SCARA: Scalable Graph Neural Networks with Feature-Oriented Optimization

Ningyi Liao*[1], Dingheng Mo*[1], Siqiang Luo[1], Xiang Li[2], Pengcheng Yin[3]

[1]Nanyang Technological University, Singapore
[2]East China Normal University, China [3]Google Research, USA

CONTRIBUTION

SCARA framework: Graph propagation decoupled as one-time precomputation, efficient GPU batch training and inference.

FEATURE-PUSH algorithm: Feature-oriented fast vector-based propagation, sub-linear precomputation complexity.

FEATURE-REUSE algorithm: Efficient result reuse among multiple features, further save computation with guaranteed precision.

Performance Evaluation: 10-100× faster precomputation, low RAM overhead, able to process billion-scale graph Papers100M in 100 seconds.

PROPOSED FRAMEWORK

Pre-Propagation Decoupling Framework

① Propagation Precomputation

$$\boldsymbol{H}^{(0)} = \boldsymbol{P} = \sum_{l=0}^{\infty} \alpha(1-\alpha)^l \tilde{\boldsymbol{A}}_{(r)}^l \cdot \boldsymbol{X}$$

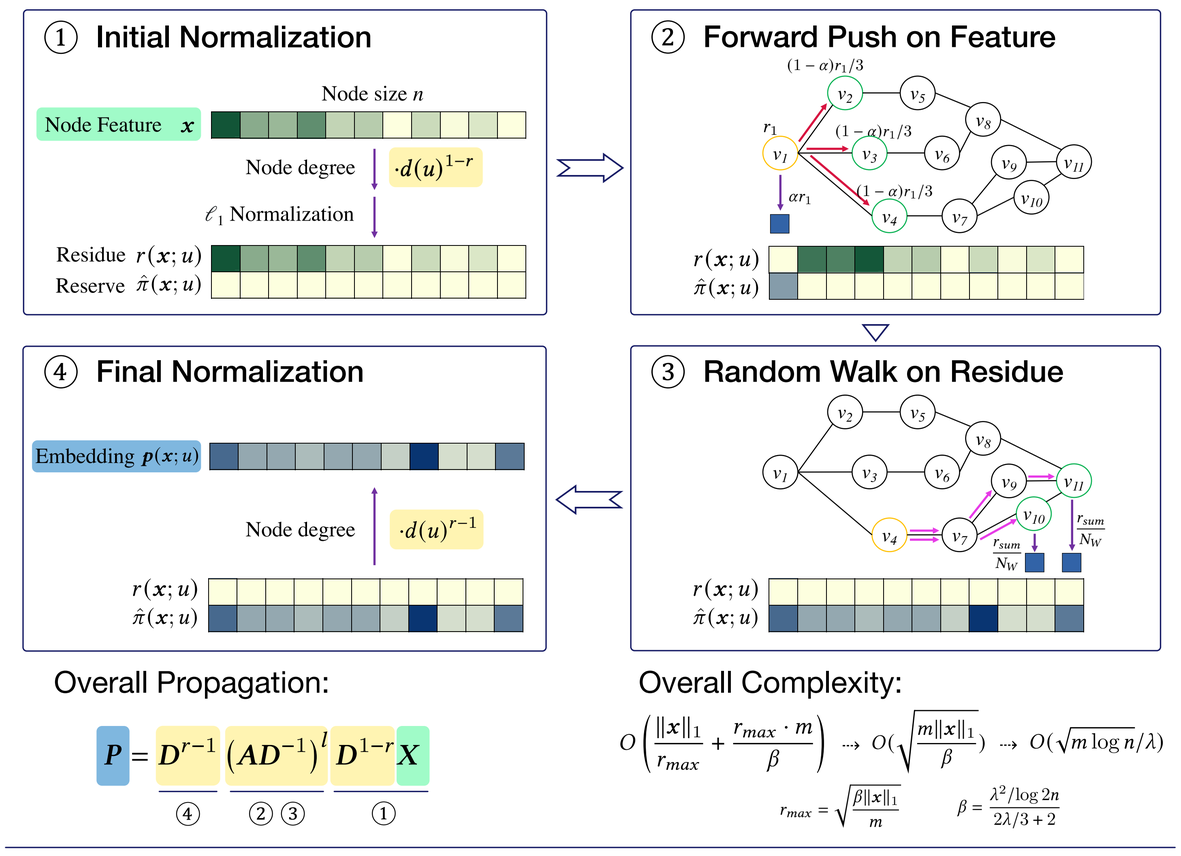
Graph Embedding ⇠ Graph Adjacency × Node Feature

② Feature Transformation

$$\boldsymbol{H}^{(l+1)} = \sigma\left(\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)}\right), \quad l = 0, 1, \cdots, L-1$$

Layer Representation ⇠ Previous Representation × Learnable Weights

FEATURE-PUSH



Overall Propagation:

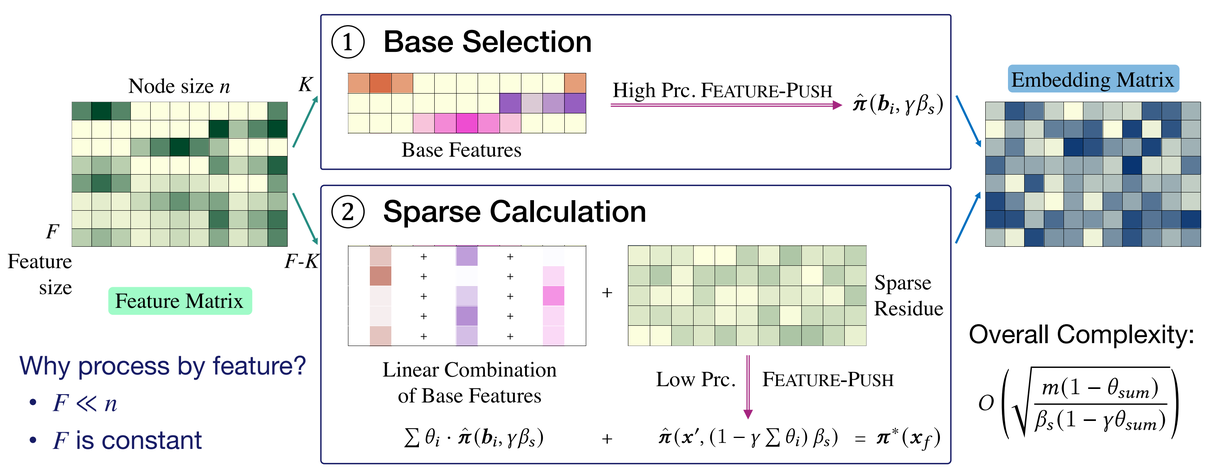
$$\boldsymbol{P} = \underbrace{\boldsymbol{D}^{r-1}}_{④}\underbrace{\left(\boldsymbol{A}\boldsymbol{D}^{-1}\right)^l}_{②\,③}\underbrace{\boldsymbol{D}^{1-r}\boldsymbol{X}}_{①}$$

Overall Complexity:

$$O\left(\frac{\|\boldsymbol{x}\|_1}{r_{max}} + \frac{r_{max}\cdot m}{\beta}\right) \dashrightarrow O\left(\sqrt{\frac{m\|\boldsymbol{x}\|_1}{\beta}}\right) \dashrightarrow O(\sqrt{m\log n}/\lambda)$$

$$r_{max} = \sqrt{\frac{\beta\|\boldsymbol{x}\|_1}{m}} \qquad \beta = \frac{\lambda^2/\log 2n}{2\lambda/3+2}$$

FEATURE-REUSE



Why process by feature?

- $F \ll n$
- F is constant

Overall Complexity:

$$O\left(\sqrt{\frac{m(1-\theta_{sum})}{\beta_s(1-\gamma\theta_{sum})}}\right)$$

Paper: vldb.org/pvldb/volumes/15#issue-11

Code: github.com/gdmnl/SCARA-PPR

RELATED WORK & COMPLEXITY

Vanilla / Sampling Model	Learn Time	Memory
GCN [1]	$O(ILmF + ILnF^2)$	$O(LnF)$
	☹ *Iterative Propagation: not scalable to m*	☹ *No GPU Utilization*

Post-Propagation Model	Precomputation Time	Train Time	Memory
PPRGo [2]	$O(m/r_{max})$	$O(IKnF + ILnF^2)$	$O(n/r_{max})$
	☹ *Not Scalable Propagation*		☹ *High Overhead*

Pre-Propagation Model	Precomputation Time	Train Time	Memory
GBP [3]	$O(LF\sqrt{Lm\log(Ln)}/\epsilon)$	$O(ILnF^2)$	$O(nF)$
	☹ *Not Feature-Oriented*	☹ *Inefficient RAM Utilization*	
SCARA (ours)	$O(F\sqrt{m\log n}/\lambda)$	$O(ILnF^2)$	$O(nF)$
	☺ *Only sublinear to n*	☺ *Efficient GPU & RAM Utilization*	

EXPERIMENTS

Performance Evaluation

Time Efficiency: fastest overall learning time, 10-100× faster precomputation, comparable or better training and inference clock time

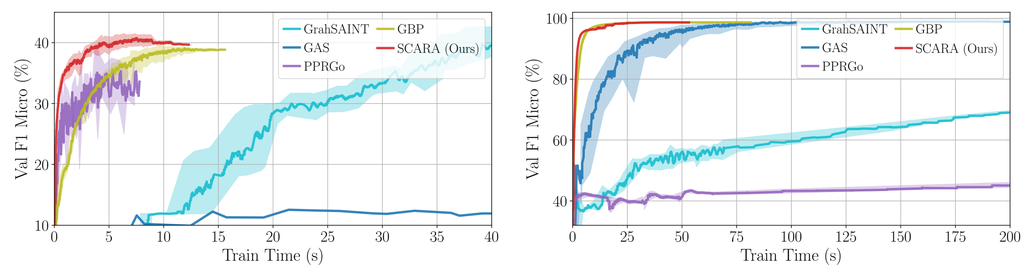
Memory Efficiency: Paper100M with 64GB without OOM

Effectiveness: similar or better F1-score, fast and stable convergence

Dataset	Nodes n	Edges m	Features F	Dataset	Nodes n	Edges m	Features F
Reddit	232,965	114,615,892	602	Papers100M	111,059,956	1,615,685,872	128

Transductive	Reddit						Papers100M					
	Learn	(Pre.	+ Train)	Infer	Mem.	F1	Learn	(Pre.	+ Train)	Infer	Mem.	F1
GraphSAINT	51.5	(–	51.5)	26.1	11.1	30.7 ±3.0	–	–	–	–	OOM	–
GAS	3563	(–	3563)	**0.1**	14.6	38.0 ±0.2	–	–	–	–	OOM	–
PPRGo	163	(157 +	4.8)	74.1	8.0	31.0 ±1.7	–	–	–	–	OOM	–
GBP	1891	(2127 +	16.3)	6.2	8.4	39.2 ±0.3	–	–	–	–	OOM	–
SCARA (ours)	**12.0**	(1.8 +	10.6)	4.8	**4.7**	**40.3 ±0.7**	**1471**	(83.5 +	1388)	**2.8**	**63.7**	**35.5 ±0.8**

Convergence Curve



Reddit: Transductive Learning PPI: Inductive Learning

Ablation Study

FEATURE-REUSE:

Up to 1.6× speed-up

No effect on precision

	Feature	$F = 100$	$F = 200$	$F = 400$	$F = 602$
Pre. Time	w/o REUSE	0.46	0.93	1.83	2.84
	w/ REUSE	0.35	0.67	1.30	1.85
	Speed-up	133%	138%	141%	155%
Accuracy	w/o REUSE	27.7	31.9	37.0	40.5
	w/ REUSE	27.8	31.7	36.7	40.3
	Δ	+0.1	−0.2	−0.3	−0.2

REFERENCES

[1] Thomas N Kipf & Max Welling. "Semi-supervised classification with graph convolutional networks". ICLR 2017.
[2] Aleksandar Bojchevski et al. "Scaling graph neural networks with approximate PageRank". SIGKDD 2020.
[3] Ming Chen et al. "Scalable graph neural networks via bidirectional propagation". NeurIPS 2020.
[4] Hanqing Zeng et al. "GraphSAINT: graph sampling based learning method". ICLR 2019.
[5] Matthias Fey et al. "GNNAutoScale: scalable and expressive graph neural networks via historical embeddings". ICML 2021.

ACKNOWLEDGMENTS

This research is supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 1 Seed (RS05/21) and in part by NTU startup grant. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore. Xiang Li is supported by Shanghai Pujiang Talent Program (Project No. 21PJ1402900) and Shanghai Science and Technology Committee General Program (Project No. 22ZR1419900). We also thank the anonymous reviewers for their valuable feedback.

* Both authors contributed equally to this research.