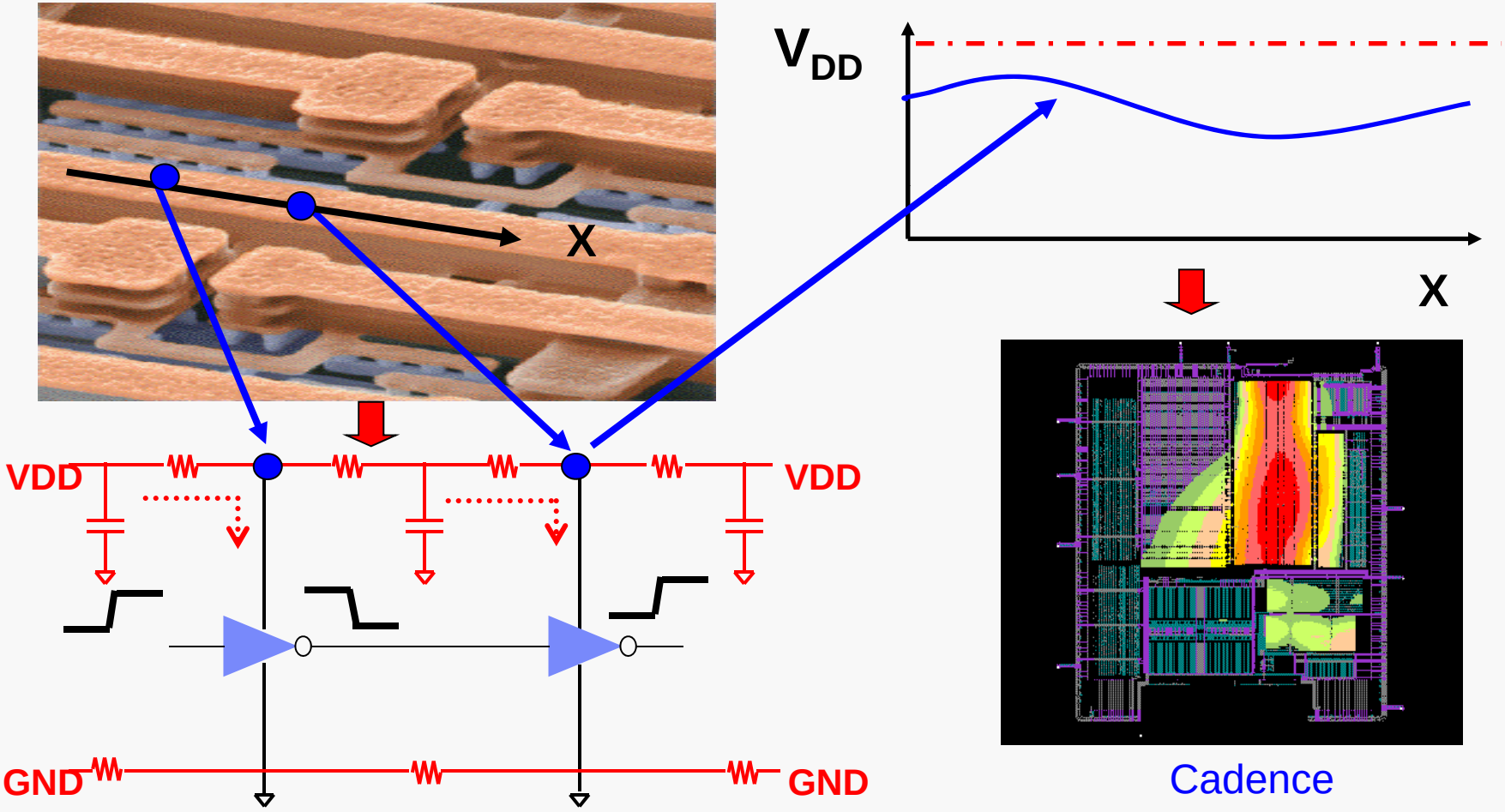
**2010 ACM/EDAC/IEEE Design Automation Conference**

# Motivation

- On-chip power distribution network verification challenge
  - Tens of millions of grid nodes (recent IBM design reaches ~400M)
  - Need long simulation time for transient power grid verification
- Parallel circuit simulation algorithms on GPUs
  - Pros: very cost efficient: 240-core GPU costs $400
  - Hardware resource usage limitations:
    - Shared memory size, number of registers, etc
  - Algorithm and data structure design preferences:
    - Multilevel iterative algorithms for SIMD computing platform
    - GPU-friendly device memory access patterns, simple control flow
- Our contribution: a robust power grid simulation method for GPU
  - Multigrid preconditioning assures fast convergence (< 20 iterations)
  - GPU-specific data structure guarantee coalesced memory access

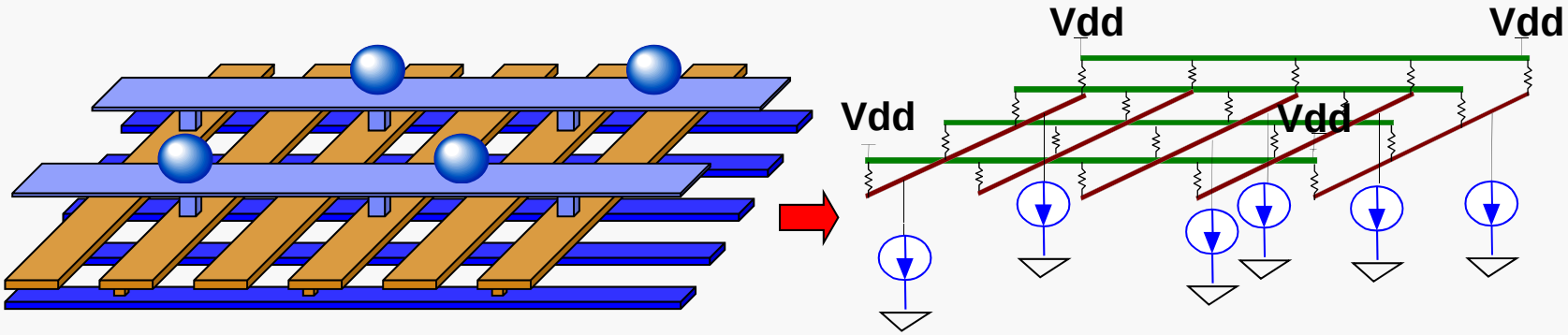
# IR Drop in Power Distribution Network

- **IR drop:** voltage drop due to non-ideal resistive wires



# Power Grid Modeling & Analysis

- Multi-layer interconnects are modeled as 3D RC network
  - Switching gate effects are modeled by time-varying current loadings




- DC analysis solves linear system

$$G \cdot \vec{v} = \vec{b}$$

**Tens of millions of unknowns !**

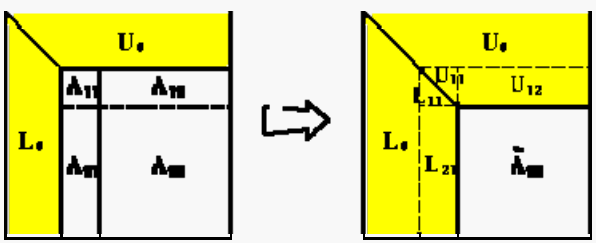
- Transient analysis solves

$$G \cdot \vec{v}(t) + C \cdot \frac{d\vec{v}(t)}{dt} = \vec{b}(t)$$

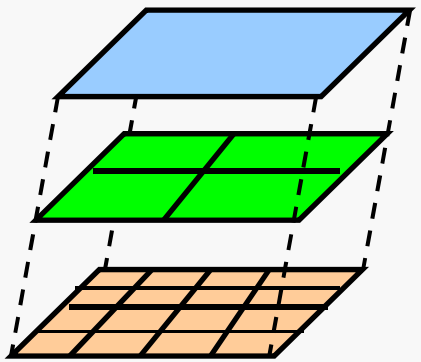
| | |
|---|---|
| $G \in \Re^{n \times n}$ : | **Conductance Matrix** |
| $C \in \Re^{n \times n}$ : | **Capacitance Matrix** |
| $\vec{v} \in \Re^{n \times 1}$ : | **Node Voltage Vector** |
| $\vec{b} \in \Re^{n \times 1}$ : | **Current Loading Vector** |

# Prior Work

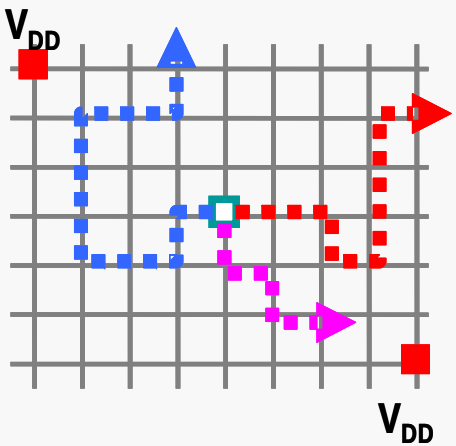
- Prior power grid analysis approaches
  - Direct methods (LU factorization, Cholesky decomposition)
    - Cholmod uses 7GB memory and >1,000 s for a 9-million grid
  - Iterative methods
    - Preconditioned conjugate gradient (*T. Chen et al, DAC'01*)
    - Multigrid methods (*S. Nassif et al, DAC'00*)
  - Stochastic method
    - Random walk (*H. Qian et al, DAC'05*)



**Direct Method**



**Multigrid**



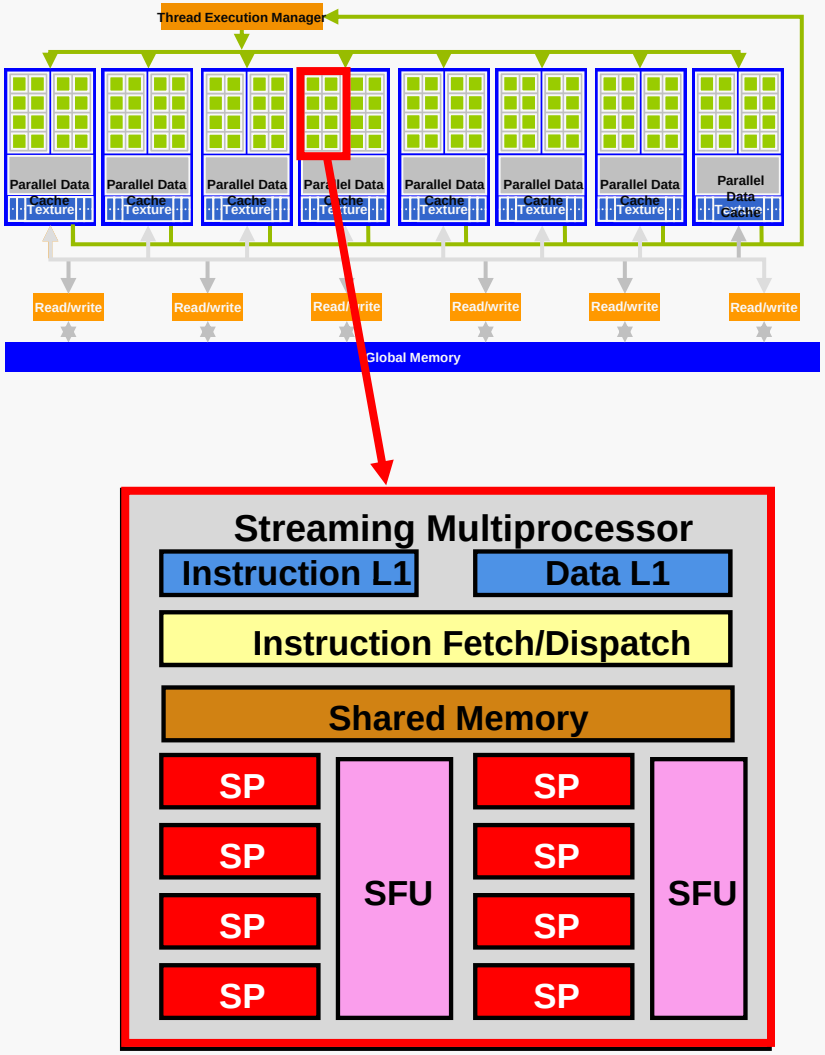

**Random walk**

# Prior Work (Cont.)

- Recent GPU based power grid analysis methods
  - Hybrid Multigrid method on GPU (*Z. Feng et al, ICCAD'08*)
    - Pros: very fast (solves four million nodes per second)
    - Cons: convergence rate depends on 2D grid approximation
  - Poisson Solver (*J. Shi et al, DAC'09*)
    - Pros: public CUFFT library -> easier implementation
    - Cons: only suitable 2D regular grids
- Robust preconditioned Krylov subspace iterative methods on GPU
  - Preconditioners using incomplete LU or Cholesky matrix factors
    - Matrix factors are hard to store and process on GPU
  - Multigrid based preconditioning methods
    - SIMD multigrid solver + sparse matrix-vector operations on GPU

# NVIDIA GPU Architecture

- Streaming Multiprocessor (SM)
  - 8 streaming processors (SP)
  - 2 special function units (SFU)
  - Multithreaded instruction fetch/dispatch unit
- Multi-threaded instruction dispatch
  - 1 to 512 active threads
  - 32 threads (a warp) share one instruction fetch
  - Cover memory load latency
- Some facts about an SM
  - 16 KB shared memory
  - 8,196 registers
  - >30 Gflops peak performance

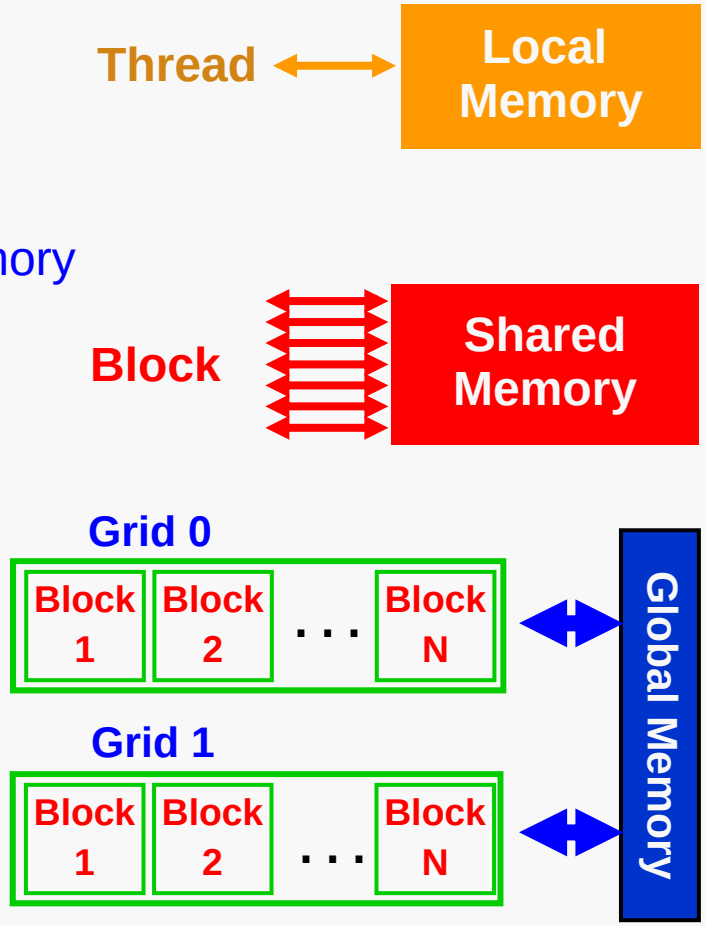


# GPU Memory Space (CUDA Memory Model)

- **Each thread:**
  - R/W per-thread local memory
  - R/W per-block shared memory
  - R/W per-grid global memory
  - Read only per-grid texture/constant memory

| | Local | Shared | Global | Texture |
|---|---|---|---|---|
| Read | Yes | Yes | Yes | Yes |
| Write | Yes | Yes | Yes | No |
| Size | Large | Small | Large | Large |
| BW | High | High | High | High |
| Cached? | No | Yes | No | Yes |
| Latency | 500 cyc. | 20 cyc. | 500 cyc. | 300 cyc. |

Device Memory Comparison

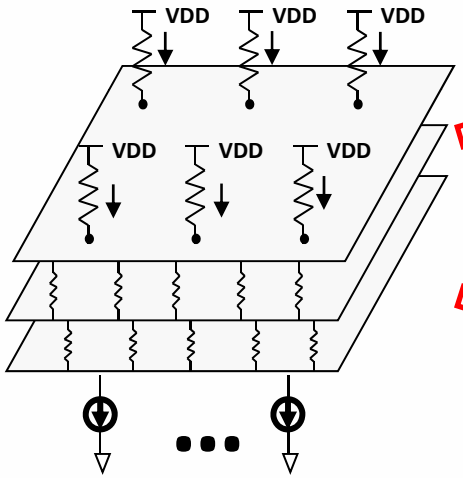


# Contribution of This Work

- **Multigrid preconditioned Krylov subspace iterative solver on GPU**

## Host (CPU) Memory

**3D Multi-layer Irregular Power Grid**




**DC :** $Gx = b$

**TR :** $Gx(t) + C\dfrac{dx(t)}{dt} = b(t)$

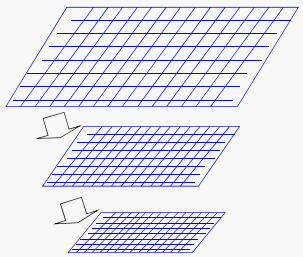
## GPU Global Memory

**Jacobi Smoother using Sparse Matrix**

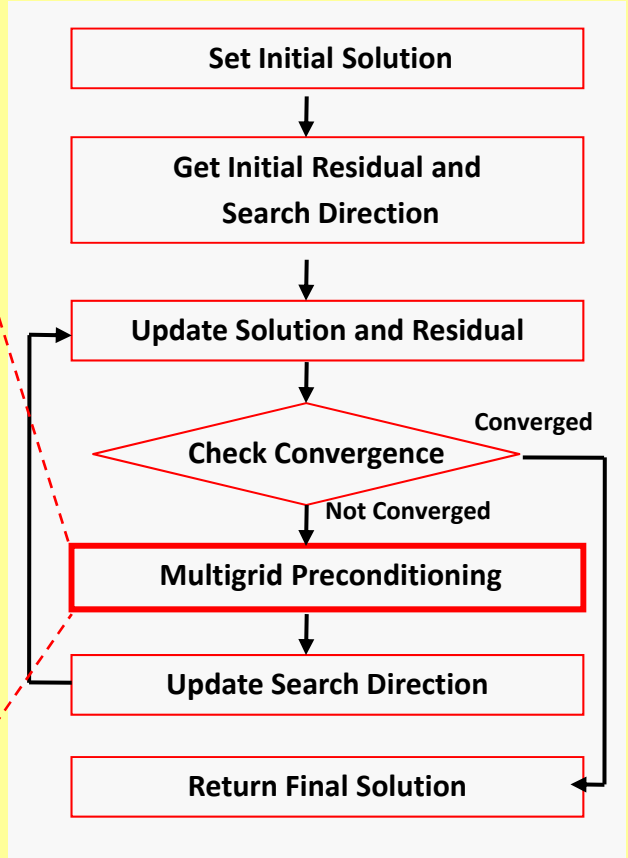
$$\begin{bmatrix} a_{1,1} & & & a_{1,4} & a_{1,5} & & & \\ & a_{2,2} & a_{2,3} & & & a_{2,6} & & \\ & a_{3,2} & a_{3,3} & & & & & a_{3,8} \\ a_{4,1} & & & a_{4,4} & & & & \\ a_{5,1} & & & & a_{5,5} & & a_{5,7} & \\ & a_{6,2} & & & & a_{6,6} & & a_{6,8} \\ & & & & a_{7,5} & & a_{7,7} & \\ & & a_{8,3} & & & a_{8,6} & & a_{8,8} \end{bmatrix}$$

+

**Geometrical Multigrid Solver (GMD)**



## MGPCG Algorithm on GPU



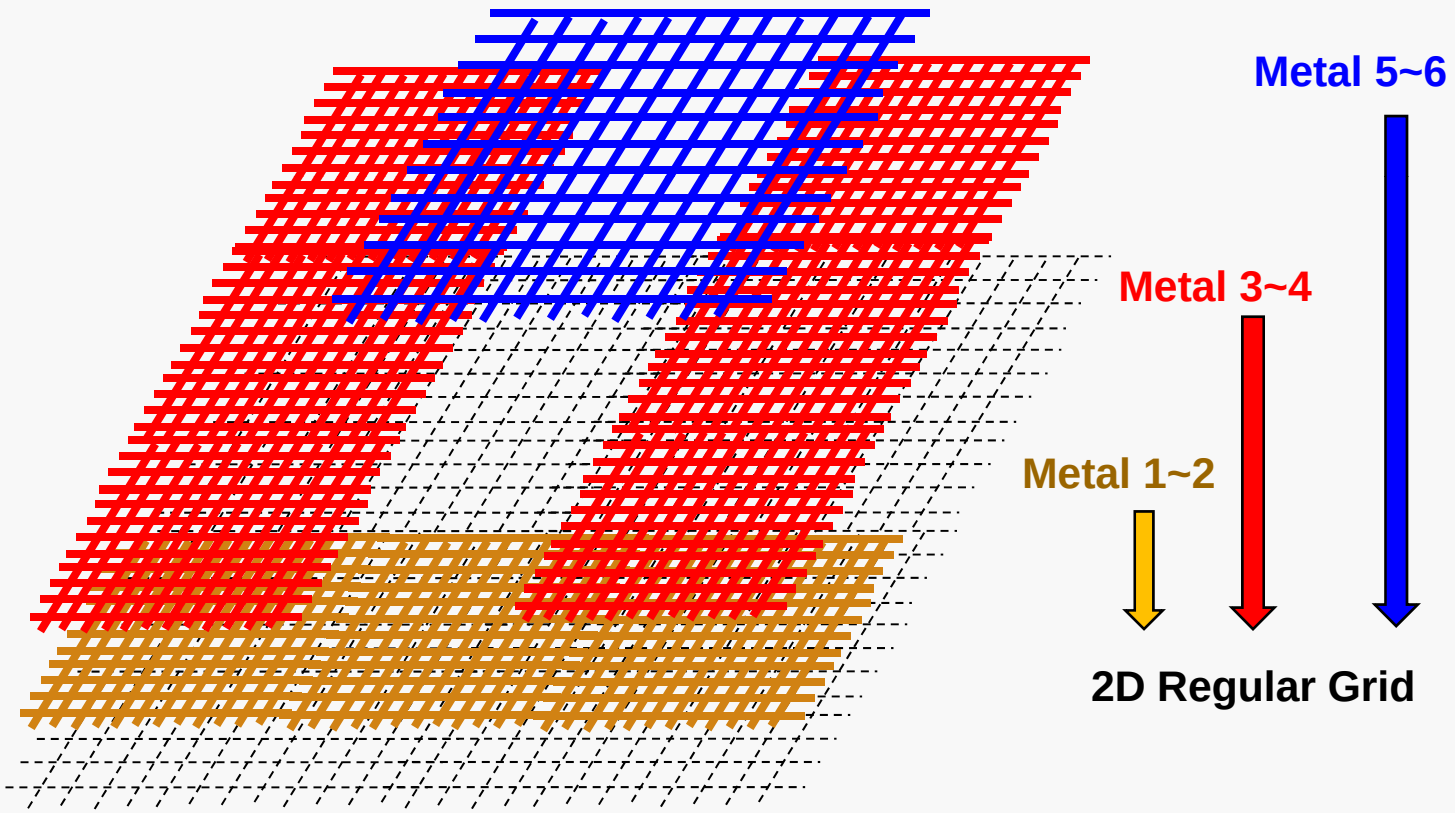

**Original Grid** → **Matrix + Geometrical Representation** → **GPU-friendly Multi-level Iterative Algorithm**

# Multigrid Methods

- Among fastest numerical algorithms for PDE-like problems
  - Linear complexity in the number of unknowns
- A hierarchy of exact to coarse replicas of the problem
  - High (low) frequency errors damped on fine (coarse) grids
  - Direct/iterative solvers for coarsest grid
- Multigrid operations
  - Smoothing, restriction, prolongation and correction, etc
- Algebraic MG (AMG) and Geometric MG (GMD)
  - GMD: suitable for GPU's SIMD computation
  - AMG: robust for handling irregular grids, but needs irregular memory access and complex control flow

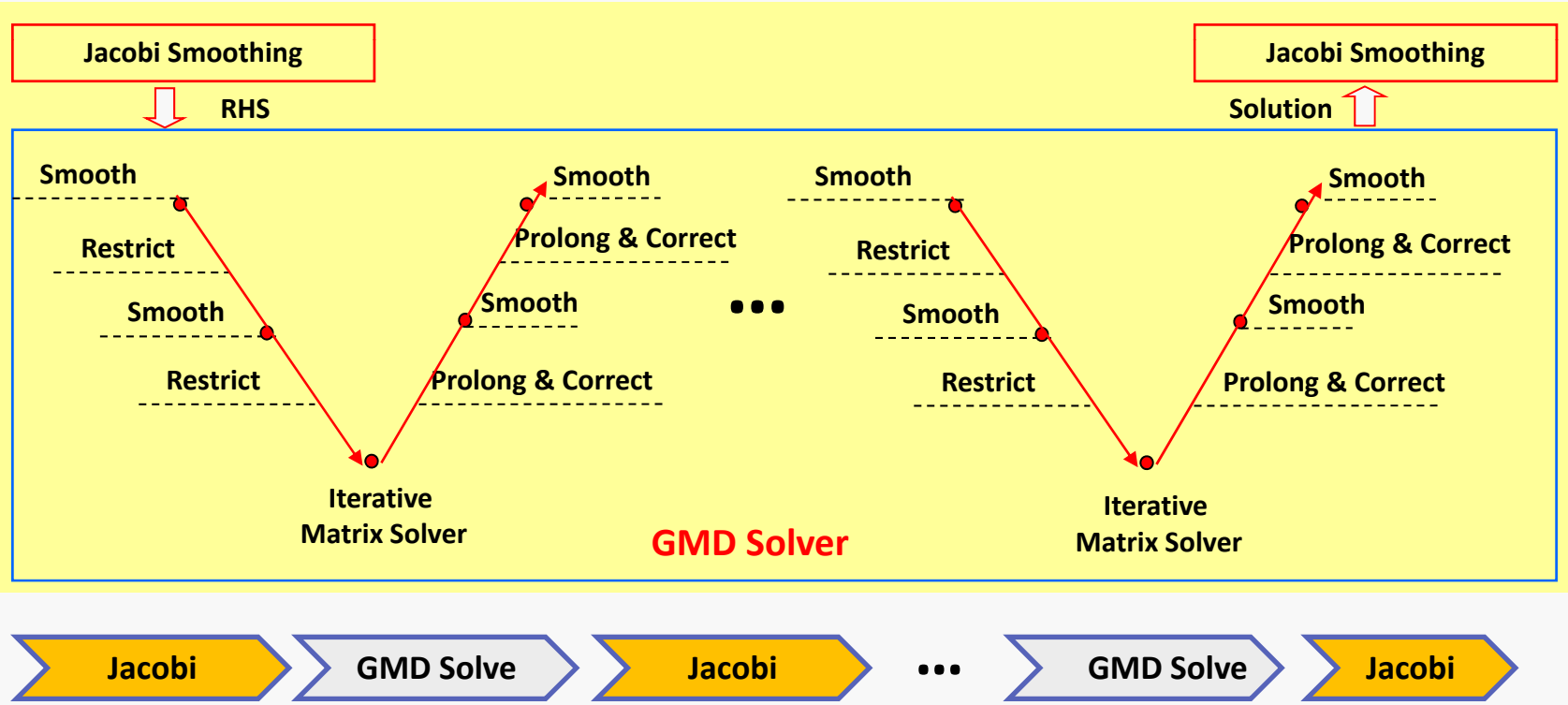
# Power Grid Topology Regularization

**Location-based mapping** (*Z. Feng et al, ICCAD'08*)



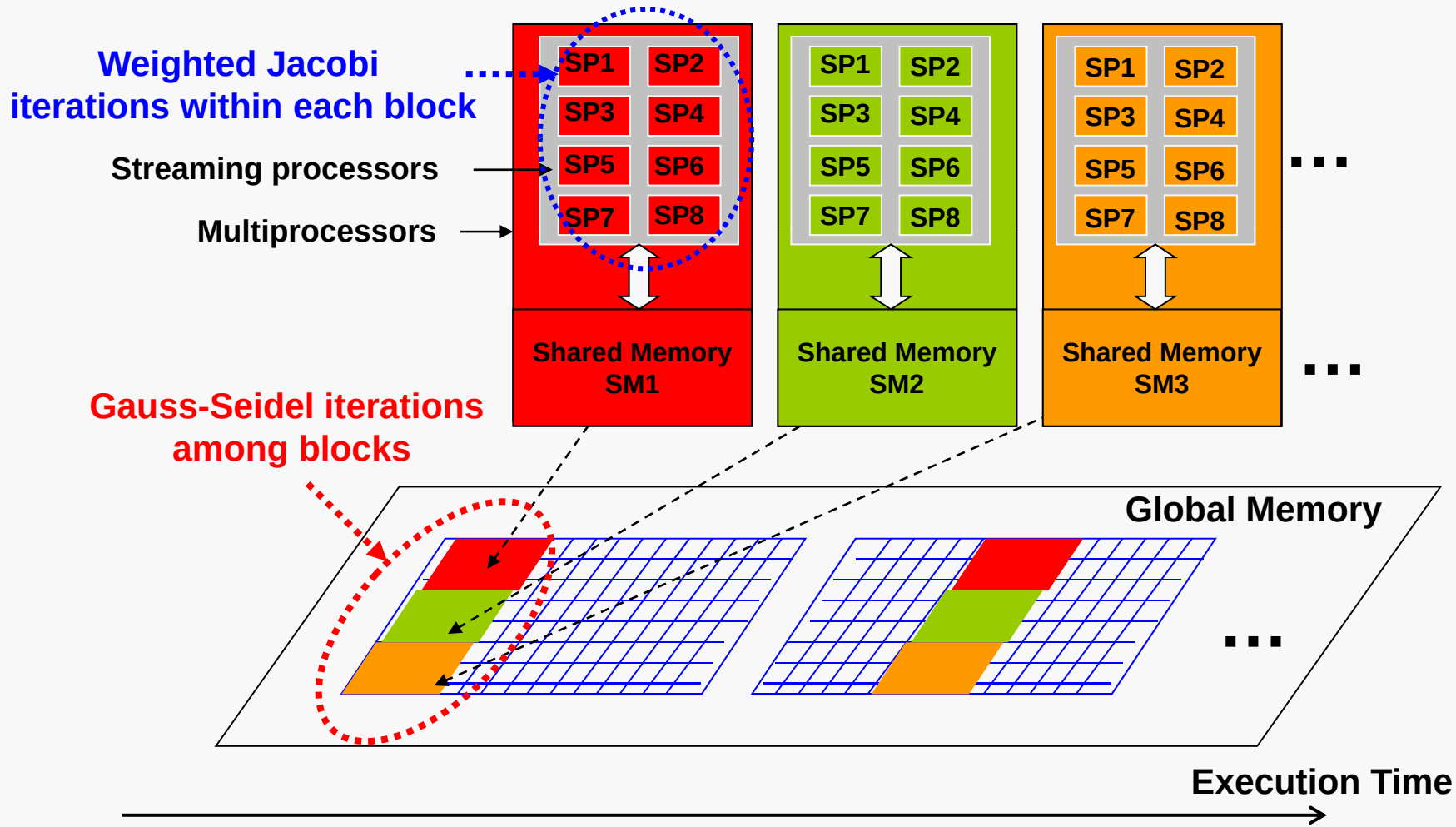
# Parallel Multigrid Preconditioning

- **3D grid smoother + 2D gird GMD solver**
  - 3D finest grid is stored using ELL-like sparse matrix format
  - 2D coarser to coarsest grids are processed geometrically
- **Coalesced memory accesses are guaranteed on GPU**



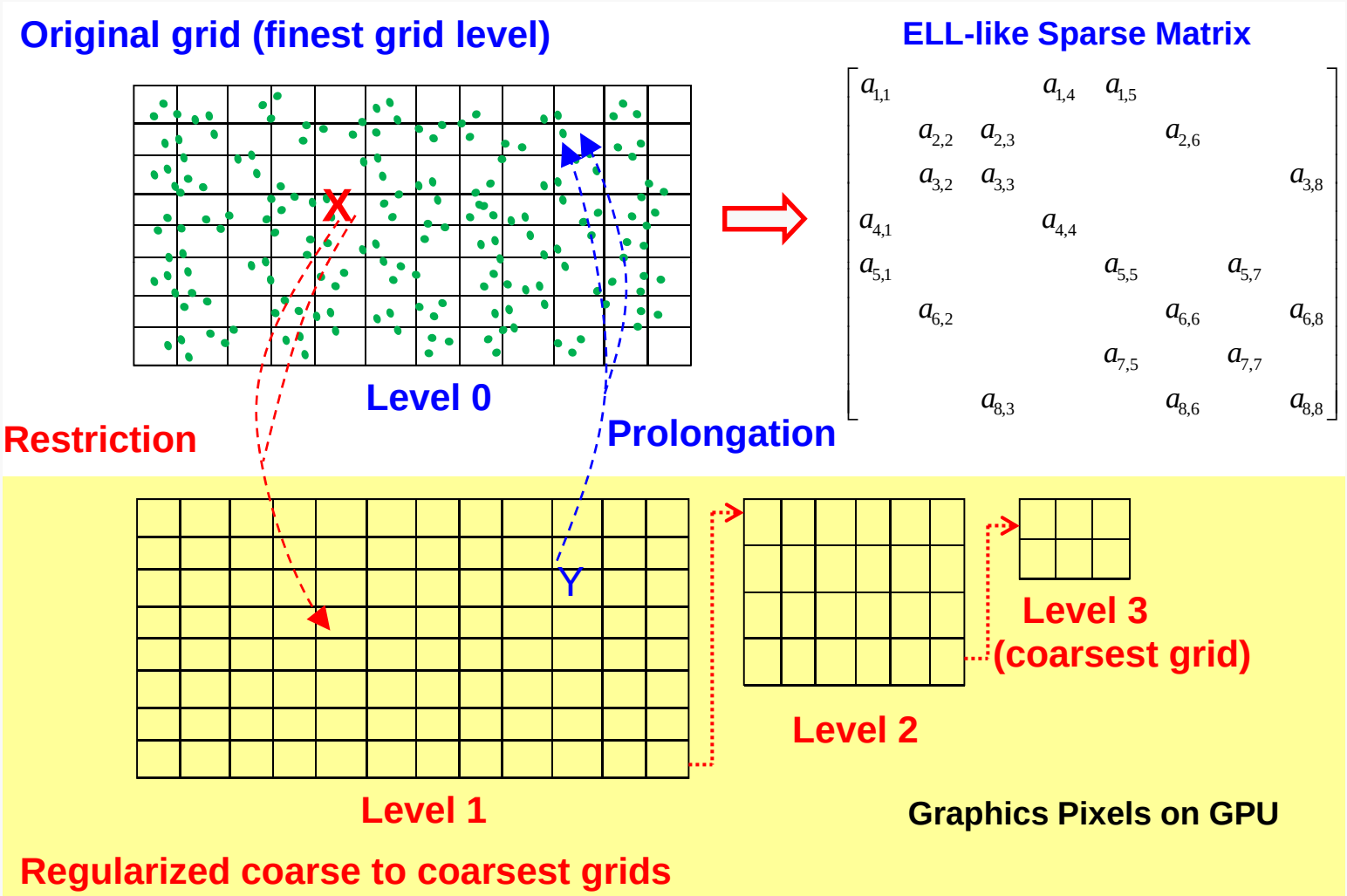
# GMD Smoother

- **Mixed block-wise relaxation on GPU**



# Memory Layout on GPU

- **Mixed data structures**



# Nodal Analysis Matrix

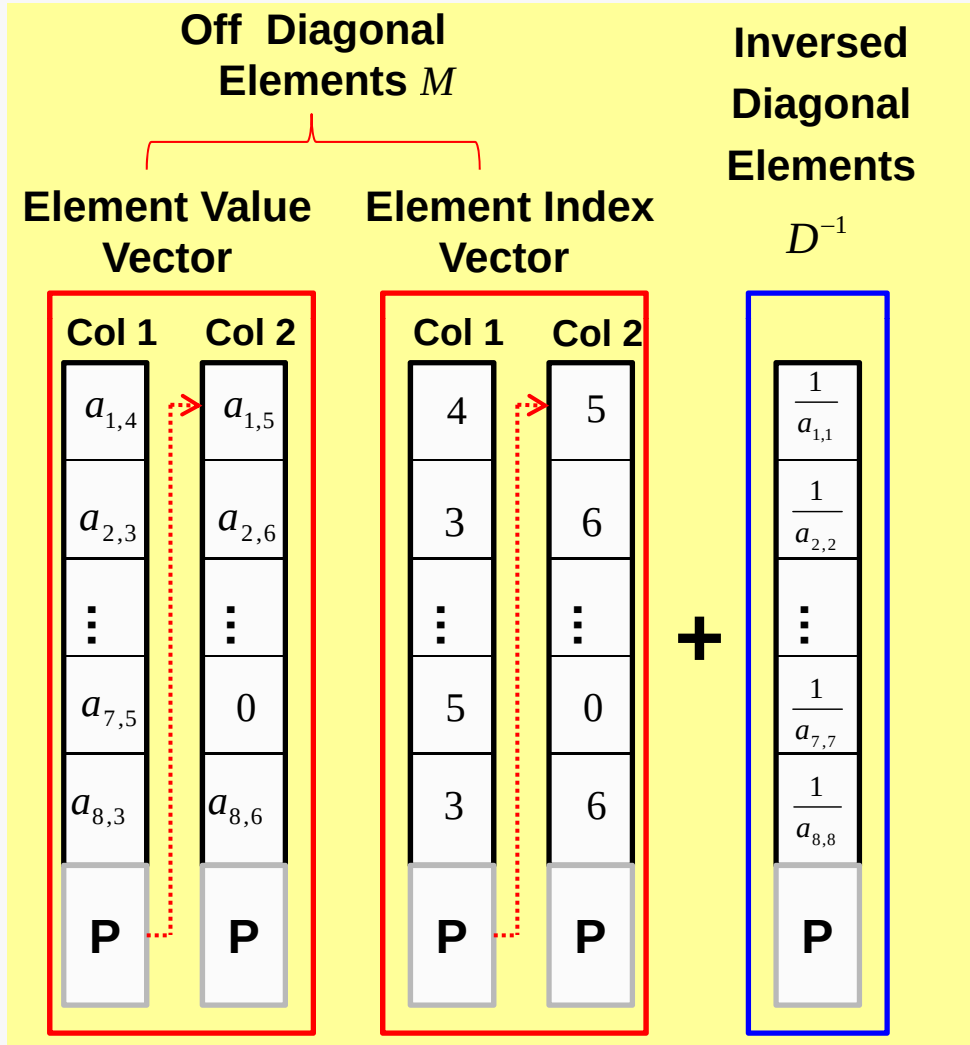
- **ELL-like sparse matrix storage**

$$\begin{bmatrix} a_{1,1} & & & a_{1,4} & a_{1,5} & & & \\ & a_{2,2} & a_{2,3} & & & a_{2,6} & & \\ & a_{3,2} & a_{3,3} & & & & & a_{3,8} \\ a_{4,1} & & & a_{4,4} & & & & \\ a_{5,1} & & & & a_{5,5} & & a_{5,7} & \\ & a_{6,2} & & & & a_{6,6} & & a_{6,8} \\ & & & & a_{7,5} & & a_{7,7} & \\ & & a_{8,3} & & & a_{8,6} & & a_{8,8} \end{bmatrix}$$

$$A = D + M$$

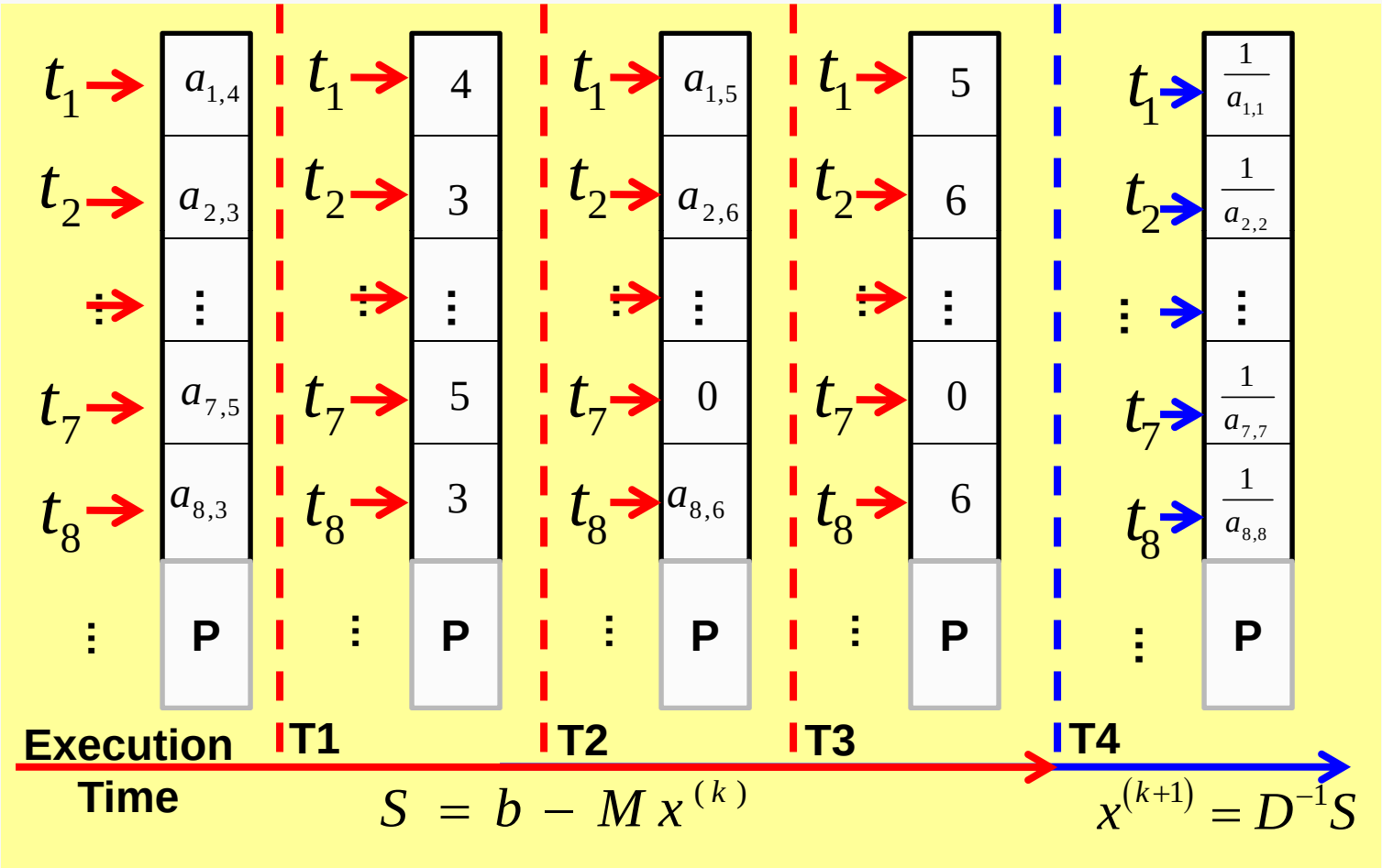
$D$ **: Diagonal Elements of A**

$M$ **: Off-Diagonal Elements of A**



# GPU Device Memory Access Pattern

- **GPU-based Jacobi Iteration (smoother):** $x^{(k+1)} = D^{-1}\left[b - Mx^{(k)}\right]$



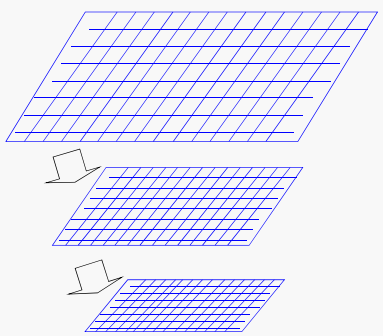
# Algorithm Flow

**Jacobi Smoother using Sparse Matrix**

$$\begin{bmatrix} a_{1,1} & & & a_{1,4} & a_{1,5} & & & \\ & a_{2,2} & a_{2,3} & & & a_{2,6} & & \\ & a_{3,2} & a_{3,3} & & & & & a_{3,8} \\ a_{4,1} & & & a_{4,4} & & & & \\ a_{5,1} & & & & a_{5,5} & & a_{5,7} & \\ & a_{6,2} & & & & a_{6,6} & & a_{6,8} \\ & & & & a_{7,5} & & a_{7,7} & \\ & & a_{8,3} & & & a_{8,6} & & a_{8,8} \end{bmatrix}$$

**+**

**Geometrical Multigrid Solver (GMD)**



**Set Initial Solution**

↓

**Get Initial Residual and Search Direction**

↓

**Update Solution and Residual**

↓

**Check Convergence**

**Not Converged** ↓

**Multigrid Preconditioning**

↓

**Update Search Direction**

**Converged** → **Return Final Solution**

# Experiment Results

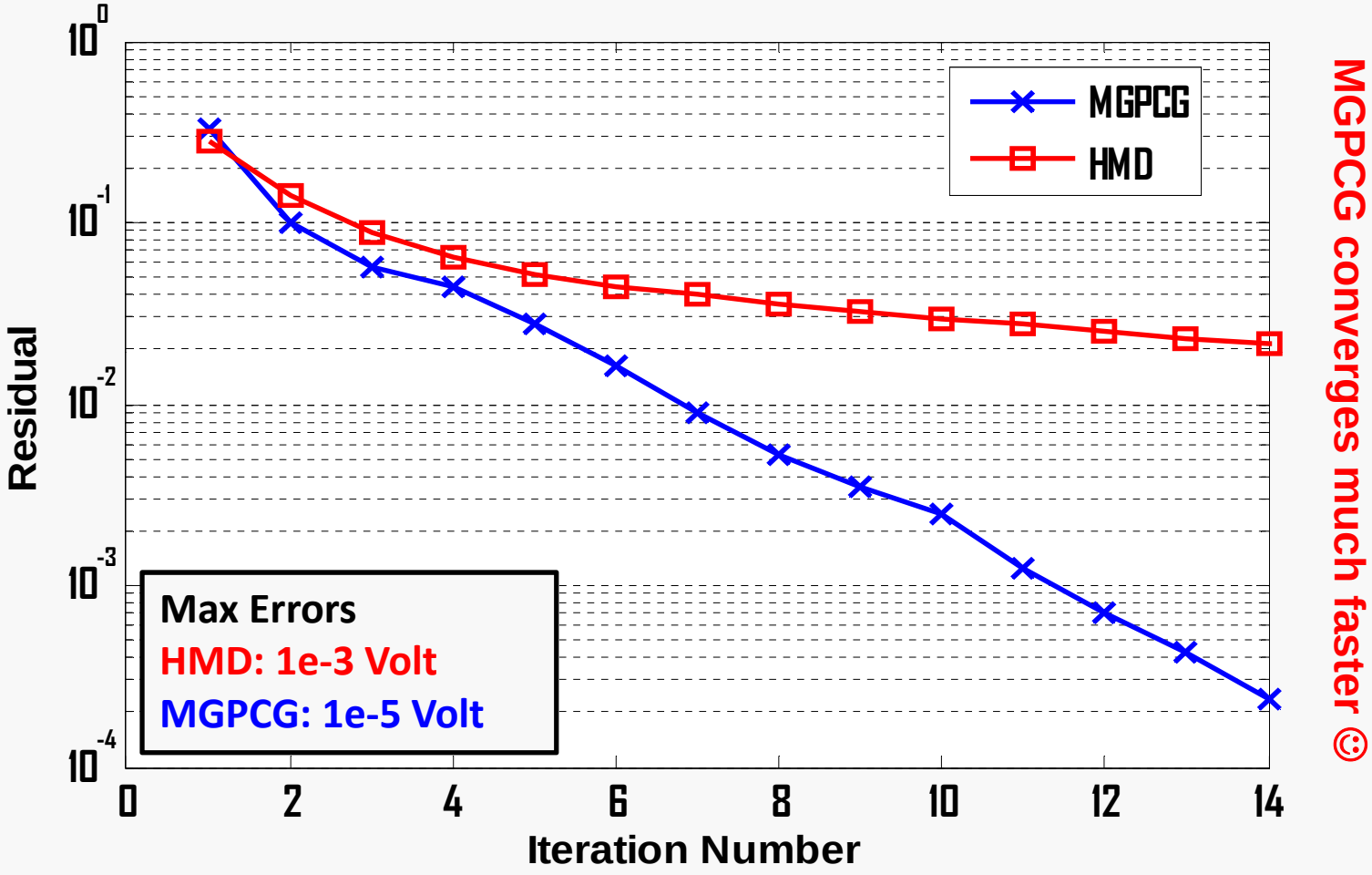
- Linux computing system: C++ & CUDA
  - CPU: Core 2 Quad 2.66GHz + 6GB DRAM
  - GPU: NVIDIA GTX 285 1.5GHz with 240 SPs ($400)
- Power grid test cases
  - IBM power grid benchmark circuits CKT1~5 (0.13M ~ 1.7 M)
  - Larger industrial power grid designs CKT6~8 (4.5 M ~ 10M)
- Direct solver on the host
  - Cholmod with Supernodal and Metis functions
- Iterative solvers on GPU
  - MGPCG: multigrid preconditioned CG
  - DPCG: diagonally preconditioned CG
  - HMD: hybrid multigrid (*Z. Feng et al, ICCAD'08*)

# Power Grid Design Information

| CKT | N_node | N_layer | N_nnz | N_res | N_cur |
|---|---|---|---|---|---|
| CKT1 | 127.0K | 5 | 532.9K | 209.7K | 37.9K |
| CKT2 | 851.6K | 5 | 3.7M | 1.4M | 201.1K |
| CKT3 | 953.6K | 6 | 4.1M | 1.5M | 277.0K |
| CKT4 | 1.0M | 3 | 4.3M | 1.7M | 540.8K |
| CKT5 | 1.7M | 3 | 6.6M | 2.5M | 761.5K |
| CKT6 | 4.7M | 8 | 18.8M | 6.8M | 185.5K |
| CKT7 | 6.7M | 8 | 26.2M | 9.7M | 267.3K |
| CKT8 | 10.5M | 8 | 40.8M | 14.8M | 419.3K |

- N_layer: the number of metal layers
- N_res: the number of resistors
- N_cur: the number of current sources

# Convergence Comparison




- HMD: Hybrid Multigrid Method
- MGPCG: Multigrid Preconditioned Conjugate Gradient Method

# Results

**Power Grid DC Analysis**

| CKT | NCG | NDPCG | NMGPCG | NHMD | TCG | TDPCG |
|---|---|---|---|---|---|---|
| CKT1 | 1,405 | 400 | 4 | 7 | 0.2 | 0.12 |
| CKT2 | 4,834 | 3,351 | 4 | 13 | 5.9 | 3.9 |
| CKT3 | 2,253 | 681 | 3 | 8 | 5.4 | 2.1 |
| CKT4 | 4,062 | 411 | 5 | >30 | 5.3 | 3.7 |
| CKT5 | 6,433 | 700 | 6 | >30 | 15.2 | 9.5 |

- NCG: the number of CG iterations
- NDPCG: the number of diagonally preconditioned CG iterations
- NMGPCG: the number of multigrid preconditioned CG iterations
- NHMD: the number of hybrid multigrid iterations
- TCG: the runtime of CG
- TDPCG: the runtime of diagonally preconditioned CG iterations

# Results (Cont.)

**Power Grid DC Analysis (Cont.)**

| CKT | TMGPCG | THMD | TCHOL | Eavg | Emax | Speedup |
|---|---|---|---|---|---|---|
| CKT1 | 0.05 | 0.1 | 1.7 | 1e-4 | 4e-4 | 34X |
| CKT2 | 0.5 | 0.78 | 20.2 | 2e-6 | 2e-5 | 40X |
| CKT3 | 0.4 | 0.72 | 21.6 | 1e-5 | 1e-4 | 54X |
| CKT4 | 0.9 | >2.4 | 19.4 | 8e-5 | 7e-4 | 22X |
| CKT5 | 1.1 | >4.5 | 25.6 | 1e-4 | 5e-4 | 25X |

- TMGPCG: the runtime of multigrid preconditioned CG iterations
- THMD: the runtime of hybrid multigrid iterations
- TCHOL: the runtime of direct matrix solver (Cholmod)
- Eavg: average error
- Emax: maximum error
- Speedup: TCHOL/TMGPCG

# Results (Cont.)

**DC Analysis of Large Circuits**

| CKT | N_node | N_MGPCG | T_MGPCG | T_CHOL | Speedup |
|---|---|---|---|---|---|
| CKT6 | 4.7M | 7 | 4.9 | 131.5 | 27X |
| CKT7 | 6.7M | 9 | 7.9 | 205.1 | 26X |
| CKT8 | 10.5M | 11 | 11.6 | N/A | N/A |

- N_MGPCG: the number of MGPCG iterations
- T_MGPCG: the runtime of MGPCG solver
- T_CHOL: the runtime of the Cholmod solver

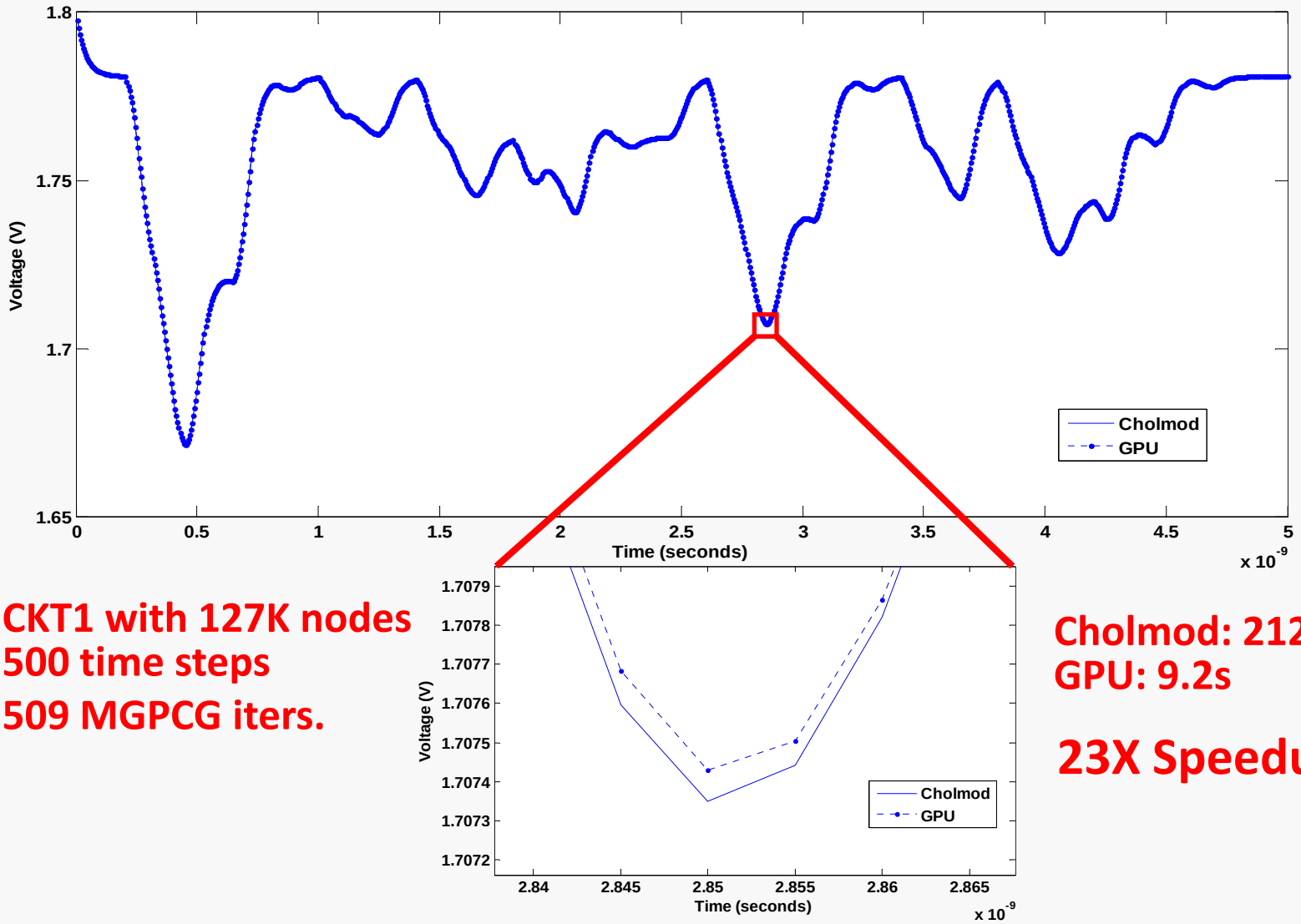
# Results (Cont.)

**Transient Analysis Results**

| CKT | Tcpu | Tgpu | Ngpu | Eavg | Emax | Speedup |
|---|---|---|---|---|---|---|
| CKT1 | 40.8 | 2.1 | 102 | 3e-6 | 8e-4 | 20X |
| CKT2 | 315.3 | 15.2 | 99 | 1e-5 | 3e-4 | 21X |
| CKT3 | 360.7 | 15.6 | 100 | 5e-6 | 1e-4 | 23X |
| CKT4 | 352.7 | 19.6 | 140 | 3e-5 | 2e-4 | 18X |
| CKT5 | 553.9 | 26.1 | 130 | 2e-5 | 1e-4 | 21X |

- Tcpu: Cholmod solve time
- Tgpu: MGPCG time
- Ngpu: the number of MGPCG iterations

# Transient Analysis: CKT1

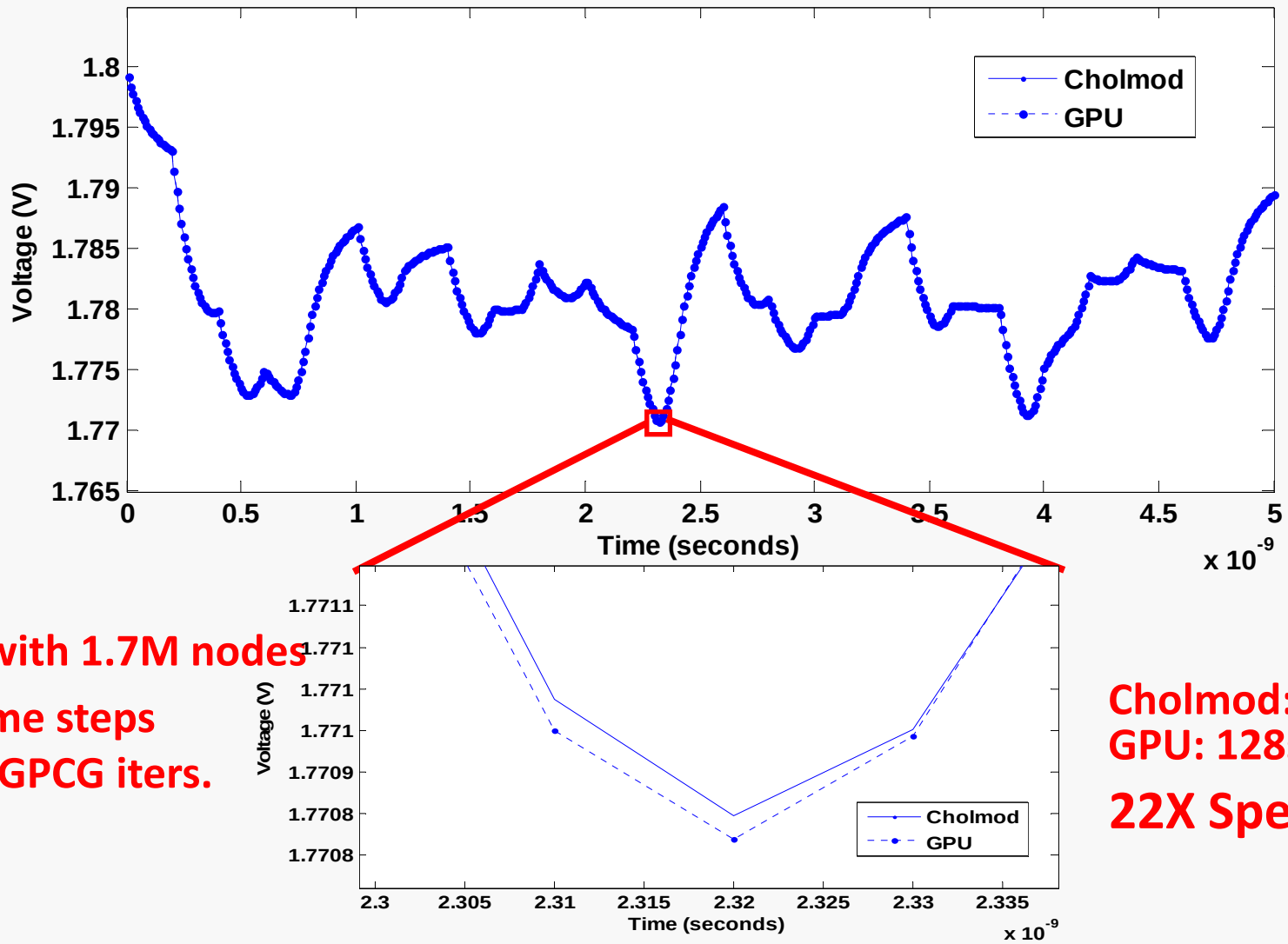



**CKT1 with 127K nodes**
**500 time steps**
**509 MGPCG iters.**

**Cholmod: 212s**
**GPU: 9.2s**

**23X Speedups**

# Transient Analysis: CKT5




**CKT5 with 1.7M nodes**

**500 time steps**

**693 MGPCG iters.**

**Cholmod: 2,700s**
**GPU: 128s**

**22X Speedups**

# Conclusion and Future Work

- Robust circuit simulation on GPU is challenging
  - How to accelerate simulations for irregular problems?
  - Hard to guarantee the accuracy and robustness?

- Parallel multigrid preconditioning method for power grid analysis
  - Multigrid preconditioning (geometrical + matrix representations)
    - Geometrical multigrid solver on GPU
    - ELL-like sparse matrix-vector operations for original grids on GPU
  - Applicable to more general power grids with strong irregularities
  - Much faster convergence & higher accuracy then ever before

- Future work
  - Node ordering and grid partitioning for Multi-Core-Multi-GPUs
  - GPU performance modeling for further improving the solver efficiency
  - Heterogeneous computing to adaptively balance the work loads