

GRID EMT

SIMULATION INITIATIVE

Solvers in RE- INTEGRATE

Phani Marthi, Vibhuti Sahu, Qianxue Xia,
Tara Aryal

08/20/2026



U.S. DEPARTMENT
of **ENERGY**

Office of
Electricity

GRID EMT

SIMULATION INITIATIVE

Intelligence Block for RE- INTEGRATE



U.S. DEPARTMENT
of **ENERGY**

Office of
Electricity

Motivation

- EMT simulations show slow nature (e.g., seconds of 100s inverters detailed model simulations takes days!).
- One-shot reordering is often insufficient for very large matrices.
- Modular matrix detection and reordering framework are needed.



Structure in Matrices

General Sparse Matrix

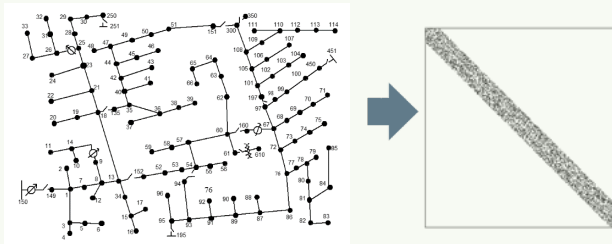
$$\begin{array}{c|cc|c} & A & x & b \\ \hline A_{11} & A_{12} & A_{13} & x_1 \\ \hline A_{21} & A_{22} & A_{23} & x_2 \\ \hline A_{31} & A_{32} & A_{33} & x_3 \\ \hline \end{array} = \begin{array}{c} b_1 \\ b_2 \\ b_3 \end{array}$$

- Block Diagonal (BD)
- Bordered Block Diagonal (BBD)
- Block Tri Diagonal (BTD)

Power system topology results in different matrix pattern

Intelligence Block

BTD Matrix

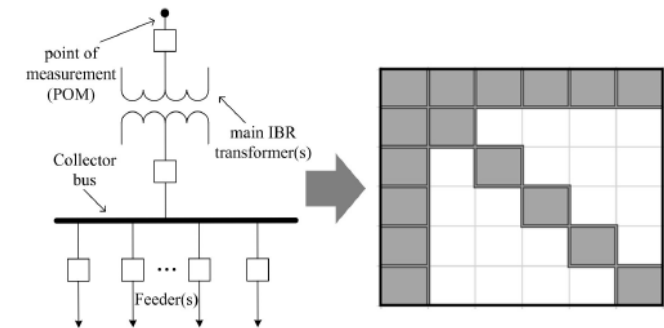


Matrix Structure based on the network topology

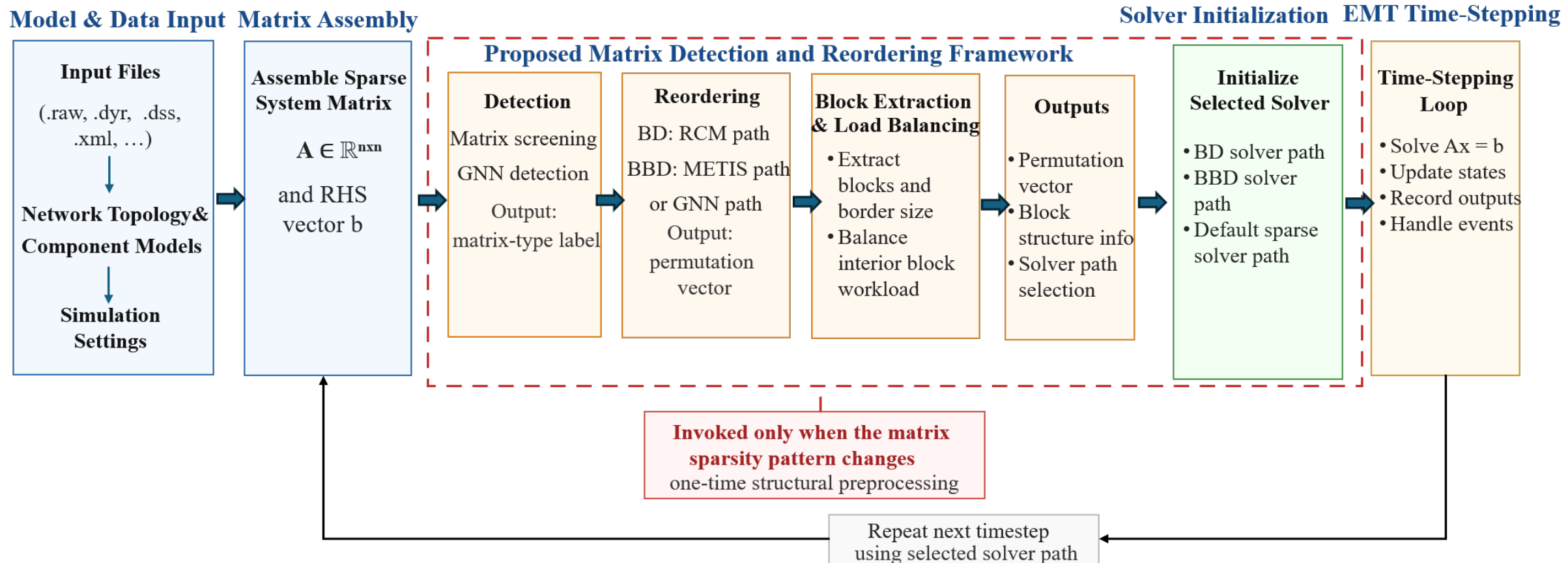
BD Matrix

$$\begin{array}{c|cc|c} & & & \\ \hline A_{11} & 0 & 0 & x_1 \\ \hline 0 & A_{22} & 0 & x_2 \\ \hline 0 & 0 & A_{33} & x_3 \\ \hline \end{array} = \begin{array}{c} b_1 \\ b_2 \\ b_3 \end{array}$$

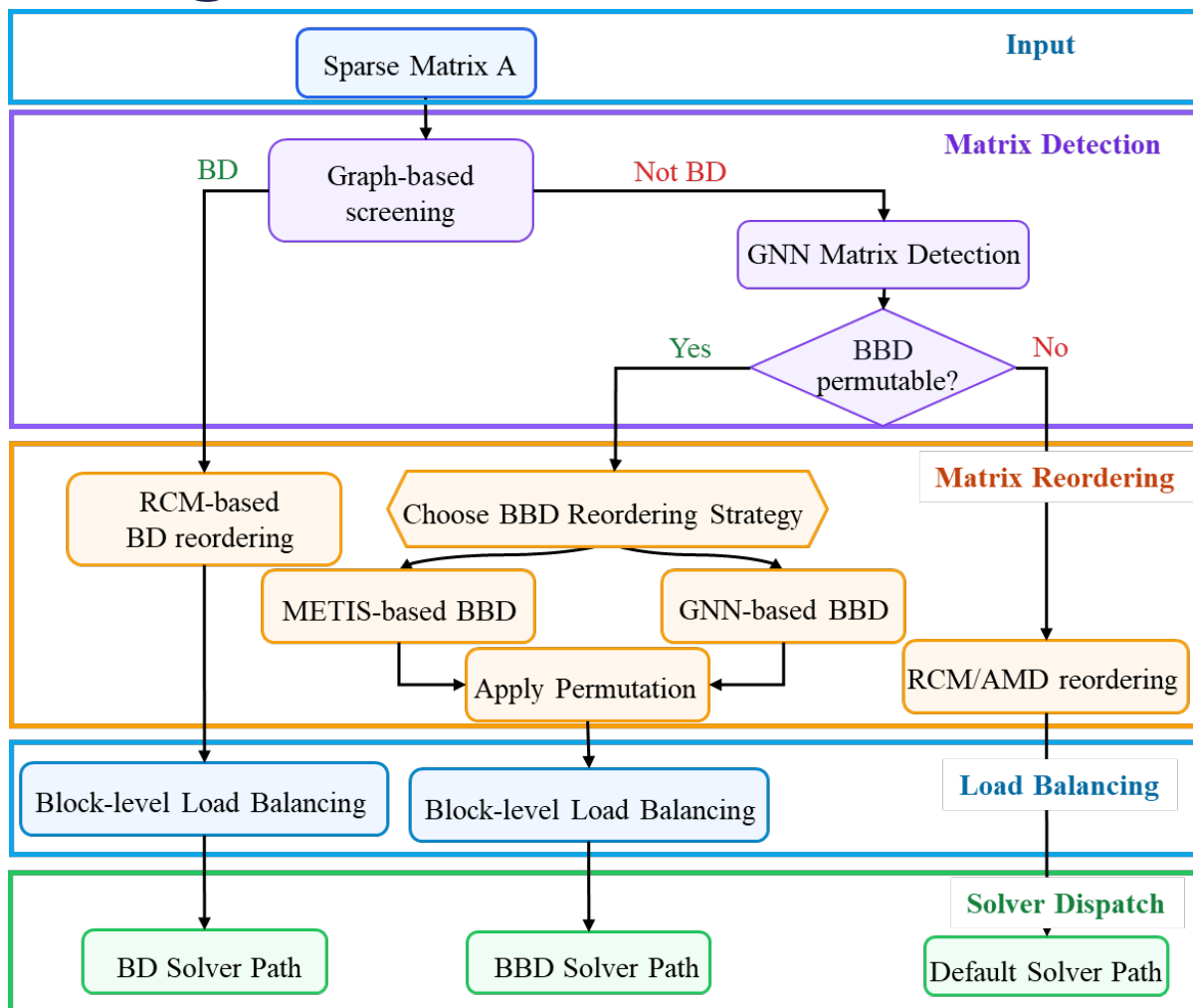
BBD Matrix



Structure-aware Runtime Path for EMT Sparse Linear Systems



Intelligence Block



Runtime pipeline

Matrix detection

Option 1: Connected path test

Option 2: GNN-matrix detection(graph classification)

Output: classify matrix structural class for solver dispatch.

BBD reordering options (two paths):

Option 1: METIS-based BBD

Option 2: GNN-based BBD

Output: reordered matrix and permutation vector

Load Balancing:

Option 1: symbolic cost balancing

Option 2: matrix size balancing

Output: grouped diagonal blocks for parallelization



U.S. DEPARTMENT
of **ENERGY**

Office of
Electricity

Metrics Used for Performance Comparison

Metric group	Metric	Meaning
Structure quality	Number of blocks	Shows how the reordered matrix is divided for parallel solution.
	Border size and border ratio	Measures the separator size. A smaller border usually means less coupling between blocks.
	Largest block size and largest block ratio	Shows load balance. A smaller largest block can reduce bottlenecks.
Online performance	Mean solver time	Average linear solver time per timestep, excluding the first recorded timestep.
	Mean total timestep time	Average full timestep runtime, excluding the first recorded timestep. It includes the solver and timestep-update work.
Offline overhead	Matrix processing time	One-time cost for detection, reordering, permutation, reindexing, and related matrix updates.
	Solver setup time	One-time cost to build and initialize the solver using the reordered matrix.



Performance Comparison: IEEE-3900 Bus System & IEEE-118 Bus System with Generation

Case	Matrix size	Metis	Framework	Key structural gain	Mean solve time	Matrix processing time	Solver setup time
IEEE-3900	93,700	BBD	BD	Border 442 → 0	-97.9% (1287ms-26ms)	-38.9%(-365.6ms-223.5ms)	-52.7% (4886.8ms-2310.8ms)
IEEE-118 + generation	5,376	BBD	BD	Border 40 → 0	-96.7% (44ms-1ms)	+3.4%	-99.9% (9323ms-0.1ms)

The biggest gains appear when the framework preserves a natural matrix pattern instead of forcing a BBD.

The framework helps for two different reasons:
1. smaller separators for BBD cases
2. zero-border preservation for BD cases.

BBD: Block Bordered Diagonal
GNN: Graph Neural Network



U.S. DEPARTMENT
of **ENERGY** | Office of
Electricity

GRID EMT

SIMULATION INITIATIVE

BTD Solver

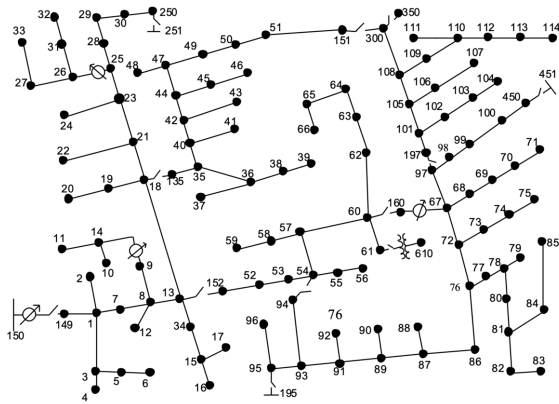


U.S. DEPARTMENT
of **ENERGY**

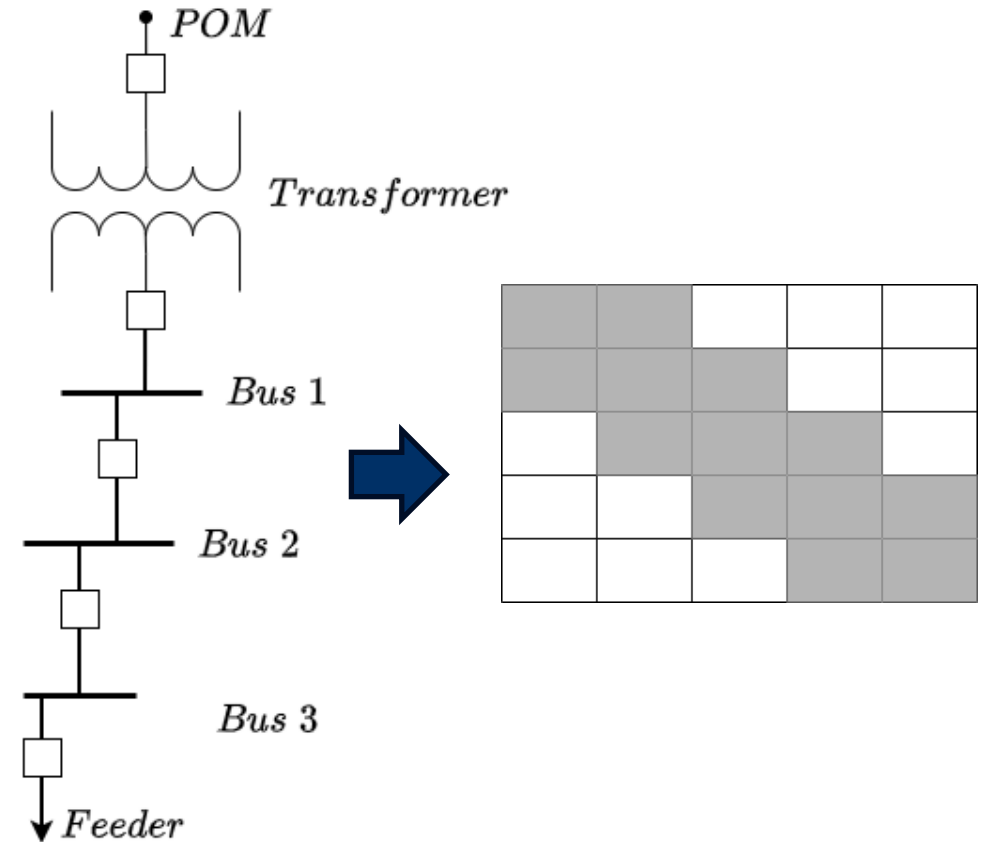
Office of
Electricity

BTD Solver: Motivation

- Parallelization -> Simulation speed up.
- Chain/grid type networks-> Block Tridiagonal (BTD) matrix.
- Parallel solve for $Ax=b$.
- Radial distribution networks result in BTD matrices.



123 bus radial distribution system



BTD Solver

Core#1

$$\begin{bmatrix} A_1 & C_1 \end{bmatrix} \begin{bmatrix} x_1 \end{bmatrix} = \begin{bmatrix} b_1 \end{bmatrix}$$

Core#2

$$\begin{bmatrix} B_2 & A_2 & C_2 \end{bmatrix} \begin{bmatrix} x_2 \end{bmatrix} = \begin{bmatrix} b_2 \end{bmatrix}$$

Core#3

$$\begin{bmatrix} B_3 & A_3 & \ddots \end{bmatrix} \begin{bmatrix} x_3 \end{bmatrix} = \begin{bmatrix} b_3 \end{bmatrix}$$

Core#n-1

$$\begin{bmatrix} \ddots & \ddots & \ddots & C_{n-1} \end{bmatrix} \begin{bmatrix} x_{n-1} \end{bmatrix} = \begin{bmatrix} b_{n-1} \end{bmatrix}$$

Core#n

$$\begin{bmatrix} B_n & A_n \end{bmatrix} \begin{bmatrix} x_n \end{bmatrix} = \begin{bmatrix} b_n \end{bmatrix}$$

$$\begin{matrix} \times A_1^{-1} \\ \times A_2^{-1} \\ \times A_3^{-1} \\ \times A_{(n-1)}^{-1} \\ \times A_n^{-1} \end{matrix} \rightarrow \begin{bmatrix} I & V_1 \\ W_2 & I & V_2 \\ \vdots & \ddots & \ddots & \ddots \\ W_n & I \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix}$$

$$\begin{bmatrix} V_1^{(t)} & & & x_1^{(t)} & g_1^{(t)} \\ I & V_1^{(m)} & & x_1^{(m)} & g_1^{(m)} \\ & V_1^{(b)} & & x_1^{(b)} & g_1^{(b)} \\ W_2^{(t)} & & V_2^{(t)} & x_2^{(t)} & g_2^{(t)} \\ W_2^{(m)} & I & V_2^{(m)} & x_2^{(m)} & g_2^{(m)} \\ W_2^{(b)} & & V_2^{(b)} & x_2^{(b)} & g_2^{(b)} \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ W_n^{(t)} & & & x_n^{(t)} & g_n^{(t)} \\ W_n^{(m)} & & I & x_n^{(m)} & g_n^{(m)} \\ W_n^{(b)} & & & x_n^{(b)} & g_n^{(b)} \end{bmatrix}$$

1. **Partitioning:** Divide the system into n independent sub blocks

2. **Calculate SPIKES:** Perform $\times A_i^{-1}$ (using LU) at each block to calculate

$$\begin{bmatrix} W_i^{(t)} \\ W_i^{(m)} \\ W_i^{(b)} \end{bmatrix} \begin{bmatrix} x_{i-1}^{(b)} + x_i^{(m)} + x_{i+1}^{(t)} \end{bmatrix} = \begin{bmatrix} g_i^{(t)} \\ g_i^{(m)} \\ g_i^{(b)} \end{bmatrix}$$

$$\begin{bmatrix} I & V_1^{(b)} & & \text{spikes} \\ W_2^{(t)} & I & V_2^{(t)} & \\ W_2^{(b)} & & I & V_2^{(b)} \\ \vdots & \ddots & \ddots & \ddots \\ W_{n-1}^{(t)} & & I & V_{n-1}^{(t)} \\ W_{n-1}^{(b)} & & & I & V_{n-1}^{(b)} \\ W_n^{(t)} & & & I \end{bmatrix} \begin{bmatrix} x_1^{(b)} \\ x_2^{(t)} \\ x_2^{(b)} \\ \vdots \\ x_{n-1}^{(t)} \\ x_{n-1}^{(b)} \\ x_n^{(t)} \end{bmatrix} = \begin{bmatrix} g_1^{(b)} \\ g_2^{(t)} \\ g_2^{(b)} \\ \vdots \\ g_{n-1}^{(t)} \\ g_{n-1}^{(b)} \\ g_n^{(t)} \end{bmatrix}$$

3. **Segment SPIKES:** Split SPIKES into top, middle and bottom sections based on the bandwidth of neighbouring SPIKES

$$\begin{bmatrix} W_i^{(t)} \\ W_i^{(m)} \\ W_i^{(b)} \end{bmatrix} \begin{bmatrix} x_{i-1}^{(b)} + x_i^{(m)} + x_{i+1}^{(t)} \end{bmatrix} = \begin{bmatrix} g_i^{(t)} \\ g_i^{(m)} \\ g_i^{(b)} \end{bmatrix}$$

4. **Boundary spikes:** Extract the top and bottom sections of spikes that connect i^{th} block to neighbour blocks

5. **Reduced system matrix:** Form the reduced system matrix from the boundary spikes extracted from each block and **Solve Sequentially**

6. **Back Substitution:** Get the values of x from neighbour blocks from reduced system solve and solve for the middle spike



U.S. DEPARTMENT
of **ENERGY**

Office of
Electricity

Demo – Example BTD Matrix

Diagonal
Blocks

Left Spike

Spike Bandwidth

Right
Spike

Representative 12727 x 12727 BTD matrix section



U.S. DEPARTMENT
of **ENERGY**

Office of
Electricity

Impact Expected & Results

- Validated on synthetic BTD matrices generated with varying parameters and benchmarked against KLU library.
- The BTD solve time depends highly on the **size of blocks**, **sparsity of the blocks** and the **off-diagonal bandwidths***.
- **Small Scale (N ~ 2,000)**: SPIKE-BTD shows mixed performance relative to KLU, yielding modest speedups (**up to 1.86x**) in higher-sparsity cases but suffering penalties (down to 0.50x) when spike widths are large.
- **Medium Scale (N ~ 10,000)**: SPIKE-BTD consistently outperforms KLU across all configurations, delivering steady speedups ranging from **1.16x to 4.05x**.
- **Large Scale (N ~ 100,000)**: SPIKE-BTD demonstrates superior scalability with speedups up to **11.03x**, significantly outperforming KLU.
- The accuracy of solver was validated on IEEE 123 bus system.

* V. Sahu, P. Marthi, S. Debnath, "Evaluating the SPIKE Algorithm on Synthetic Block-Tridiagonal Matrices to Accelerate Radial Network EMT Solvers", accepted, IECON2026



U.S. DEPARTMENT
of **ENERGY**

Office of
Electricity

Next Steps

- **Support more matrix types**
 - Add BTD and other block-structured matrix patterns.
- **Add GPU-aware load balancing**
 - Map matrix blocks and border regions to GPU execution groups.
 - Improve GPU solver utilization and reduce idle time.
- **Expand validation cases**
 - Evaluate robustness across different matrix sizes and structures.
- **Validation, evaluation, and demonstration of BTD solver for real-world power system examples**
 - Large loads and system of such sub-systems



U.S. DEPARTMENT
of **ENERGY**

Office of
Electricity

Thank You

Acknowledgement: The authors thank U.S. Department of Energy (DOE) Office of Electricity (OE) Advanced Grid Modeling (AGM) and the EMT initiative for their invaluable guidance and kind support in this work!

Legal Disclaimer: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

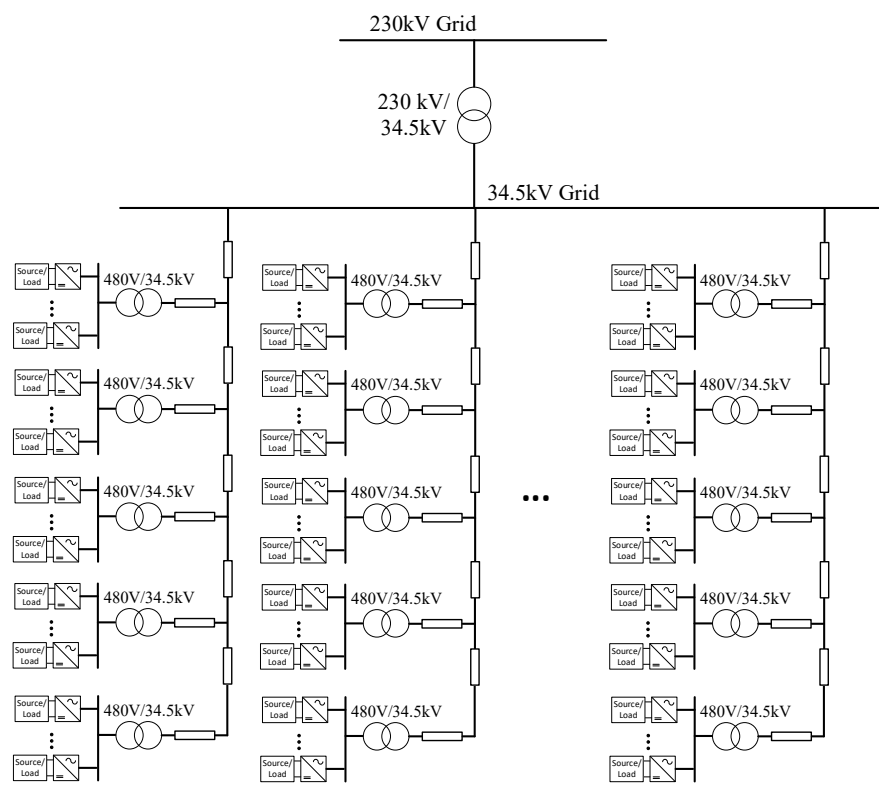




U.S. DEPARTMENT *of* ENERGY

Office of Electricity

Performance Comparison: 12-Feeder IBL



IBL Plant Overview
12 feeders, each with 25 IBL units

5723 × 5723 “A” matrix

Category	Metric	METIS	Framework	Relative change
Structure quality	Border size	112	6	-94.6%
Online performance	Steady Total Time (ms)	41.7029	38.7358	-7.1%

GNN-based BBD achieves **better reordering quality** than **METIS** with **comparable offline overhead**.

Performance Comparison: IEEE-3900 Bus System & IEEE-118 Bus System with generation

Case	Matrix size	Metis	Framework	Key structural gain	Mean solve time	Matrix processing time	Solver setup time
IEEE-3900	93,700	BBD	BD	Border 442 → 0	-97.9% (1287ms-26ms)	-38.9%(-365.6ms-223.5ms)	-52.7% (4886.8ms-2310.8ms)
IEEE-118 + generation	5,376	BBD	BD	Border 40 → 0	-96.7% (44ms-1ms)	+3.4%	-99.9% (9323ms-0.1ms)

The biggest gains appear when the framework preserves a natural matrix pattern instead of forcing a BBD.

The framework helps for two different reasons:

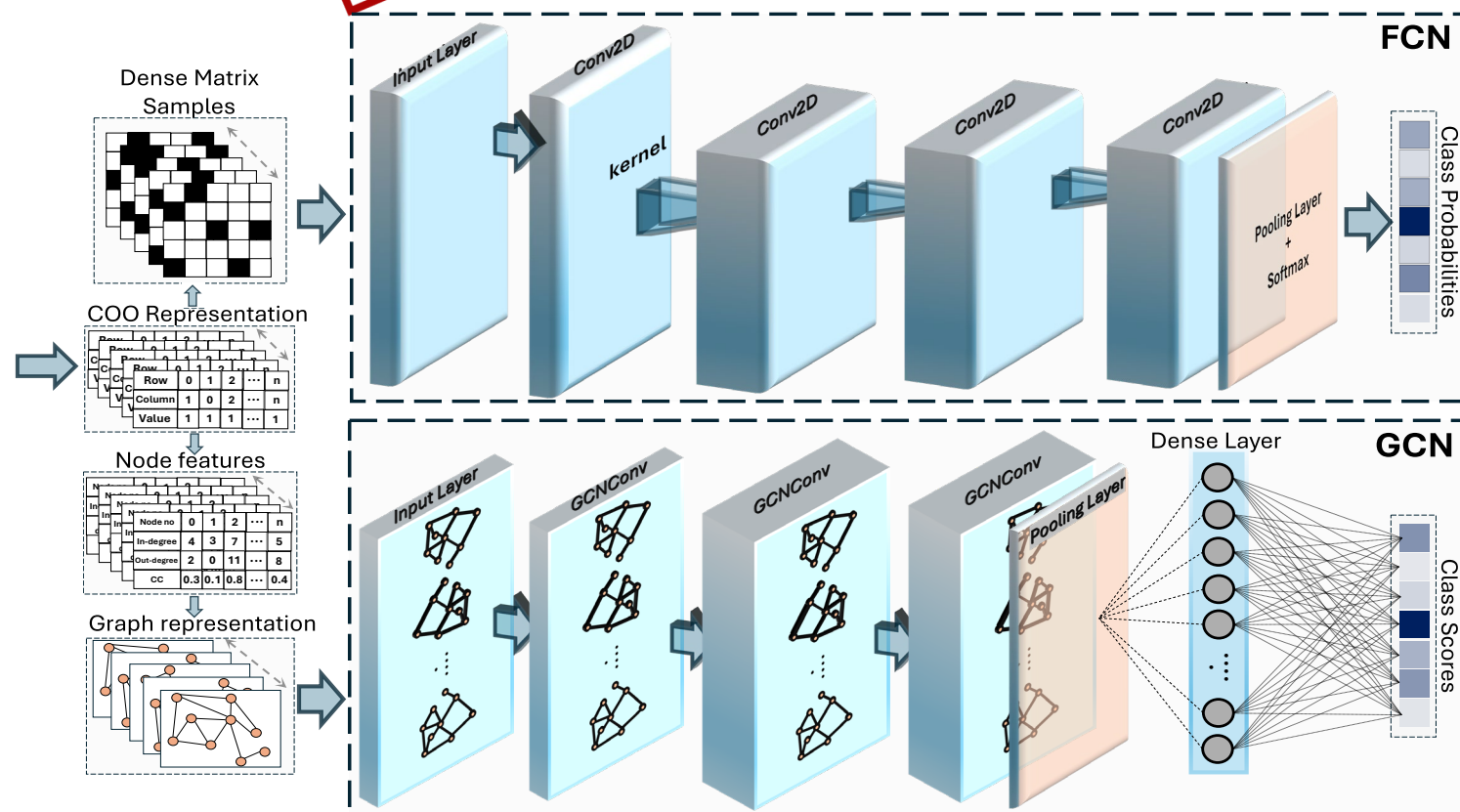
- 1.smaller separators for BBD cases**
- 2. zero-border preservation for BD cases.**

BBD: Block Bordered Diagonal
GNN: Graph Neural Network

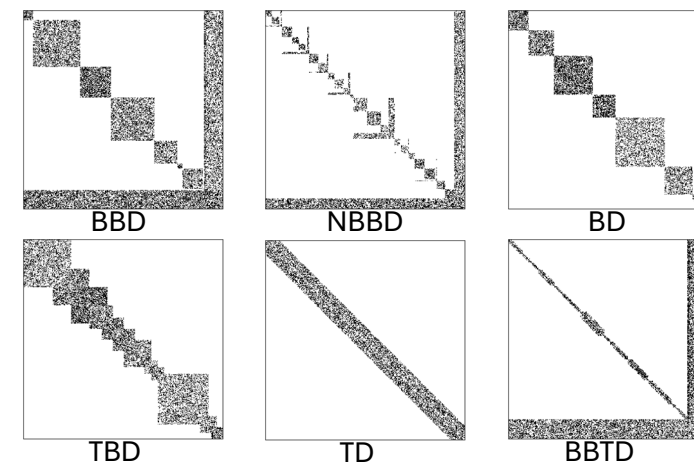
Matrix Pattern Detection for Optimal Solver Selection

Can we detect the optimal form of the matrix before reordering?

FCN: Fully Convolutional Network
GCN: Graph Convolutional Network

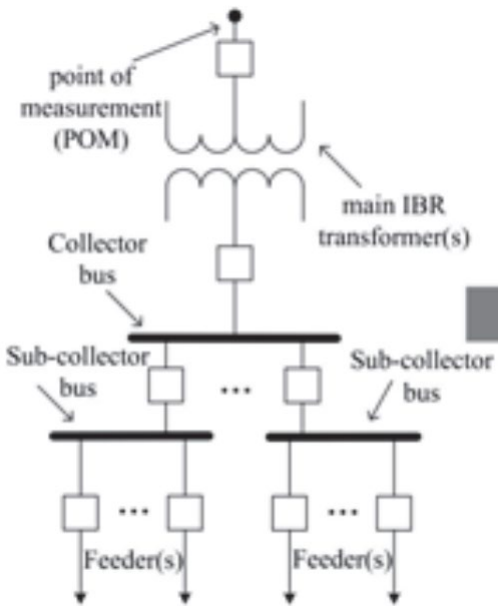


Matrix patterns classified

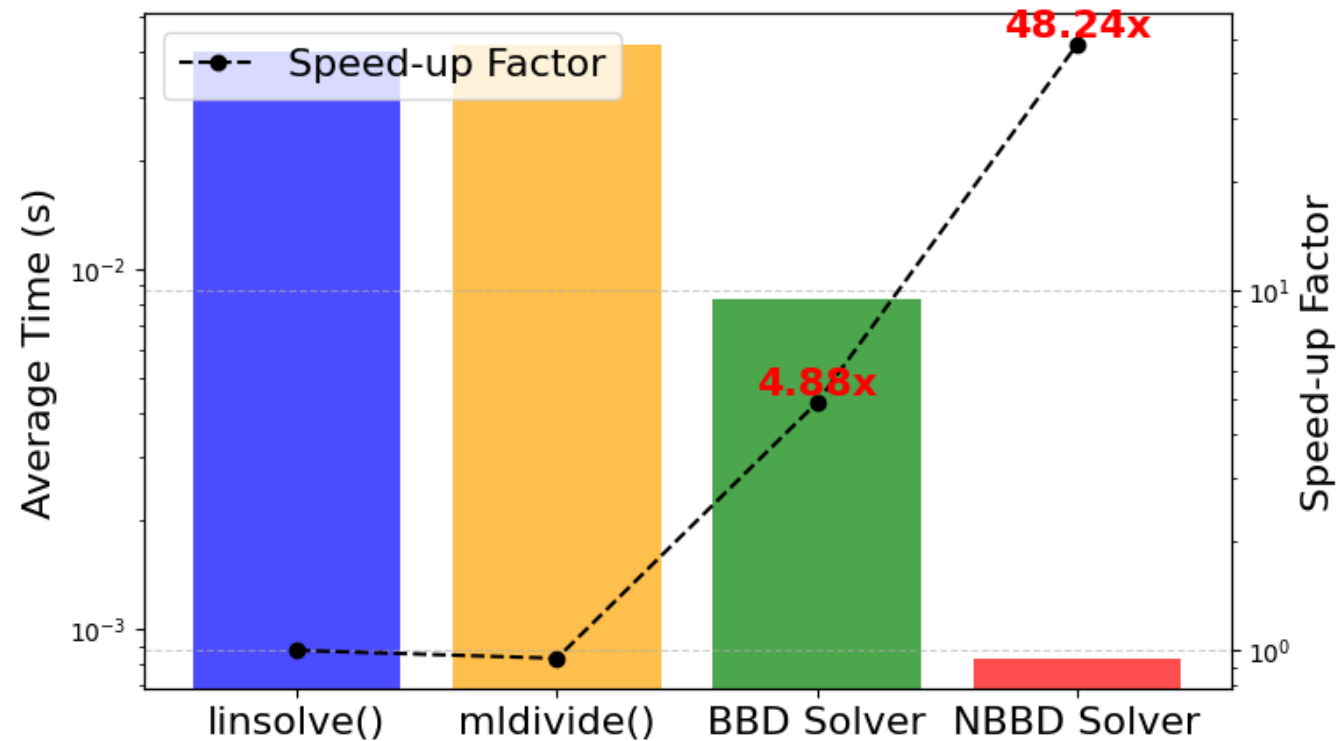


- Block Diagonal (BD)
- Bordered Block Diagonal (BBD)
- Nested Bordered Block Diagonal (NBBD)
- Tri Diagonal (TD)
- Tri Block Diagonal (TBD)
- Bordered Block Tri Diagonal (BBTD)

Successfully detect 191,616x191,616 matrix (4992 bus transmission system with 384 PV plants)

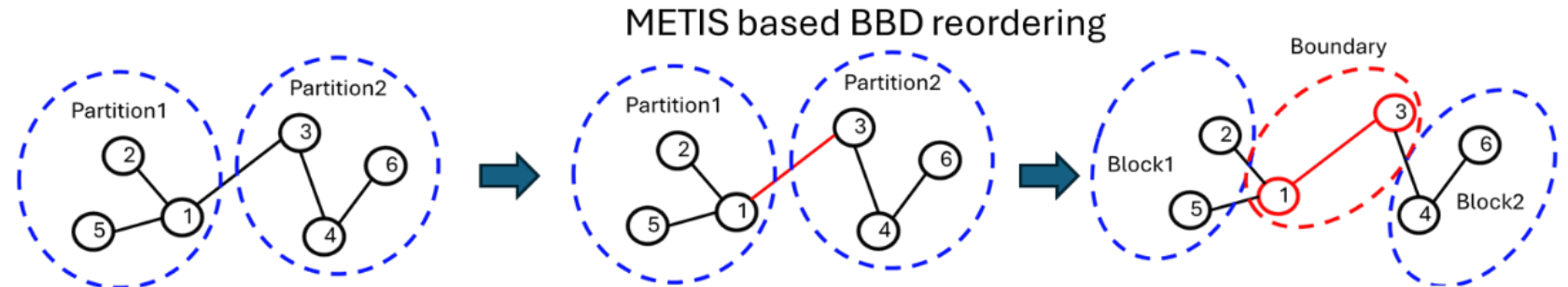
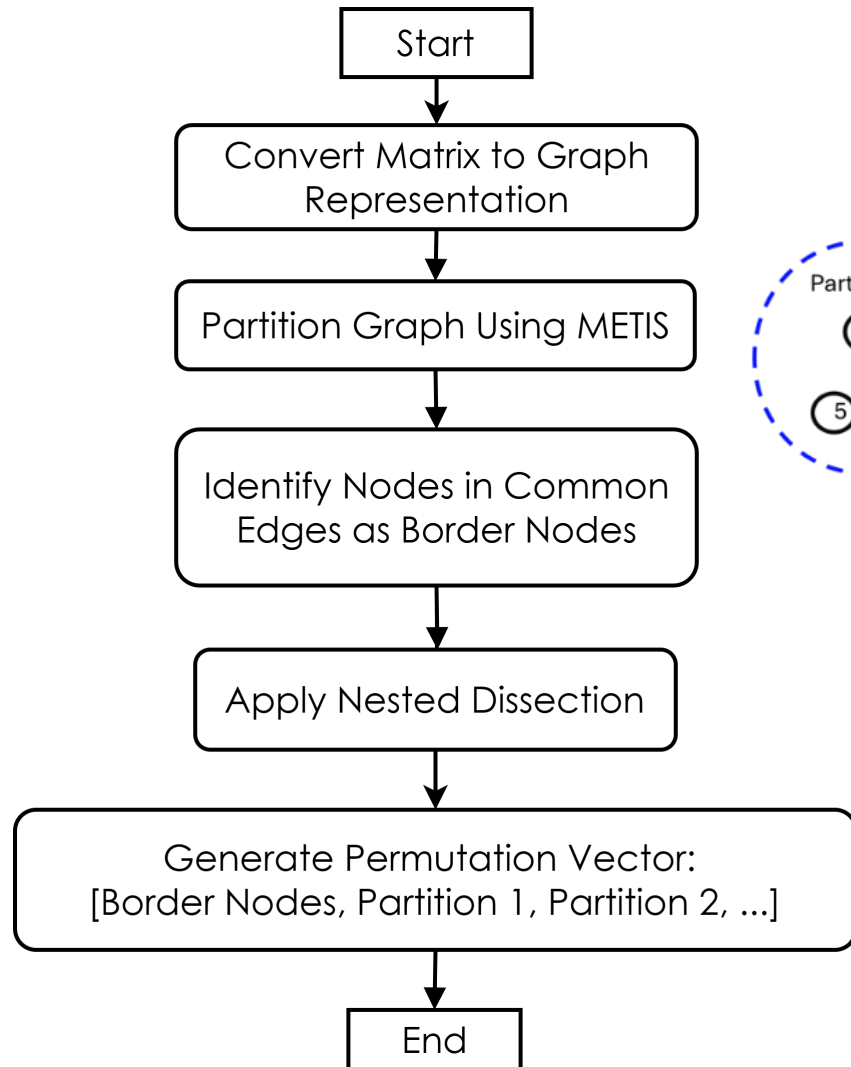


Multiple collector buses with feeders



Pre-identifying the sparse pattern enables efficient reordering and selection of optimal solvers, reducing computational demands in EMT simulations.

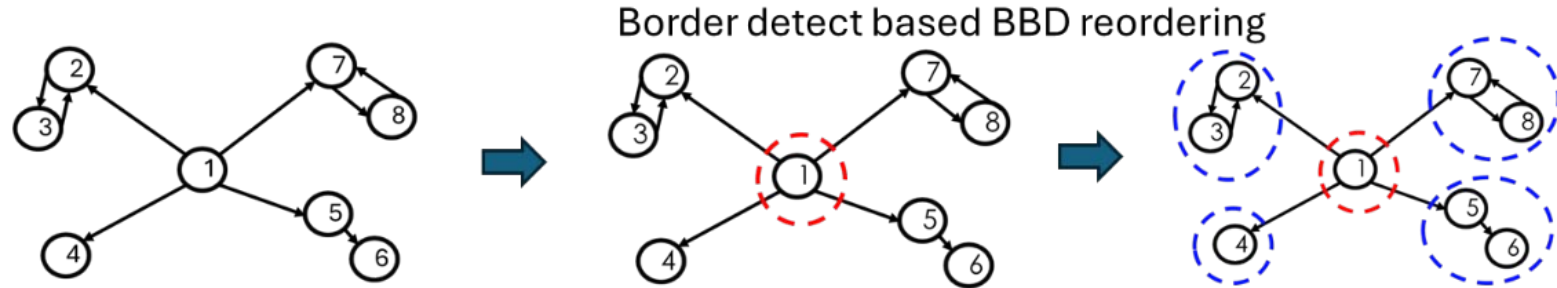
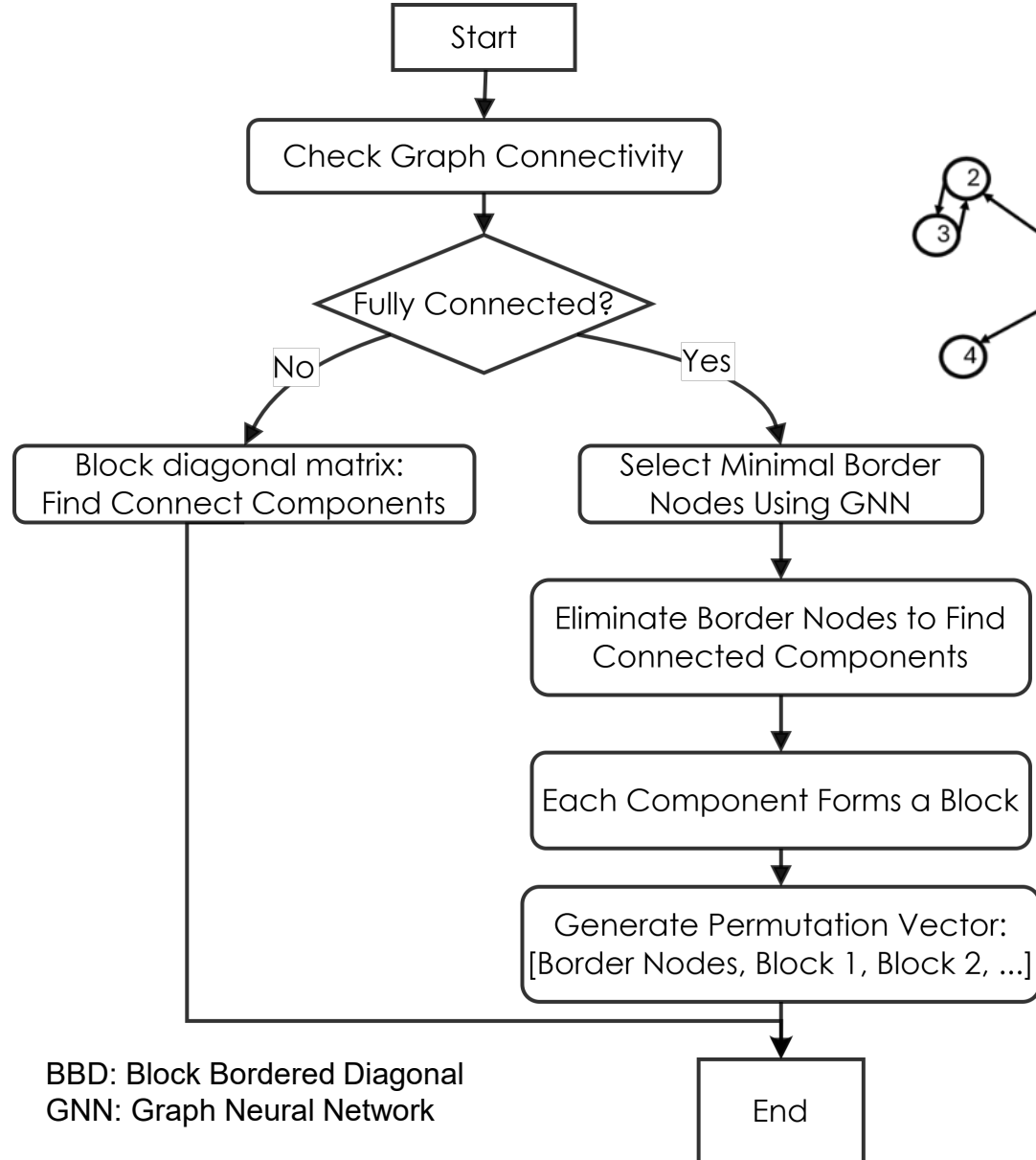
Benchmark: METIS BBD Reordering



Challenges:

- Specifying the number of partitions manually.
- Finding the optimal partitioning is an NP-hard problem.

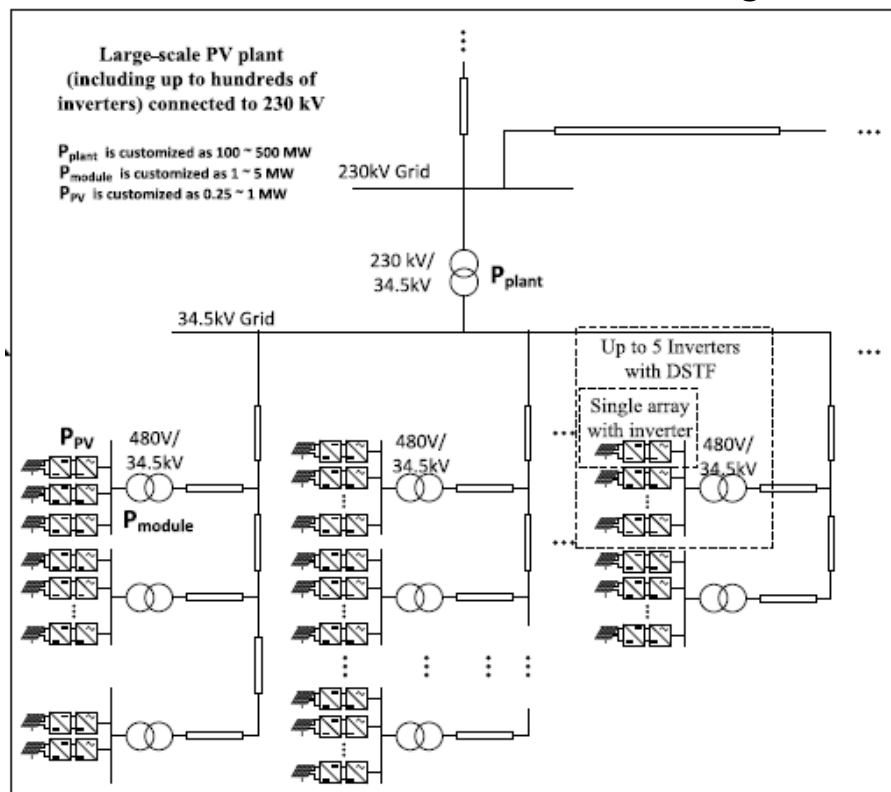
GNN-Based Sparse Matrix BBD Reordering



Key Advantages:

- Eliminates the need for **repeated computations** when scaling CPU cores (e.g., from 3 to 5 CPUs).
- Handle **nonsymmetric matrix** well
- Provides **better** and **faster** reordering solutions for improved solver performance.

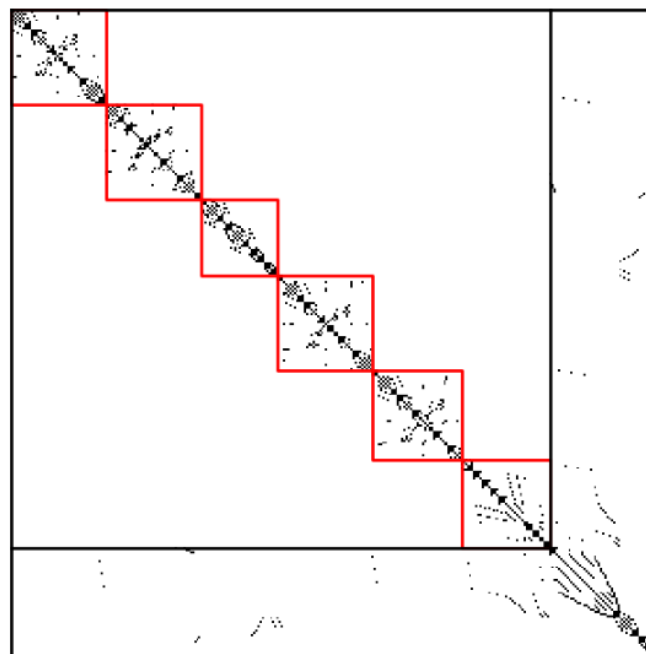
1. reorder time; 2. solve time using same solver



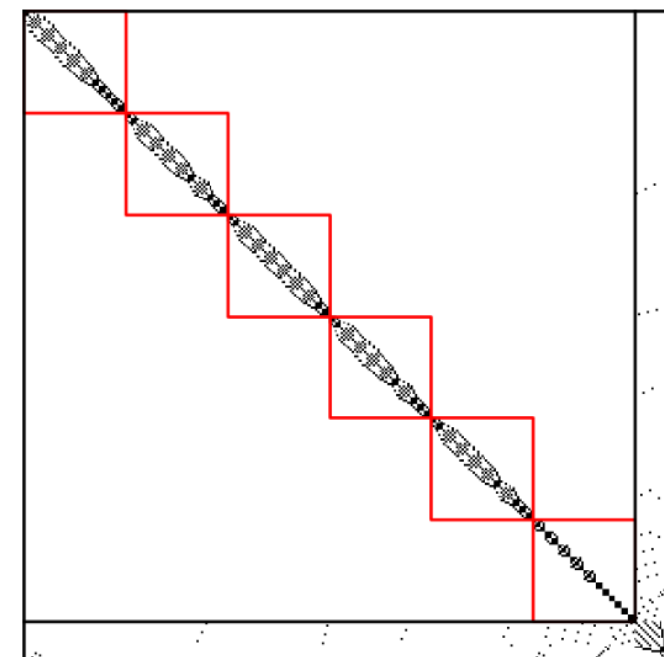
PV Plant Overview

5 feeders, each with 25 PV systems.

306×306 “A” matrix



Metis BBD reordering



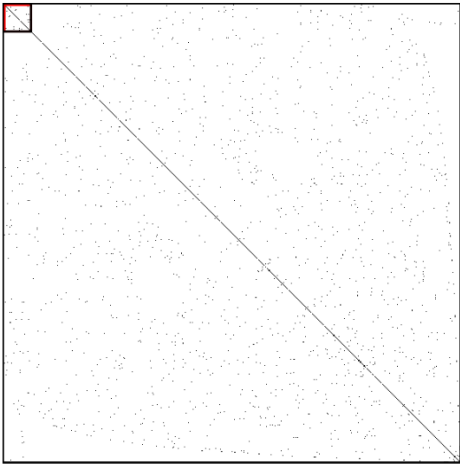
GNN BBD reordering

Method	Process Time (s)	Solve Time (s)	Performance Gain
METIS BBD	0.0036	0.0018	Baseline
GNN BBD	0.0030	0.0010	~20% faster

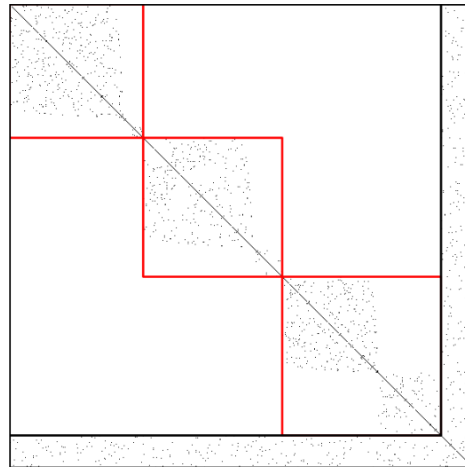
Performance Comparison for Large System

8052×8052 Generated Matrix

Metis BBD cannot partition more than 3.

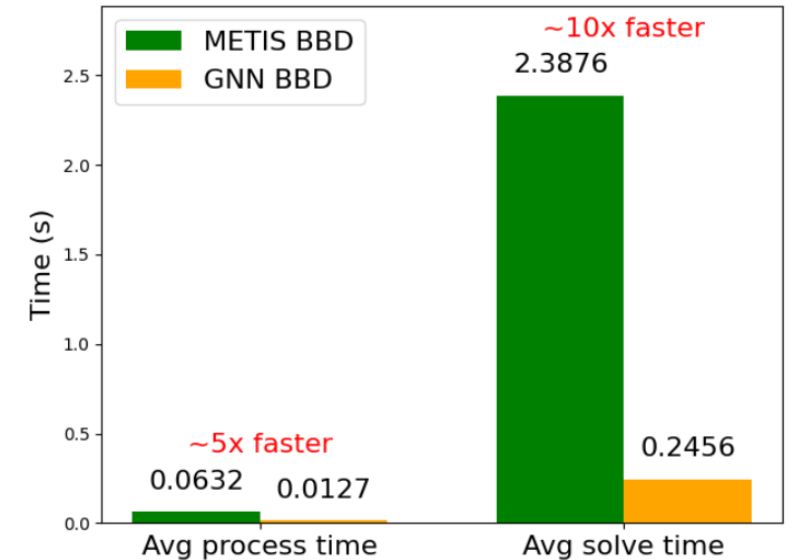


Common nodes = 7553
Blocks: 4, 21, 474



Common nodes = 560
Block Sizes: 1, 511, 353,
1901, 1971, 1663, 1091, 1

SuiteSparse matrix collection data (100 matrices)



	METIS BBD	GNN BBD	Improvement
Process time	0.6218 s	0.0276 s	~22x faster
Solve time	113.4193 s	3.0913 s	~37x faster

- Significant improvements in processing and solving time for large sparse matrices
- Can handle highly unsymmetric matrix that METIS BBD cannot.

- **Add MPI-aware load balancing**
 - Assign blocks to MPI ranks based on estimated workload.
 - Reduce bottlenecks from oversized blocks or uneven partitions.