A Simple and Efficient Parallel Laplacian Solver

Yibin Zhao
University of Toronto

Joint work with Sushant Sachdeva

Solving Linear Systems

$$Ax = b$$

Given A, b find x .

If A is a Laplacian, it's called a Laplacian linear system

[Spielman-Teng '04] *Laplacian linear systems can be solved in nearly linear time*

[Peng-Spielman '13] *Laplacian linear systems can be solved in parallel with nearly linear work and polylog depth*

Laplacians

Symmetric $n \times n$ matrix, associated with a weighted, undirected graph

$$G = (V, E, w) \qquad n = |V|, \;\; m = |E|, \;\; w : E \mapsto \mathbb{R}_+$$

$$L = D - A$$

Quadratic form

$$L = \sum_{(u,v) \in E} w_{uv} b_{uv} b_{uv}^T$$

$$b_{uv} = e_u - e_v$$

Laplacians

Symmetric $n \times n$ matrix, associated with a weighted, undirected graph

$$G = (V, E, w) \qquad n = |V|, \quad m = |E|, \quad w : E \mapsto \mathbb{R}_+$$

$$L = D - A$$

Quadratic form

$$L = BWB^T$$

B: Incident matrix for the (multi-)graph

Application of Laplacian Solvers

PDEs via Finite Element Methods **[Str85, BHV08]**

IPM for Optimization **[KRS15, CMSV16]**

Learning on graph **[ZGL03, ZBLW04, BMN04]**

Flow algorithms **[DS08, CKMST11, KMP12, LS14, DGGLPSY22, LKLPGS22]**

Graph sparsification **[SS08, LKP12]**

Related Works

	Work	Depth	Technique
[Peng-Spielman ‘13]	$\widetilde{O}(m)$	$O(\log^c n)$	Repeated Squaring + Expander
[Lee-Peng-Spielman ‘15, Kyng-L-P-S-Sachdeva ‘16]	$\widetilde{O}(m)$	$O(\log^6 n \log^4 \log n)$	Block Cholesky factorization + Sparsification + Expander

The work and depth are for constant approximation. An additional $O(\log 1/\epsilon)$ factor for both work and depth is required for ϵ-approximate solution.

Related Works

	Work	Depth	Technique
[Peng-Spielman ‘13]	$\widetilde{O}(m)$	$O(\log^c n)$	Repeated Squaring + Expander
[Lee-Peng-Spielman ‘15, Kyng-L-P-S-Sachdeva ‘16]	$\widetilde{O}(m)$	$O(\log^6 n \log^4 \log n)$	Block Cholesky factorization + Sparsification + Expander

The work and depth are for constant approximation. An additional $O(\log 1/\epsilon)$ factor for both work and depth is required for ϵ-approximate solution.

Related Works

	Work	Depth	Technique
[Peng-Spielman '13]	$\widetilde{O}(m)$	$O(\log^c n)$	Repeated Squaring + Expander
[Lee-Peng-Spielman '15, Kyng-L-P-S-Sachdeva '16]	$\widetilde{O}(m)$	$O(\log^6 n \log^4 \log n)$	Block Cholesky factorization + Sparsification + Expander
[Kyng-Sachdeva '16]	$\widetilde{O}(m)$		Sparse Cholesky factorization

The work and depth are for constant approximation. An additional $O(\log 1/\epsilon)$ factor for both work and depth is required for ϵ-approximate solution.

Related Works

	Work	Depth	Technique
[Peng-Spielman '13]	$\widetilde{O}(m)$	$O(\log^c n)$	Repeated Squaring + Expander
[Lee-Peng-Spielman '15, Kyng-L-P-S-Sachdeva '16]	$\widetilde{O}(m)$	$O(\log^6 n \log^4 \log n)$	Block Cholesky factorization + Sparsification + Expander
[Kyng-Sachdeva '16]	$\widetilde{O}(m)$		Sparse Cholesky factorization
[Sachdeva-Z '22]	$\widetilde{O}(m)$	$O(\log^2 n \log \log n)$	Sparse block Cholesky factorization

The work and depth are for constant approximation. An additional $O(\log 1/\epsilon)$ factor for both work and depth is required for ϵ-approximate solution.

Background

Positive Semi-Definite (PSD) Matrices

A is PSD if A is symmetric and for all x,

$$x^\top A x \geq 0$$

Laplacians are PSD

Define a natural norm

$$\|x\|_A = \sqrt{x^\top A x}$$

Approximately Solving a System

A ϵ-approximate solution to $Lx = b$ is $\widetilde{x}$ s.t.

$$\|\widetilde{x} - x\|_L \leq \epsilon \|x\|_L$$

Recall $\|x\|_A = \sqrt{x^\top A x}$

This suffices for most applications

Parallel Model

Work: The running time on a single node

Depth: The running time given unlimited number node with shared memory

The “longest dependency” in the computation

Concurrent Read Exclusive Write (CREW) PRAM

Parallel Primitives

Linear Operator: One can apply a linear operator with m nonzero entries to a column vector of compatible dimension in $O(m)$ work and $O(\log m)$ depth

Weighted Sampling [HS19]: For n items, there is an algorithm that takes $O(n)$ work and $O(\log n)$ depth preprocessing and $O(1)$ work and depth each query

Graph representation [BM10]: One can transform between an adjacency list representation and edge list representation of a (multi-)graph in $O(m)$ work and $O(\log m)$ depth

Our Result

[Sachdeva-Z '22] *Can find an* ϵ*-approximate solution to a Laplacian system in*

$$O(m \log^3 n \log\log n \log(1/\epsilon))$$ work and

$$O(\log^2 n \log\log n \log(1/\epsilon))$$ depth

Approximating PSD Matrices

If A is PSD, we write

$$A \succcurlyeq 0$$

This induces a partial order

$$A \succcurlyeq B \qquad \text{if} \qquad A - B \succcurlyeq 0$$

Spectral approximation

$$A \approx_\epsilon B \qquad \text{if} \qquad e^\epsilon B \succcurlyeq A \succcurlyeq e^{-\epsilon} B$$

Iterative Refinement

To solve $Ax = b$ approximately,

Find a preconditioner B such that $A \approx_{O(1)} B$ and easy to invert

Solve $B^{-1}Ax = B^{-1}b$ by

$$x^{(i+1)} \leftarrow x^{(i)} - \frac{1}{2}B^{-1}(Ax^{(i)} - b)$$

Block Cholesky Factorization

$$L = \begin{array}{cc} & \begin{array}{cc} F & T \end{array} \\ & \begin{pmatrix} A & B \\ B^T & C \end{pmatrix} \begin{array}{c} F \\ T \end{array} \end{array}$$

$$V = F \cup T$$

Block Cholesky Factorization

$$\begin{array}{cc} & \begin{array}{cc} F & T \end{array} \\ L = & \begin{pmatrix} A & B \\ B^T & C \end{pmatrix} \begin{array}{c} F \\ T \end{array} \end{array} \qquad V = F \cup T$$

$$L = \begin{pmatrix} I & 0 \\ B^\top A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & SC(L,T) \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$

Schur Complement: $SC(L,T) = C - B^\top A^{-1} B$

Block Cholesky Factorization

$$L = \begin{array}{cc} & \begin{array}{cc} F & T \end{array} \\ & \begin{pmatrix} A & B \\ B^T & C \end{pmatrix} \begin{array}{c} F \\ T \end{array} \end{array} \qquad V = F \cup T$$

$$L^+ = \begin{pmatrix} I & -A^{-1}B \\ 0 & I \end{pmatrix} \begin{pmatrix} A^{-1} & 0 \\ 0 & SC(L,T)^+ \end{pmatrix} \begin{pmatrix} I & 0 \\ -B^\top A^{-1} & I \end{pmatrix}$$

Schur Complement: $SC(L,T) = C - B^\top A^{-1} B$

Block Cholesky Factorization

$$L = \begin{array}{cc} & \begin{array}{cc} F & T \end{array} \\ & \begin{pmatrix} A & B \\ B^T & C \end{pmatrix} \begin{array}{c} F \\ T \end{array} \end{array} \qquad V = F \cup T$$

$$L^{+} = \begin{pmatrix} I & -\boxed{A^{-1}}B \\ 0 & I \end{pmatrix} \begin{pmatrix} \boxed{A^{-1}} & 0 \\ 0 & \boxed{SC(L,T)^{+}} \end{pmatrix} \begin{pmatrix} I & 0 \\ -B^{\top}\boxed{A^{-1}} & I \end{pmatrix}$$

Schur Complement: $SC(L,T) = C - B^{\top}A^{-1}B$

Fact: Schur Complement of a Laplacian is also Laplacian

Solving Laplacian System

$$L = \begin{pmatrix} I & 0 \\ B^\top A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & SC(L,T) \end{pmatrix} \begin{pmatrix} I & A^{-1}B^\top \\ 0 & I \end{pmatrix}$$

$$\approx$$

$$L_2$$

$$\vdots$$

$$L_d$$

$$L \approx_{O(1)} U_1^\top U_2^\top \cdots U_d^\top L_d U_d \cdots U_2 U_1$$

Solving Laplacian System

$$L = \begin{pmatrix} I & 0 \\ B^\top A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & SC(L,T) \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$

$$L \approx_{O(1)} U_1^\top U_2^\top \cdots U_d^\top L_d U_d \cdots U_2 U_1$$

Apply Iterative Refinement for $O(\log \epsilon^{-1})$ iterations using this factorization as preconditioner

Solving Laplacian System

$$L = \begin{pmatrix} I & 0 \\ B^\top A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & SC(L,T) \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$

$$L \approx_{O(1)} U_1^\top U_2^\top \cdots U_d^\top L_d U_d \cdots U_2 U_1$$

Apply Iterative Refinement for $O(\log \epsilon^{-1})$ iterations using this factorization as preconditioner

1. L_d can be inverted easily. Trivial if size is constant
2. Each A_i can be inverted easily
3. Each L_i is sparse

Solving Laplacian System

$$L = \begin{pmatrix} I & 0 \\ B^\top A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & SC(L,T) \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$

$$L \approx_{O(1)} U_1^\top U_2^\top \cdots U_d^\top L_d U_d \cdots U_2 U_1$$

Apply Iterative Refinement for $O(\log \epsilon^{-1})$ iterations using this factorization as preconditioner

1. L_d can be inverted easily. Trivial if size is constant
2. Each A_i can be inverted easily
3. Each L_i is sparse

Our Algorithm

Strongly Diagonally Dominant Blocks

A matrix A is said to be α-Diagonally Dominant (DD) if

$$A_{ii} \geq \alpha \sum_{j \neq i} |A_{ij}|$$

Strongly Diagonally Dominant Blocks

A matrix A is said to be α-Diagonally Dominant (DD) if

$$A_{ii} \geq \alpha \sum_{j \neq i} |A_{ij}|$$

[LPS '15, KLPSS '16] *Can find a O(1)-DD subblock with 1/O(1) fraction size of a multi-graph in* $O(m)$ *work and* $O(\log m)$ *depth w.h.p.*

[LPS '15, KLPSS '16] *A system of O(1)-DD matrix can be solved* ϵ*-approximately in* $O(\log \epsilon^{-1})$ *iterations using iterative refinement*

For simplicity, we pick the smallest possible: 5-DD

Our Algorithm

Sparse Block Cholesky Factorization

Iterate until the remaining matrix has $O(1)$ vertices $\qquad O(\log n)$

Find a 5-DD subblock F to eliminate

Sample the Schur complement estimation created by eliminating F

Solving Laplacian System

$$L = \begin{pmatrix} I & 0 \\ B^\top A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & SC(L,T) \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$

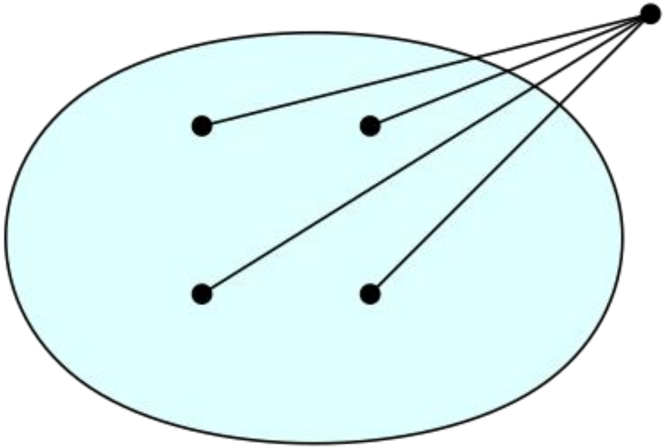
$$L \approx_{O(1)} U_1^\top U_2^\top \cdots U_d^\top L_d U_d \cdots U_2 U_1$$

Apply Iterative Refinement for $O(\log \epsilon^{-1})$ iterations using this factorization as preconditioner

1. L_d can be inverted easily. Trivial if size is constant
2. Each A_i can be inverted easily
3. Each L_i is sparse

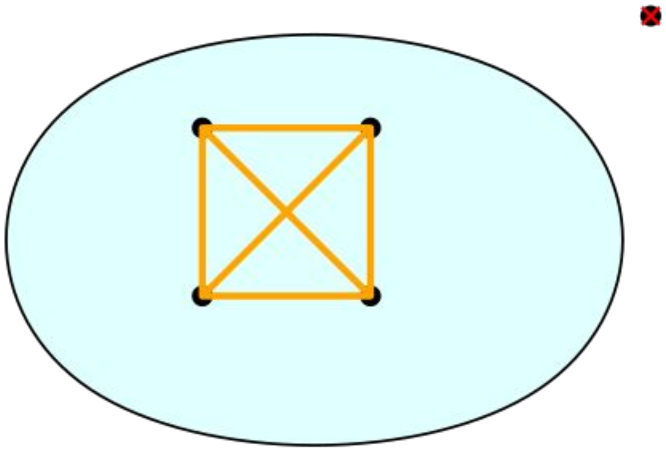
Problem of (Block) Cholesky Factorization

Fill-in phenomenon



Problem of (Block) Cholesky Factorization

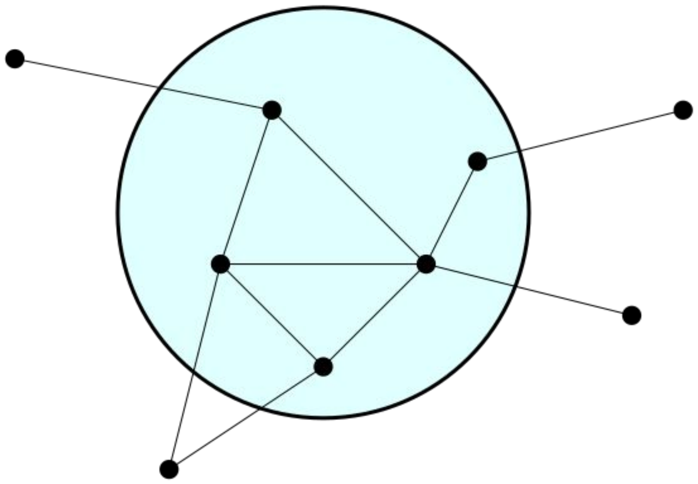
Fill-in phenomenon



Schur Complement Approximation

Approach: Random Walk Sampling

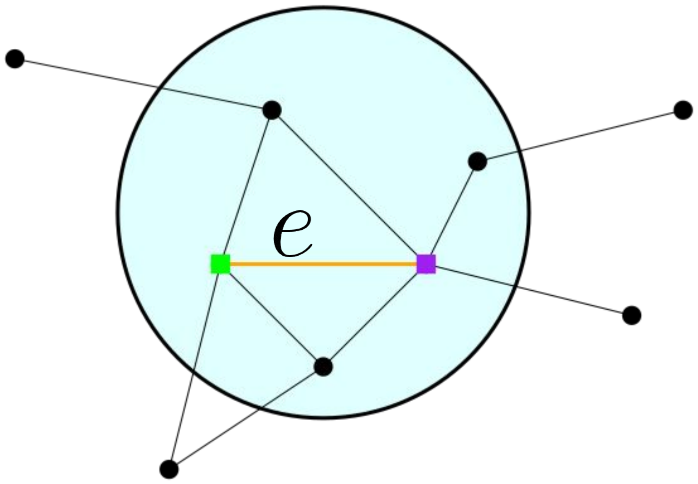
[SZ '22] *There is an **unbiased** estimation of multi-graph Schur Complement using "Terminal Random Walk" on the multi-graph.*



Schur Complement Approximation

Approach: Random Walk Sampling

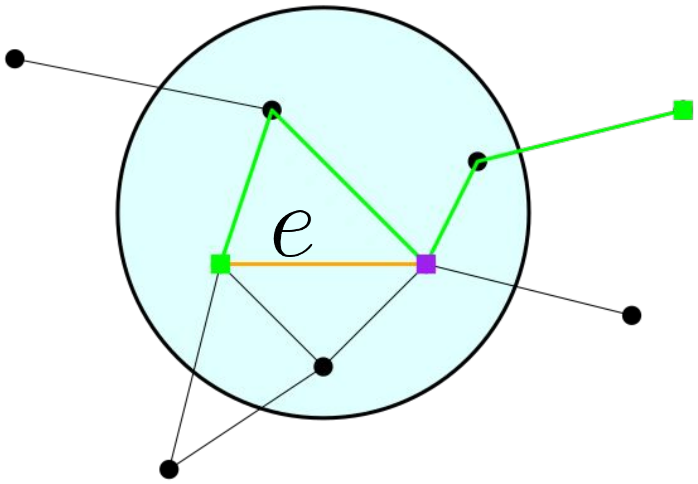
[SZ '22] *There is an **unbiased** estimation of multi-graph Schur Complement using "Terminal Random Walk" on the multi-graph.*



Schur Complement Approximation

Approach: Random Walk Sampling

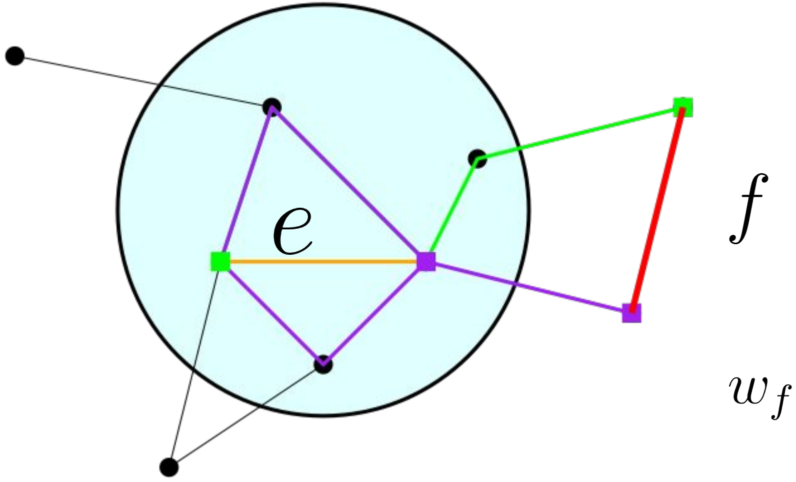
[SZ '22] *There is an* ***unbiased*** *estimation of multi-graph Schur Complement using "Terminal Random Walk" on the multi-graph.*



Schur Complement Approximation

Approach: Random Walk Sampling

[SZ '22] *There is an **unbiased** estimation of multi-graph Schur Complement using "Terminal Random Walk" on the multi-graph.*



$$w_f = \frac{1}{\sum_{l \in W(e)} 1/w_l}$$

Schur Complement Approximation

Approach: Random Walk Sampling

[SZ '22] *There is an **unbiased** estimation of multi-graph Schur Complement using "Terminal Random Walk" on the multi-graph.*

1. If eliminating a 5-DD block, each RW has length $O(\log m)$ and total length of all RWs is O(m) w.h.p.
2. The number of multi-edges in Schur is at most that of the multi-graph
3. RW samples can be carried out independently in parallel

Simple Matrix Concentration: Bernstein's inequality

For independent random symmetric matrices $X_1, \ldots X_m$ s.t.

$$\|X_i\| \leq R \quad \text{and} \quad \mathbf{E}(X_i) = 0$$

Let total variance be $\sigma^2 = \|\sum_i \mathbf{E}(X_i^2)\|$

For all $t \geq 0$

$$\mathbf{P}\left[\|\sum_i X \| \geq t\right] \leq n \cdot \exp\left(-\frac{t^2}{\sigma^2 + Rt/3}\right)$$

Simple Matrix Concentration: Bernstein's inequality

For independent random symmetric matrices $X_1, \ldots X_m$ s.t.

$$\|X_i\| \leq R \quad \text{and} \quad \mathbf{E}(X_i) = 0$$

Let total variance be $\sigma^2 = \|\sum_i \mathbf{E}(X_i^2)\|$

For all $t \geq 0$

$$\mathbf{P}\left[\| \sum_i X \| \geq t\right] \leq n \cdot \exp\left(-\frac{t^2}{\sigma^2 + Rt/3}\right)$$

Effective Resistance

The effective resistance of two distinct vertices u,v in a graph with Laplacian … is

$$b_{uv}^\top L^+ b_{uv}$$

[Klein-Randić '93] *Effective Resistance is a distance. For any vertices x,y,z,*

$$b_{xy}^\top L^+ b_{xy} \leq b_{xz}^\top L^+ b_{xz} + b_{yz}^\top L^+ b_{yz}$$

For any distinct $u, v \in T$

$$b_{uv}^\top SC(L,T)^+ b_{uv} = b_{uv}^\top L^+ b_{uv}$$

α-boundedness w.r.t. Laplacians

A multi-edge e is α-bounded w.r.t. a Laplacian L if

$$w_e b_e^\top L^+ b_e \le \alpha$$

α-boundedness w.r.t. Laplacians

A multi-edge e is α-bounded w.r.t. a Laplacian L if

$$w_e b_e^\top L^+ b_e \le \alpha$$

If all multi-edges in a graph is α-bounded w.r.t. its Laplacian L, then for each sampled multi-edge

$$b_f^\top SC(L,T)^+ b_f = b_f^\top L^+ b_f \le \sum_{l \in W(e)} \frac{w_l b_l^\top L^+ b_l}{w_l} \le \alpha \sum_{l \in W(e)} \frac{1}{w_l}$$

α-boundedness w.r.t. Laplacians

A multi-edge e is α-bounded w.r.t. a Laplacian L if

$$w_e b_e^\top L^+ b_e \leq \alpha$$

If all multi-edges in a graph is α-bounded w.r.t. its Laplacian L, then for each sampled multi-edge

$$b_f^\top SC(L,T)^+ b_f = b_f^\top L^+ b_f \leq \sum_{l \in W(e)} \frac{w_l b_l^\top L^+ b_l}{w_l} \leq \alpha \sum_{l \in W(e)} \frac{1}{w_l}$$

α-boundedness is (almost) preserved!

$$w_f = \frac{1}{\sum_{l \in W(e)} 1/w_l}$$

α-boundedness w.r.t. Laplacians

A multi-edge e is α-bounded w.r.t. a Laplacian L if

$$w_e b_e^\top L^+ b_e \leq \alpha$$

To achieve α-boundedness **initially**: split each edge to 1/α copies of multi-edges with α times its original weight

Recall we want $\|X_i\| \leq R$ roughly $R \approx \alpha$

Matrix Concentration

Suffices for $1/\alpha = O(\log^3 n)$ to achieve

$$L \approx_{O(1)} U_1^\top U_2^\top \cdots U_d^\top L_d U_d \cdots U_2 U_1$$

Aside: Matrix Martingale

Recall: sampling procedure is **unbiased**.

Additive view of Block Cholesky Factorization, generalized from **[KS ‘16]**

Suffices to set $1/\alpha = O(\log^2 n)$

Summing Up

Sparse Block Cholesky Factorization

Iterate until the remaining matrix has $O(1)$ vertices $\qquad O(\log n)$

Find a 5-DD subblock F to eliminate

Sample the Schur complement estimation created by eliminating F

Summing Up

Sparse Block Cholesky Factorization

#multi-edges = $O(m \log^2 n)$

Iterate until the remaining matrix has $O(1)$ vertices $O(\log n)$

Find a 5-DD subblock F to eliminate

Sample the Schur complement estimation created by eliminating F

each O(#multi-edges) work, O(log(#multi-edges)) depth w.h.p.

Total: $O(m \log^3 n)$ work and $O(\log^2 n)$ depth

Further Results

Generalizes to a wider range of matrices:

SDD, Magnetic Laplacian, HDD, Connection Laplacian, bDD

Direct Applications (all in parallel):

1. Approximate Schur Complement of any fixed subset
2. Spectral sparsification, leverage score estimation

Future Directions

$\widetilde{O}(\log n)$ depth?

Intermediate: $O(\log^2 n)$ construction, $\widetilde{O}(\log n)$ apply?

Directed Laplacian in polylog depth?

Simple algorithm in CONGEST?

Intermediate: EREW PRAM

Flow algorithms in parallel?

Thanks!