ETH zürich

Department of Mathematics
Computational Science Engineering

Semester Project Kick off meeting

Mingyuan Chi

21-09

Outline

1. Introduction
2. Background
3. Methodology
4. Experiment
5. Future work

Outline

Finite Element Analysis

Strong Form : $\nabla \cdot \sigma(\varepsilon) + f = \rho \frac{\partial^2 u}{\partial t^2}$

Weak form : $\int_\Omega \sigma : \nabla v + b \cdot v \mathrm{d}\Omega - \int_{\partial\Omega} (\sigma \cdot n) \cdot v \mathrm{d}S = \rho \int_\Omega \frac{\partial^2 u}{\partial t^2} \cdot v \mathrm{d}\Omega$

Galerkin discretilization : $K_{ij} = \int_\Omega \sigma(\sum u_i \phi_i) : \nabla \psi_j + b \cdot \psi_j \mathrm{d}\Omega \quad \phi_i = N_i, \psi_j = N_j$

- σ stress, it's a function of strain $\sigma(\varepsilon)$
  Lame's approximation:
  $\sigma = \lambda \mathrm{Tr}(\varepsilon) I + 2\mu\varepsilon \quad \lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}, \mu = \frac{E}{1+2\nu}$
- ε strain, $\varepsilon = \nabla u$
- : double contraction, $\sigma : \nabla v = \nabla v^\top \cdot \sigma \cdot \nabla v \quad \sigma \in \mathbb{R}^{d \times d}, \nabla v \in \mathbb{R}^d$
- bilinear form : $a(u, v) = \int_\Omega \sigma(\varepsilon) : \nabla v + b \cdot v \mathrm{d}\Omega$ and linear form : $\ell(v) = \int_{\partial\Omega} (\sigma \cdot n) \cdot \mathrm{d}S$

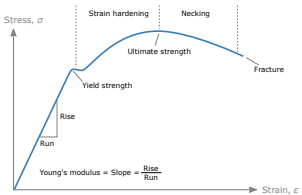


Figure: stress-strain relation

Linear Case

When **Dirichlet Boundary Condition** applied to mesh $\mathcal{M}$, $u_B, K, f \rightarrow u_I$

$$\begin{bmatrix} K_{II} & K_{IB} \\ K_{IB}^\top & K_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} f_I \\ f_B \end{bmatrix} \rightarrow K_{II} x_I = f_I - K_{IB} x_B$$

- $A_{\{II,BB,IB\}}$: Galerkin matrix for interior/boundary/interior-boundary connection
  - element matrix : $K_e = \int_\Omega B_e^\top D_e B_e \mathrm{d}\Omega$
  - B : Strain-Displacement Matrix, $\varepsilon = Bu = \frac{1}{2}(\nabla u + \nabla u^\top)$
    - Triangle Element : $B = \begin{bmatrix} \frac{\partial N_1}{\partial x} & 0 & \frac{\partial N_2}{\partial x} & 0 & \frac{\partial N_3}{\partial x} & 0 \\ 0 & \frac{\partial N_1}{\partial y} & 0 & \frac{\partial N_2}{\partial y} & 0 & \frac{\partial N_3}{\partial y} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial y} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial y} & \frac{\partial N_3}{\partial x} \end{bmatrix} \quad \varepsilon = \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ 2\varepsilon_{xy} \end{bmatrix}$
  - D : Material's Constitutive (Elasticity) Matrix, $\sigma = D\varepsilon = DBu$
    - Triangle Element : $D = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}$
- $u_{\{I,B\}}$: displacement for interior/boundary nodes
- $f_{\{I,B\}}$: applied force for interior/ boundary nodes

GNN view

Traditional Case:

$$Ku = f \Leftrightarrow u = \text{GNN}(A, f) \quad A_{ij} = K_{ij} > 0$$

considering **Dirichlet Boundary Condition**

$$K_{II}u_I = f_I - K_{IB}u_B \Leftrightarrow u_I = \text{GNN}(A_{II}, f_I - A_{IB}u_B)$$

Neumann Boundary Condition and Radiation Boundary can be coupled in the PDE form, which is simulated by the GNN

GNN is used as an inverse operator/sparse linear system solver

Traditional sparse linear system solver method: LUsparse, conjugated gradient .etc

Chanllege and Purpose

1. Theoretical proof of boundary condition GNN for FEM
2. End2end GNN framework applied to Dirichlet Boundary Condition
3. Better precision
4. Faster speed, larger scale (distributed, and even on disk)
5. Non-Lame's approximation

Outline

Graph Neural Network

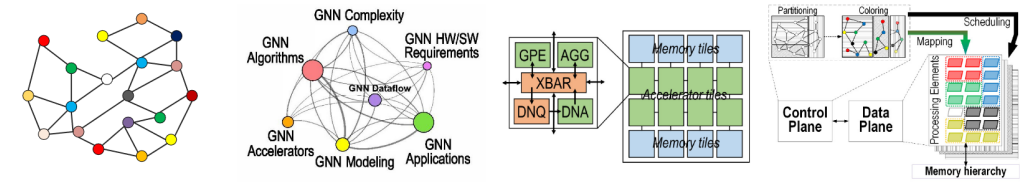


Figure: Graph Neural Network Survey: Algorithms, Application, Modeling, Accelerators [1]

Large Scale Graph Neural Network(1/4)

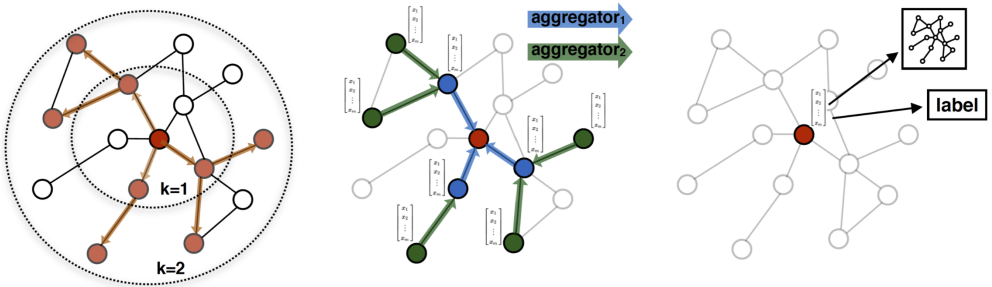


Figure: GraphSAGE:Sample K-hop neighbors to train as large graph [12]

Large Scale Graph Neural Network(2/4)

GraphSAINT: Sample random subgraph by weight to train as large graph [22]

- Aggregation : $\xi_v^{l+1} = \sum_{u \in \mathcal{N}(v)} \frac{\mathbf{A}_{u,v}}{\alpha_{u,v}} x_u^l \quad \alpha_{u,v} = \frac{p_{u,v}}{p_v}, p_{u,v} \propto \frac{1}{\text{deg}(u)} + \frac{1}{\text{deg}(v)}$
- Loss function : $\mathbb{E}[L] = \frac{1}{|\mathbb{G}|} \sum_{\mathcal{G}_s \in \mathbb{G}} \sum_{v \in \mathcal{V}_s} \frac{L_v}{\lambda_v} \quad \lambda_v = |\mathcal{V}| \cdot p_v$

Require: Training graph $G(V, E, X)$; Labels Y; Sampler SAMPLE
Ensure: GCN model with trained weights
1: **Pre-processing:** Setup SAMPLE parameters; Compute normalization coefficients α, λ.
2: **for** each minibatch **do**
3: $Gs(Vs, Es) \leftarrow$ Sampled sub-graph of G according to SAMPLE
4: GCN construction on Gs.
5: $\{y_v | v \in Vs\} \leftarrow$ Forward propagation of $\{x_v | v \in Vs\}$, normalized by α
6: Backward propagation from λ-normalized loss $L(y_v, y_v)$. Update weights.
7: **end for**

Large Scale Graph Neural Network(3/4)

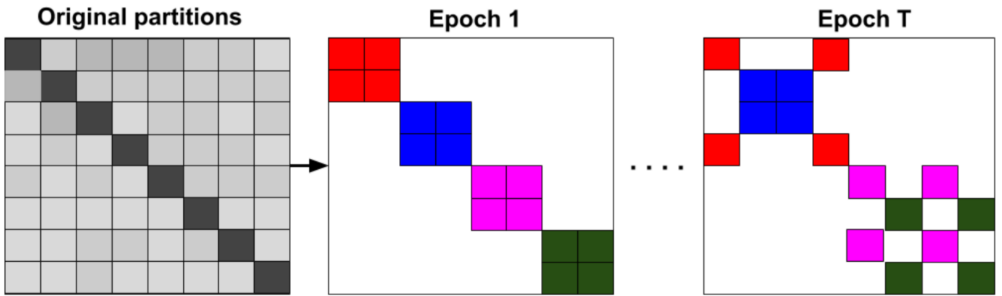


Figure: ClusterGCN:Partition Graph Manually rather than sample [12], commonly by METIS [14]

Large Scale Graph Neural Network(4/4)

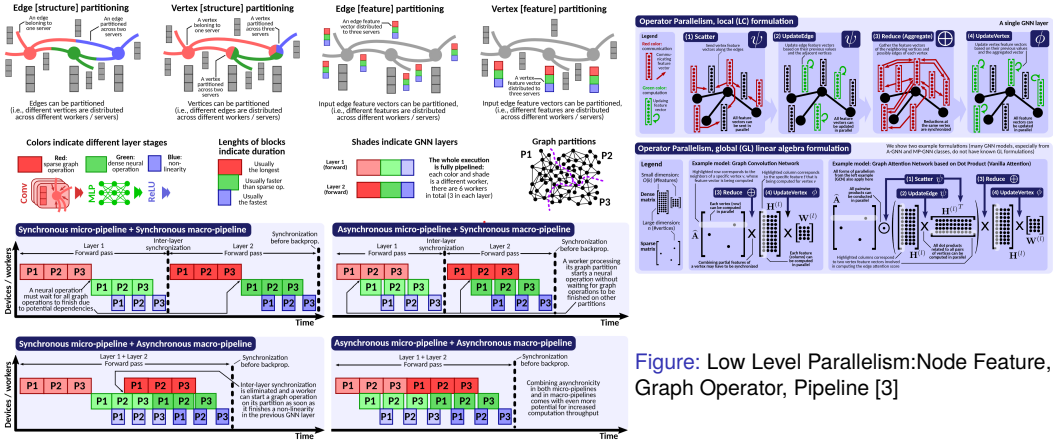


Figure: Low Level Parallelism:Node Feature, Graph Operator, Pipeline [3]

GNN for FEM(1/2)

- No Neighbor Explosion
- No super node

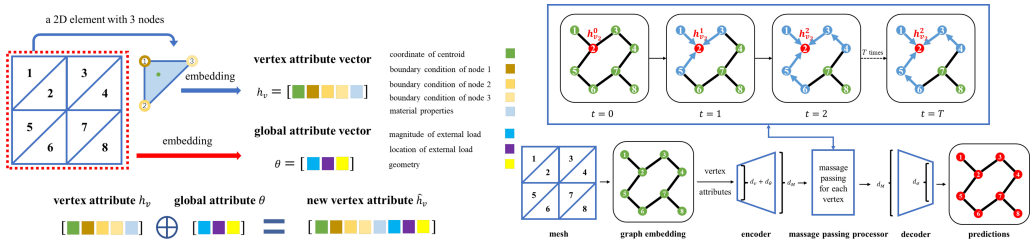


Figure: Encoder-Decoder structure [13]

- encoder-decoder structure
- element as node (FVM)

GNN for FEM(2/2)

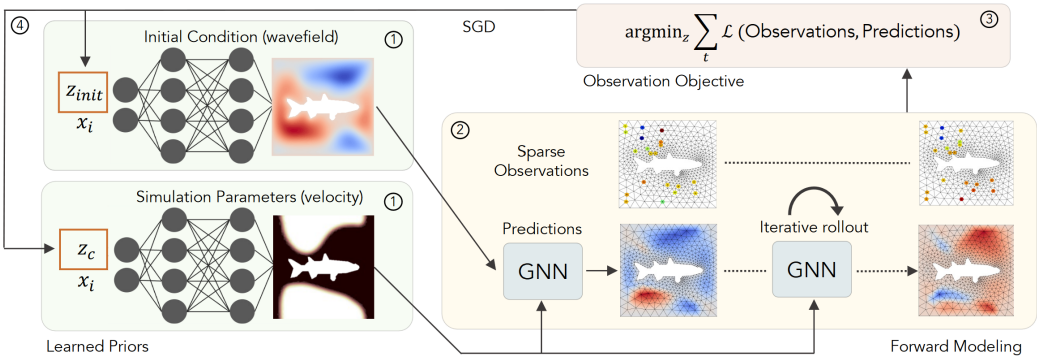


Figure: Inverse Problem [16]

- focus on inverse time sequencing problem
- data driven

GNN for FEM DBC(1/2)

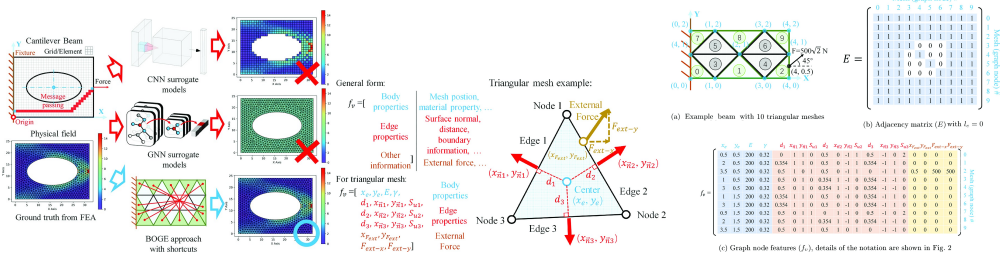


Figure: Fully Connected Solution: each $(v_i, v_j) \in \hat{\mathcal{E}} \quad \forall v_i \in \mathcal{M}_I \quad \forall v_j \in \mathcal{M}_B$, interior nodes and boundary nodes fully connected, element as node in graph [9]

GNN for FEM DBC(2/2)

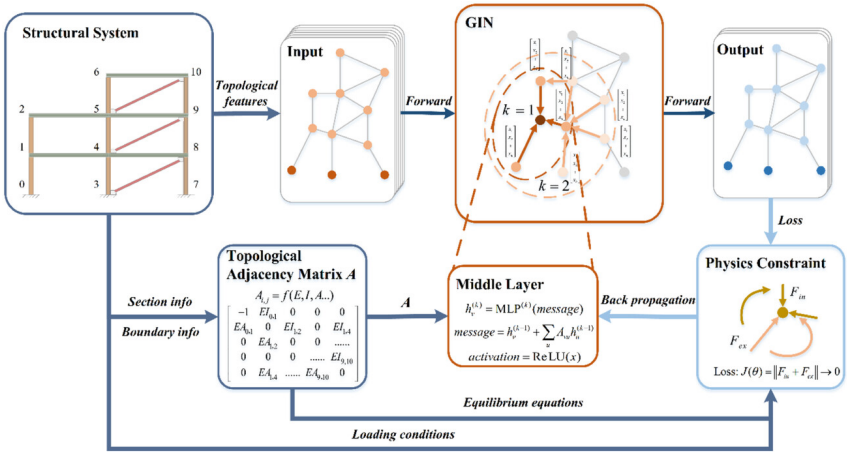


Figure: Physics Informed: $\|f_I + f_B\| = 0$, node as node in graph [20]

Graph Pooling(1/4)

Method	**Select**	**(Feature) Reduce**	(Edge) Connect
DiffPool [21]	$\mathbf{S} = \text{GNN}_1(\mathbf{A}, \mathbf{X})$	$\mathbf{X}' = \mathbf{S}^\top \cdot \text{GNN}_2(\mathbf{A}, \mathbf{X})$	$\mathbf{A}' = \mathbf{S}^\top \mathbf{A} \mathbf{S}$
MinCut [4]	$\mathbf{S} = \text{MLP}(\mathbf{X})$	$\mathbf{X}' = \mathbf{S}^\top \mathbf{X}$	$\mathbf{A}' = \mathbf{S}^\top \mathbf{A} \mathbf{S}$
NMF [2]	Factorize : $\mathbf{A} = \mathbf{W}\mathbf{H} \rightarrow \mathbf{S} = \mathbf{H}^\top$	$\mathbf{X}' = \mathbf{S}^\top \mathbf{X}$	$\mathbf{A}' = \mathbf{S}^\top \mathbf{A} \mathbf{S}$
LaPool [19]	$\begin{cases} \mathbf{V} = \lvert\mathbf{L}\mathbf{X}\rvert_d \\ \mathbf{i} = \{i \lvert \forall j \in \mathcal{N}(i) : \mathbf{V}_i > \mathbf{V}_j\} \\ \mathbf{S} = \text{SparseMax}\left(\beta \frac{\mathbf{X}\mathbf{X}_\mathbf{i}^\top}{\lVert\mathbf{X}\rVert \lVert\mathbf{X}_\mathbf{i}\rVert}\right) \end{cases}$	$\mathbf{X}' = \mathbf{S}^\top \mathbf{X}$	$\mathbf{A}' = \mathbf{S}^\top \mathbf{A} \mathbf{S}$
Graclus [7]	$S_k = \left\{ \mathbf{x}_i, \mathbf{x}_j \middle\vert \underset{j}{\text{argmax}} \left(\frac{\mathbf{A}_{ij}}{\mathbf{D}_{ii}} + \frac{\mathbf{A}_{ij}}{\mathbf{D}_{jj}} \right) \right\}$	$\mathbf{X}' = \mathbf{S}^\top \mathbf{X}$	METIS [14]
NDP [5]	$\mathbf{i} = \{i \lvert u_{max,i} > 0\}$	$\mathbf{X}' = \mathbf{X}_\mathbf{i}$	Kron [8]
Top-K	$y = \frac{\mathbf{X}\mathbf{p}}{\lVert\mathbf{p}\rVert}; \mathbf{i} = \text{top}_K(y)$	$\mathbf{X}' = (\mathbf{X} \odot \sigma(y))_\mathbf{i}$	$\mathbf{A}' = \mathbf{A}_{\mathbf{i},\mathbf{i}}$
SAGPool [15]	$y = \text{GNN}(\mathbf{A}, \mathbf{X}); \mathbf{i} = \text{top}_K(y)$	$\mathbf{X}' = (\mathbf{X} \odot \sigma(y))_\mathbf{i}$	$\mathbf{A}' = \mathbf{A}_{i,i}$

Table: Incomplete Summary of Graph Pooling [18, 11]

Graph Pooling(2/4)

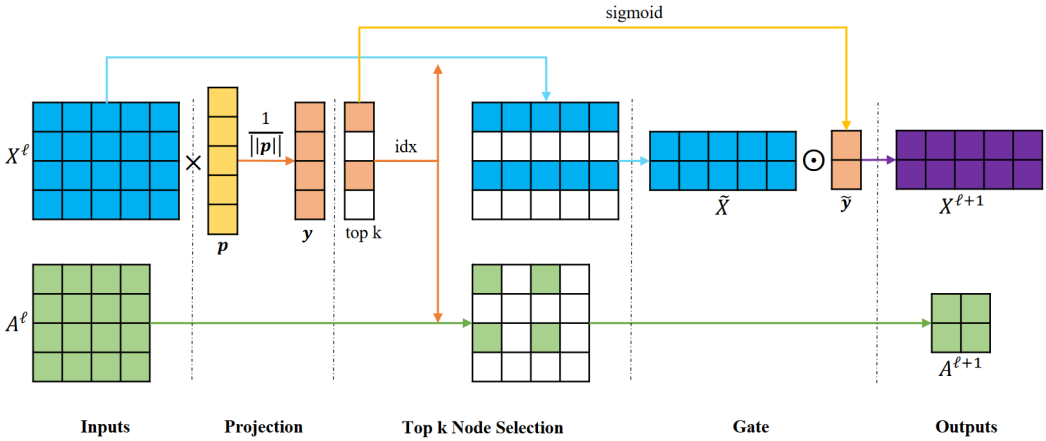
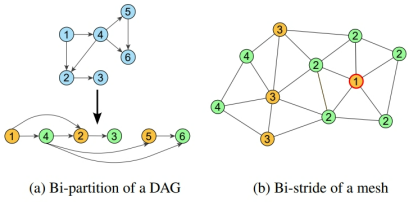


Figure: Naive Pooling [10]

Graph Pooling(3/4)



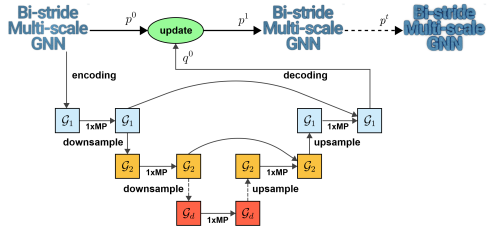
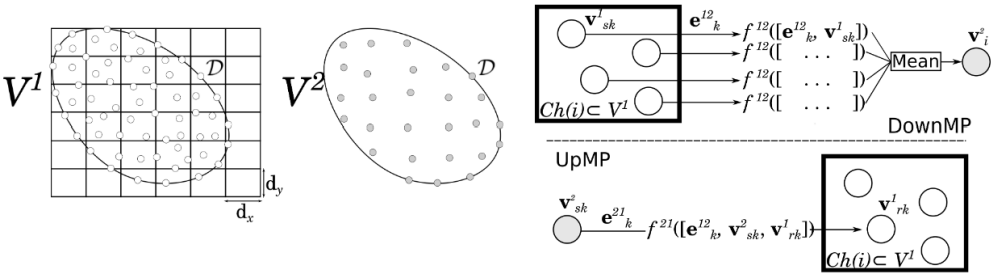


Figure: 2-Strided Pooling [6]

Graph Pooling(4/4)



(a) Building V^2 from V^1 (b) DownMP and UpMP layers

Figure: Rasterization Pooling [17]

- Also U-Net structure
- Looks like Particle To Cell method (widely used in N-body problem simulation)

Outline

Methodology

- GNN Introduction
- How FEM different boundary condition be tackled
- How our method work(Sampled Graph U Net)

Outline

Experiment

- Strain-Stress precision and baseline comparison
- Case study on airfoil example
- Parameter Sensitivity
- Speed/Memory Comparation(and compare with FEM)
- Topology Optimization? $z = \underset{z}{\text{argmin}} \sum_{e=1}^{N} (z_e)^p u_e^\top A_e u_e \quad z_e > 0$

Outline

Future work

- FEM for Structure Engineering Benchmark / Leaderboard?
- FEM for torch

References I

[1] Sergi Abadal, Akshay Jain, Robert Guirado, Jorge López-Alonso, and Eduard Alarcón. Computing graph neural networks: A survey from algorithms to accelerators. *ACM Comput. Surv.*, 54(9), oct 2021.

[2] Davide Bacciu and Luigi Di Sotto. A non-negative factorization approach to node pooling in graph convolutional neural networks. In *AI* IA 2019–Advances in Artificial Intelligence: XVIIIth International Conference of the Italian Association for Artificial Intelligence, Rende, Italy, November 19–22, 2019, Proceedings 18*, pages 294–306. Springer, 2019.

[3] Maciej Besta and Torsten Hoefler. Parallel and distributed graph neural networks: An in-depth concurrency analysis. *arXiv preprint arXiv:2205.09702*, 2022.

[4] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *International conference on machine learning*, pages 874–883. PMLR, 2020.

[5] Filippo Maria Bianchi, Daniele Grattarola, Lorenzo Livi, and Cesare Alippi. Hierarchical representation learning in graph neural networks with node decimation pooling. *IEEE Transactions on Neural Networks and Learning Systems*, 33(5):2195–2207, 2020.

[6] Yadi Cao, Menglei Chai, Minchen Li, and Chenfanfu Jiang. Efficient learning of mesh-based physical simulation with bi-stride multi-scale graph neural network. 2023.

[7] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE transactions on pattern analysis and machine intelligence*, 29(11):1944–1957, 2007.

[8] Florian Dorfler and Francesco Bullo. Kron reduction of graphs with applications to electrical networks. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 60(1):150–163, 2012.

[9] Xingyu Fu, Fengfeng Zhou, Dheeraj Peddireddy, Zhengyang Kang, Martin Byung-Guk Jun, and Vaneet Aggarwal. An finite element analysis surrogate model with boundary oriented graph embedding approach for rapid design. *Journal of Computational Design and Engineering*, 10(3):1026–1046, 03 2023.

[10] Hongyang Gao and Shuiwang Ji. Graph u-nets. In *international conference on machine learning*, pages 2083–2092. PMLR, 2019.

References II

[11] Daniele Grattarola, Daniele Zambon, Filippo Maria Bianchi, and Cesare Alippi. Understanding pooling in graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[12] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[13] Chunhao Jiang and Nian-Zhong Chen. Graph neural networks (gnns) based accelerated numerical simulation. *Engineering Applications of Artificial Intelligence*, 123:106370, 2023.

[14] George Karypis. Metis: Unstructured graph partitioning and sparse matrix ordering system. *Technical report*, 1997.

[15] Boris Knyazev, Graham W Taylor, and Mohamed Amer. Understanding attention and generalization in graph neural networks. *Advances in neural information processing systems*, 32, 2019.

[16] Yangfan Li, Jun Liu, Wenyu Liang, and Zhuangjian Liu. Towards optimal design of dielectric elastomer actuators using a graph neural network encoder. *IEEE Robotics and Automation Letters*, 8(10):6339–6346, 2023.

[17] Mario Lino, Chris Cantwell, Anil A Bharath, and Stathi Fotiadis. Simulating continuum mechanics with multi-scale graph neural networks. *arXiv preprint arXiv:2106.04900*, 2021.

[18] Chuang Liu, Yibing Zhan, Jia Wu, Chang Li, Bo Du, Wenbin Hu, Tongliang Liu, and Dacheng Tao. Graph pooling for graph neural networks: Progress, challenges, and opportunities. *arXiv preprint arXiv:2204.07321*, 2022.

[19] Emmanuel Noutahi, Dominique Beaini, Julien Horwood, Sébastien Giguère, and Prudencio Tossou. Towards interpretable sparse graph representation learning with laplacian pooling. *arXiv preprint arXiv:1905.11577*, 2019.

[20] Ling-Han Song, Chen Wang, Jian-Sheng Fan, and Hong-Ming Lu. Elastic structural analysis based on graph neural network without labeled data. *Computer-Aided Civil and Infrastructure Engineering*, 38(10):1307–1323, 2023.

[21] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31, 2018.

[22] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931*, 2019.

ETH zürich

The current project description is available at walkerchi.github.io/ETHz-SP/kickoff

D MATH

Mingyuan Chi
minchi@student.ethz.ch